Knowledge Acquisition

William P. Wagner
Villanova University

GLOSSARY

automated knowledge acquisition techniques A set of tools and techniques designed to overcome the limitations of human-to-human interactions. Strategies for automating the knowledge acquisition process may range from using high-level expert system shells, which enable an expert to enter his or her knowledge directly into the computer, to programs that interview and prompt the expert.

contextual factors These are aspects of the organizational context, human and physical factors that may have an impact on the quality of knowledge acquisition episodes.

heuristics Derived from the Greek verb *heurisko,* meaning "to find or discover," a heuristic is knowledge discovered by domain experts in the process of performing their tasks. Usually expressed as an If-Then rule, a heuristic is similar to a "rule of thumb" in that it is knowledge that is informal and a matter of judgement.

knowledge acquisition (KA) A process whose goal is the elicitation and representation of knowledge for developing artificial intelligence applications such as expert systems.

knowledge engineer Artificial intelligence specialist responsible for the development of a knowledge-based application. Duties usually include knowledge elicitation, representation, and coding.

manual knowledge acquisition techniques A set of tools and techniques typically drawn from sociology and psychotherapy to assist a knowledge engineer in eliciting and representing the knowledge of one or more human experts.

protocol analysis One popular knowledge acquisition technique that involves the creation of a protocol or record of an expert's decision-making processes. The expert is encouraged to "think aloud" while working through a problem; this is recorded and serves as the protocol. This protocol is then analyzed by a knowledge engineer and converted to an intermediate or final knowledge representation.

KNOWLEDGE ACQUISITION (KA) is a goal-directed process that involves the subprocesses of elicitation and representation of knowledge. This can include a wide variety of knowledge sources, such as text, video, data, and one or more human experts. KA techniques can range from machine learning and induction to unstructured interviews. KA processes may be involved in the development of any knowledge-based applications.

I. KNOWLEDGE ACQUISITION

The study and development of expert systems is one of the more mature fields within artificial intelligence (AI). These systems have either been developed completely in-house, purchased as proprietary software, or developed using an expert system shell. The most commonly cited problems in developing these systems are the unavailability of both the experts and the knowledge engineers and difficulties with the rule extraction process.

Within the field of AI this has been called the "knowledge acquisition" problem and has been identified as one of the biggest bottlenecks in the expert system development process. Simply stated, the problem is how to efficiently acquire the specific knowledge for a well-defined problem domain from one or more experts and represent it in the appropriate computer format.

Because of what has been called the "paradox of expertise," experts have often proceduralized their knowledge to the point where they have difficulty in explaining exactly what they know and how they know it. However, new empirical research in the field of expert systems reveals that certain KA techniques are significantly more efficient than others in different KA domains and scenarios. Adelman, then Dhaliwal and Benbasat, and later Holsapple and Wagner identified six sets of determinants of the quality of the resulting knowledge base.

1. Domain experts
2. Knowledge engineers
3. Knowledge representation schemes
4. Contextual factors
5. Knowledge elicitation methods
6. Problem domains.

Each of these determinants has its own body of research and implications for developing knowledge-based applications.

Conceptions of KA phenomena are complicated tremendously when we consider all the possible sources and combinations of expertise, the nature of the entity that elicits and structures this expertise, and all the different factors that may have a moderating influence on the process. When all these issues are taken into consideration, it becomes apparent that the classic KA scenario, where a knowledge engineer attempts to elicit and structure the knowledge of a human expert, is a complex phenomenon with respect to the number of parameters involved.

A. Knowledge Acquisition Processes

KA is mentioned as a problem within fields as diverse as machine learning, natural language processing, and expert systems research. However, the most typical KA scenario is where a knowledge engineer engages in an unstructured interview with one or more human experts. After these initial sessions, the engineer may use the results of these sessions to develop tools for structuring more in-depth KA sessions, such as focused lists of questions, simulations, or even expert system prototypes to use as the basis for later sessions. Researchers have used a systems analytic approach to modeling the KA process and have arrived at a variety of models that are generally in agreement. The KA process is typically described as being iterative with loops that depend upon the size of the system to be built, the depth and breadth of the task to be supported, and the quality of the knowledge as it is acquired. Steps in the process are identified as:

1. Conduct an initial unstructured interview.
2. Knowledge engineer analyzes results of interview.
3. Knowledge engineer represents the resulting knowledge.
4. Experts test, and their comments on the system performance are recorded.
5. If the knowledge base is complete, then stop.
6. Acquire the missing knowledge from an expert.
7. Represent the missing knowledge in the knowledge base.
8. Return to step 4 until finished.

In reality, KA often involves a number of different KA techniques used in multiple KA sessions or episodes with multiple knowledge sources including texts and various human experts. A classic example of how complicated KA can be in reality is the case of how the ExperTAX system was developed by Coopers & Lybrand. This classic case recorded by Shpilberg et al. describes an expert system that was designed to help guide staff accountants out in the field collecting tax information. This case is somewhat unique even today in that it details the overall activities involved in development and is not limited to illustrating specific KA techniques used or novel features of the system.

As is usually the case, the development of ExperTAX began with an unspecified number of unstructured exchanges with multiple experts (20 experts participated overall). After this, the knowledge engineering team decided to set up a simulated tax information gathering case with three different accountants playing roles. The team videotaped this session and several subsequent variations on it. Other interactive KA sessions were held using a computer to help in the design of the user interface. After these different sessions, a prototype of ExperTAX was developed. This then served as the basis of all the KA sessions that followed with individual experts, leading to significant additions. The whole development process spanned about 2 years and almost 6 person-year's labor. This particular case describes in unusual detail how varied KA can be in reality with respect to the combinations of KA techniques and participants at each stage of the

project. Because KA is such a mix of activities, in actual practice it has been difficult for researchers to evaluate how these activities may be best implemented.

II. KNOWLEDGE ACQUISITION TOOLS AND TECHNIQUES

KA tools and techniques fall into two general categories: manual and automated. Manual KA techniques are derived mainly from psychology and involve a human knowledge engineer interacting with or observing one or more domain experts. In an effort to increase the efficiency of the KA process, researchers from the field of computer science wrote programs that would enable the computer to interview the expert and help in the structuring of the resulting expertise. This form of automated KA might also encompass some of the machine learning techniques from AI, such as neural networks and genetic algorithms, which may or may not require supervision in the learning process.

The role that the human knowledge engineer plays in the KA process varies considerably depending on the particular KA technique or method used. In some cases it may be appropriate for the knowledge engineer to become an apprentice to the expert or to participate somehow in the actual problem-solving process. In other cases it may be better for the knowledge engineer to conduct an unstructured interview or to simply observe the expert perform a given task.

A. Manual Knowledge Acquisition Tools and Techniques

Many different techniques have been developed especially for knowledge engineers in these different situations or have been drawn from existing research in fields such as psychology. Of these techniques, it should not be surprising that a survey found that the most commonly used knowledge elicitation technique was the "unstructured interview," where the knowledge engineer asks general questions and hopes for the best. However, each technique requires different abilities from the knowledge engineer and the knowledge source and allows a different set of knowledge representations to be used.

Although the "manual" KA techniques described here are certainly the most common used today, they are certainly not without their problems. Not only do they require an enormous amount of time and labor on the part of both the knowledge engineer and the domain expert, but they also require the knowledge engineer to have an unusually wide variety of interviewing and knowledge representation skills in order for them to be successful.

Despite being the most commonly used KA technique, it is difficult to describe the unstructured interview as a true technique, since as its name implies it is just a wandering conversation between the expert and the knowledge engineer. Even though, it still has a valuable place in the knowledge engineer's bag of tools since it allows the greatest possible freedom for the knowledge engineer and expert alike to explore the topic. Many researchers have documented and described its usage, although researchers in the field usually downplay its value as a real tool. In fact, one field study of KA techniques cited the use of the unstructured interview as one of the biggest failings of knowledge engineers who were attempting to develop expert systems.

This lack of detail regarding KA techniques used in reported expert systems cases seems to indicate that some developers assume that an unstructured interview is the only way to elicit an expert's knowledge. However, an unstructured interview can be helpful both in acclimating the expert to the development process and in helping the knowledge engineer learn general ideas about the problem domain. It serves the additional purpose of building rapport between the system developer and the human source. Unfortunately, this has been shown by researchers such as Burton *et al.* to be a time-consuming and inefficient process.

The notion of the structured or "focused" interview has its roots in psychotherapy research. Interviews may be structured by developing a list of questions in advance. In addition, the allowed activities of the interviewer are also carefully specified. The implied hypothesis is that by providing structure to the interview it can be made more efficient. Psychologists developed other interviewing techniques and tools which were designed to structure the interview process and have been, in turn, generally applied to the KA problem. These techniques can often be applied to situations where the expert is being interviewed while actually performing a task. Tasks used to structure the interview may also be simulated or reconstructed using case studies or scenarios or simply may be generated from the expert's own past experience. Techniques most commonly discussed in the literature include protocol analysis, psychological scaling, and card sorting.

Some simple structuring can be given to the interview process by having the expert perform a particular

task while the knowledge engineer asks questions freely. The task may be typical of the problem-solving situation which the knowledge engineer wishes to explore or it may be a special case identified in earlier sessions. The simplest task the knowledge engineer could give the expert could be to prepare a brief lecture designed to introduce someone to the particular problem domain. This type of task might be more appropriate for early KA sessions, whereas the special task might be better for when the knowledge engineer is more familiar with the particular domain.

Protocol analysis is one of the most frequently mentioned elicitation techniques in the KA literature. A survey of KA techniques in actual usage by Cullen and Bryman found protocol analysis to be second only to unstructured interviews in actual usage. Basically, subjects are asked to think aloud while solving a problem or making a decision. These verbalizations are usually taped and then transcribed. The transcription is then analyzed using a particular coding scheme. The transcript itself is termed a "protocol" and may be used to refer to a word-for-word record or a summary of the major points. Whatever the form of the protocol, it should enable the knowledge engineer to easily access, index, and code specific pieces of information.

Depending on the problem domain, it may be desirable to also generate "motor" protocols or even "eye-movement" protocols to more clearly understand an expert's performance of a task. Motor protocols require the expert's physical movements to be closely observed and noted by the knowledge engineer. This may only be appropriate for acquiring certain types of expertise. However, all these protocols can be thought of as being either "concurrent" or "retrospective." Concurrent protocols are records of the expert's thought processes during the same time he or she is solving a problem, while retrospective protocols are records of the expert's review of his or her verbalizations after the task is completed. It may be appropriate for a knowledge engineer to use a retrospective protocol when it is felt that the task of verbalization may interfere with the expert's performance of the actual task.

The actual transcript of the expert's thoughts, while faced with a given task such as the transcript above, is called the protocol. This transcript may, in turn, be translated by the knowledge engineer into a more formal protocol that attempts to summarize the major points in a format designed for easy review. Once the protocol has been worked into the desired format, the actual analysis of such a protocol by a knowledge engineer can begin. Knowledge engineers then may use one of many possible intermediate representation schemes to map out the expert's processes. This helps the knowledge engineer to uncover the decision rules used by the expert, classifying them further into different types of rules. So in analyzing the protocol, the knowledge engineer may find that some rules are general rules, other rules may be dependent on the results of prior decisions, and yet other rules may be totally distinct. These rules can then be refined further by either the domain expert or even another expert before they are implemented in the final system.

Protocol analysis has become popular as a KA tool because it forces the expert to focus on a specific task or problem without interruptions from the knowledge engineer. It forces the expert to consciously consider the problem-solving process and, therefore, may be a source of new self-understanding. It is also very flexible in that many different types of tasks (simulations, special cases, etc.) may serve as a basis for the protocol. Having a record encourages the knowledge engineer to target topics and possibly develop further structured interviews around missing steps in the process. It also allows the knowledge engineer to exercise a great degree of flexibility in the choice of analysis used to structure the protocols. Also, on a practical level, protocol analysis requires little equipment or special training for the knowledge engineer.

The main disadvantage to protocol analysis is the very necessity of forcing the expert to verbalize his or her actions. It is often the case that expertise has become so proceduralized that the expert is either unable to express it or is completely unaware of it. As mentioned previously, this phenomenon is commonly referred to as the paradox of expertise and is one of the major motivations for research in the field of KA. Not only may they be unaware of their problem-solving methods, but they may actually verbalize them incorrectly and thus introduce error or bias into the resulting system. If the knowledge engineer uses special or difficult test cases as the basis for the interview, the expert may be uncomfortable trying to verbalize the problem-solving process. So the success of protocol analysis may also depend on the personality of the expert and, in turn, their ability to be introspective and verbalize thought processes. In addition, generating protocols can be a very time-consuming process, and they may result in more data than the knowledge engineer can efficiently handle.

While protocol analysis involves little interaction between the expert and the knowledge engineer, several techniques have been suggested which require the knowledge engineer to actively participate in the problem-solving process. These techniques capitalize on the idea that the knowledge engineer must be-

come somewhat of an expert in order to translate the expert's knowledge into a machine representation. Thus, the interview may be used as a tutorial where the expert delivers a lecture, which the knowledge engineer may paraphrase or use to solve similar problem scenarios. The knowledge engineer may also play the role of an apprentice and actively participate in the expert's problem-solving process. Making the knowledge engineer become like the expert is certainly the most time-consuming approach to KA, but ensures the highest quality resulting system.

Other interviewing techniques, which have been proposed in the literature, have been drawn directly from psychology. These include multidimensional/psychological scaling, network scaling, cluster analysis, and card sorting. Many of these techniques combine elicitation and structuring aspects and thus are difficult to consider as simply "elicitation" or "structuring" techniques. They are considered to be more objective than more traditional interviewing methods and therefore may be especially useful in the later refinement stages of the KA process. Burton et al.'s empirical comparison of elicitation techniques supports this contention somewhat in that it found that nontraditional techniques such as card sorting performed better than protocol analysis and interviewing.

A number of different techniques fall under the heading of what may be called "psychological scaling" techniques. These include multidimensional scaling, network scaling, and hierarchical cluster analysis. Generally speaking, experts are asked to rate the similarity of different objects (usually chosen beforehand) and this rating is represented as a distance on a seven-point scale ranging from no similarity to completely similar. The purpose of this is to discover the expert's rank ordering of objects within a problem domain.

The most complex method within this group is probably multidimensional scaling (MDS). It is based on the use of the least-squares method to fit the elicited data. A grid of data is obtained by comparing the similarity of a set of key variables or concepts on a scaled number of different dimensions. This is supposed to give an overall picture of the space in which the objects lie. The relative location of the variables or concepts in the different dimensions is then inferred using the least-squares method. However, use of a scaling technique requires that both the concepts and the dimensions be identified beforehand. This is most likely done in an earlier interview. Thus, this technique should probably be considered more of a structuring tool than an elicitation tool. It does force the expert to quantify the relatedness among a set of concepts on a set of dimensions. Because the various scaling techniques are quantitative, they are more suited to use in automated KA programs such as AQUINAS, KITTEN, KSSO, PATHFINDER, and PLANET.

Card or concept sorting techniques are also used to help structure an expert's knowledge. As its name implies, it involves having the knowledge engineer write the names of previously identified objects, experiences, and rules on cards which the expert is asked to sort into groups. The expert describes for the knowledge engineer what each group has in common and the groups can then be organized to form a hierarchy. Like MDS, some empirical research by Burton et al. suggests that card sorting may be a more efficient elicitation technique than some of the more traditional techniques such as protocol analysis or interviewing. It has been used with some success to develop applications described in the literature. It is also a tool that can be easily implemented on a computer as an automated KA tool.

B. Automated Knowledge Acquisition Tools and Techniques

As pointed out in the general discussion of KA processes, KA is not limited to just the acquisition of expertise by a human knowledge engineer. Because of the many problems associated with using humans as KA agents, much of the current research thrust has been directed at developing tools and techniques that can be used to automate the KA process. Strategies for automating the KA process may range from using high-level expert system shells, which enable an expert to enter his or her knowledge directly into the computer, to programs that interview and prompt the expert. These programs use many of the same tools and techniques that a knowledge engineer would use in a typical face-to-face KA session.

In a very real sense a computerized acquisition mechanism plays the same role as the knowledge engineer, but brings a different set of abilities to bear on the problem. This section seeks to examine the research on what the characteristics of computer-based mechanisms are and how they have been implemented into the KA process as a whole. As expert system projects grew in scope and application, the problem of how to acquire the domain knowledge became more of an obstacle. Anecdotal evidence in the literature suggest that a basic prototype of a medium-sized expert system could take anywhere from 6–24 months to build, largely due to problems of accessing, eliciting, expressing, and representing the appropriate

expertise. Thus, the cost in terms of the time of experts and knowledge engineers alike of building even simple applications was the principal motivation in trying to streamline the process via automation.

Basically, automation strategies approach the KA bottleneck in two different ways. The first, as mentioned previously, is to design a system that allows the role of the knowledge engineer to be collapsed into that of the expert. This can be done by installing a sophisticated natural language interface on an expert system shell, thus enabling the expert to enter his or her knowledge directly into the computer. Another way of doing this is to allow the computer to induce the expert's knowledge based upon his or her performance on simulated problems or cases. These are chosen to be representative of the desired problem domain. These latter types of automated KA tools are directly related to work in machine learning and induction. In both cases, automating parts of the KA process are designed to turn the role of the knowledge engineer into more of a facilitator rather than interacting directly with the domain expert.

The second approach is at the other end of the automated KA spectrum. This strategy replaces the knowledge engineer by designing a computer program that would perform the function of interviewing the expert. Tools of this type usually borrow interviewing techniques used by humans and simply make the computer a proxy for the knowledge engineer. These tools include expert system shells or environments, simulation programs, and automated KA "workbenches."

Sophisticated expert system development "shells" or "environments" have been developed within the last decade that can be viewed as automated KA tools. The original motivation for developing the expert system shell was to support the rapid prototyping approach to developing expert systems. The heart of a typical shell is the generalized inference engine capability. This general inference engine gives the system created with the shell the basic ability to reason about a wide variety of problems. Early shells, however, required sophisticated hardware in order to run and were designed with the experienced programmer or AI expert in mind.

Easier-to-use shells or environments were later developed which were able to provide more KA support for either the domain expert or the knowledge engineer. Typically, they could be run on a personal computer (PC), were independent of particular problem domains, contained a general inference engine, and had more sophisticated user interfaces. Examples of common shells might be Guru, EXSYS, MacSMARTS, and VPExpert. Some of the more sophisticated expert system environments have the capability to include and access text, database, spreadsheet, and graphics functions within a rule set. In addition, easy-to-use rule set managers require little expertise to use and thus serve to facilitate the elicitation and representation phases of the KA process.

As the complexity of the KA problem grew, it became clearer that knowledge engineers might need a wide variety of different tools in order to successfully elicit and structure an expert's knowledge when developing an expert system. Where some KA tools may assist the knowledge engineer in early phases, they may not be of much help in translating the knowledge into the appropriate machine representation. KA workbenches or environments were designed to support a variety of elicitation and representation techniques to be combined in a KA session by a knowledge engineer. Thus, actual KA workbenches tend to vary widely in terms of the different tools and KA methods they support. Some are based on specific interviewing techniques (e.g., repertory grids), while others attempt to provide an integrated KA environment with many different tools.

Probably the most cited example of a KA workbench is that developed for use within Boeing's information services division. It was designed to run on a LISP machine and is based on a repertory grid technique. By forcing the expert and/or knowledge engineer to develop a repertory grid, it helps to decompose problems into simpler subproblems. It is also capable of integrating input from multiple experts and helps in making finer distinctions between objects.

Another workbench, KRITON, supports a variety of KA methods. KRITON was designed to support KA tasks via automated interviewing, text analysis, and protocol analysis. It combines repertory grid interviewing and protocol analysis to elicit and capture the domain expert's knowledge. The repertory grids have been automated and allow the expert to interact with the workbench itself. Procedural knowledge may then be acquired using protocol analysis, and the content of the transcribed protocol is broken down into semantic segments and further analyzed by the system. This text analysis feature examines text for the frequency of certain words for a problem domain. Using keywords found in the protocol, KRITON attempts to generate basic propositions about the domain that can be used as the basis for later KA sessions.

The most common approach appears to be to program the automated KA system so that it serves to interview the domain expert much as a knowledge engineer would. At least 19 such systems have been

designed with this facility. The most successful are AQUINAS, ETS, KRITON, KSSO, and TEIRESIAS. Other tools have been constructed which serve to automatically record and/or analyze the transcripts from experts thinking out loud while they perform a specific task. Tools that support some portion or all of the protocol analysis function include KRITON, LAPS, and MEDKAT.

These techniques have been expanded in several cases so that they may be employed in cases involving multiple experts. AQUINAS, ETS, MEDKAT, KITTEN, and KSSO may all be used to elicit and analyze knowledge from multiple sources using repertory grids. The knowledge of multiple experts may also be elicited via the use of the Delphi method. Some of the automated KA workbenches that support the Delphi include AQUINAS and KRITON. Other tools such as the analytic hierarchy process and group decision support systems (DSSs) may be appropriate for eliciting and structuring the knowledge of multiple experts, but have not been used extensively.

C. Knowledge Acquisition Tools and Techniques for Multiple Experts

As part of the push to expand the range of application for expert systems, some effort has been directed at developing strategies and tools for integrating multiple human sources of expertise. While using a single expert as the source for an application is usually preferred, increasingly it is the case that multiple experts are needed to develop more robust knowledge-based applications. Expert system projects, such as the ExperTAX system mentioned earlier, requiring the use of groups of experts are becoming more popular. This is causing knowledge engineers to rethink the KA methods they may use under this scenario.

As might be expected, the problems a knowledge engineer might encounter when trying to acquire the knowledge of one expert will be similar to the case where a group is involved only amplified. The four most frequently cited problems associated with using single experts are: (1) obtaining the necessary amount of an expert's time, (2) identifying possible biases of the expert, (3) identifying the limitations of a single line of reasoning, and (4) possessing only incomplete domain expertise. When using multiple experts, the problems of allocating time to a development project and of sorting out biases certainly may become exaggerated. But it may also be the case that the use of multiple experts would offer an improvement over a single expert when one considers the diversity of reasoning and domain knowledge that may be accessible. The question then becomes one of how the knowledge engineer must go about the task of accessing and structuring this knowledge.

It may be the case that one of the techniques discussed earlier, such as interviewing (structured and unstructured) and protocol analysis, may be adapted for use with multiple experts. But these techniques may not be suitable for combining multiple sources. One technique with a long history in the social sciences is that of the Delphi method. This method employs a questionnaire that is circulated among a group of experts who anonymously comment on it. Answers are aggregated and a new questionnaire is constructed and circulated based upon the results. Originally designed to generate a forecast by a group that was scattered around the world, the fact that comments are anonymous limits the likelihood that one individual may dominate the process. However, it seems to be most effective for certain group tasks such as forecasting and planning.

Brainstorming is another possible group KA technique. Experts are usually given a question or a problem scenario and asked to come up with one idea regarding it which may be either written on a piece of paper or entered into a computer file. These ideas and comments are collected and randomly distributed among the participants who are asked to comment or to develop them further. When the number of new ideas and comments being generated slows down, the knowledge engineer in charge collects the ideas together so that they may be further voted on and organized by the group. Once idea generation has slowed, the knowledge engineer may use one of several different voting algorithms to rank the alternatives or ideas to determine which one is best. Voting can again be done anonymously or publicly as a group. Brainstorming has the advantage of being anonymous and also encouraging the generation of ideas and their interaction. It can be easily implemented on a computer network and has been used with some success with the group DSS at the University of Arizona.

Many of these group KA techniques are now being adapted to the Internet. Software such as Microsoft's Netmeeting and Internet chat rooms and bulletin boards are being used to collect expertise and support KA. A more robust example might even be the usage of Lotus Notes to support enterprise-wide KA and also to manage the corporate knowledge base. This area of organization learning and KA has not been explored in great detail at this point.

III. CONTEXTUAL FACTORS INFLUENCING KNOWLEDGE ACQUISITION

It should not be overlooked that there are many contextual factors that may have a moderating impact on the quality of KA episodes. Research by Holsapple and Wagner showed that contextual aspects surrounding KA processes can be organized into three major categories: organizational factors, human factors, and physical factors. The characteristics of the problem domain itself are another aspect of the project development context that may impact the quality of KA processes.

One KA research direction tries to exploit the linkage between some KA techniques and expert system tasks by designing tools and methods to elicit knowledge for specific problem domains. The choice of a KA method is regarded as a function of the knowledge types associated with the application and certain attributes of the problem domain, such as its size, complexity, and degree of structure. The assumption behind this task-method association is that particular domains, such as medical problem solving, naturally decompose into certain knowledge substructures. If these can be identified, tools may be designed to acquire knowledge for specific domains more effectively than acquisition via general-purpose KA techniques.

The most commonly used taxonomy of problem domains is the one proposed by Clancy in 1986. He classifies problems as being analytic (interpretive), synthetic (constructive), and combinations of the two. Analytic problems often involve defining categories of objects based on their attributes using inductive or enumerative techniques. Synthetic problems start from a general idea of what the solution should be and break it into subproblems. Generally, this means that there are too many possible solutions to enumerate and test in the knowledge base. Examples of analytic problems include those that necessitate classification, diagnosis, or interpretation of sensory data. Synthetic problems include those that involve planning, scheduling, design, or configuring groups of objects given a set of constraints. Problems that combine aspects of analytic and synthetic problems include command and control, instruction, monitoring, and predicting likely consequences given a certain situation.

A. Organizational Factors and Knowledge Acquisition

Often omitted from the consideration of contextual factors is the idea that various organizational factors may be important in choosing and implementing specific KA techniques and tools. Potential organizational factors include managerial support, past expert system development experience, organizational design, and technological culture.

Management support is a vital factor in determining the success of an expert system project or any information system development project. The level of management support refers to the extent of commitment and organizational resources furnished by the organization for a particular expert system project. Resource needs can be further partitioned into four major areas of interest for researchers and developers: material, financial, human, and knowledge resources allotted for developing an expert system. The amount of resources allotted from these areas offer a measure of the degree of management support. Material resources range from the hardware to the physical facilities allocated for expert system development. Financial resources are gauged by management budgets for the acquisition and use of other development resources. Human resources refer to personnel made available for development, including knowledge engineers and an allocation of source experts' time. Knowledge resources include relevant books, data banks, and audio/visual archives of expertise.

Past experience is another factor that may have an impact on the KA process. An organization's past experience refers to all relevant KA experience resulting from past development of expert systems. In particular, this factor is characterized in terms of the number of expert systems developed or attempted, the frequency of these efforts, the type of each (e.g., simple vs. complex, diagnostic vs. predictive), the KA methods used, and the actual outcomes of each of the KA methods used. An organization that has little experience in designing and building expert systems (or other types of computer-based systems) might choose a development strategy such as prototyping, which is designed to produce results quickly and at minimal cost. Conversely, an organization with considerable experience and a high level of commitment to expert systems might choose a complex set of specialized techniques, custom-tailored to fit a particular application. For instance, two organizations may be identical on the quantity and frequency measures, but may differ qualitatively with respect to the type variable, with one having wide-ranging system development experience while the other has extensive experience developing a single kind of system within a very narrow problem domain. Such qualitative differences yield distinct contexts for KA.

Similarly, an organization's past experience with respect to the methods used in developing expert systems can play a large role in determining methods

used in future projects. Development methods refer not only to general system development approaches such as prototyping and system development life cycle (SDLC) methods, but also to specific KA tools and methods employed within the KA process. The variable outcome may be assessed in terms of the success or quality of the resultant expert system or the success of the KA technique used. In the former case, care must be taken to realize that the KA technique is not the sole influence on expert system success or quality. In the latter case, there are only a handful of comparative studies that may give some guidance on measuring the success of employing a KA technique.

The design of the organization is another important factor of the organization context. In general, an organization design governs the manner in which specific tasks are assigned to workers or groups of workers within an organization. Two variables for characterizing the organizational design factor are structure and size. For instance, structure and size can affect the nature of knowledge sources used in a KA project and the various ways in which a knowledge engineer may interact with these sources. Consider the case where desired expertise is distributed among multiple individual sources existing within an organization. It could be easier to access these individuals in a matrix structure than a hierarchy. A matrix does not involve going through multiple levels of authority, and individuals in a matrix organization could be more accustomed to operating together on a project basis. In a related vein, it may also be the case that organizations with "strong technological cultures" would be more likely to choose more complex KA techniques and develop more ambitious knowledge-based applications.

B. Immediate Context Factors

Aside from the organizational context, we should also consider how factors from what might be labeled the immediate context might affect KA processes. This concerns relevant characteristics of the immediate environs in which the KA participants interact. Such characteristics received scant mention in past KA research. Yet with respect to interviewing techniques, anthropologists and psychologists have long studied effects that immediate surroundings can have on the quality of an interview process. It is known that various physical traits may affect the course of an interview session, including the sex of participants, their apparent relative ages, the relative positioning of each, the perceived physical attractiveness of each, and the impact of the actual surroundings. There is no reason to suspect that such traits cannot influence KA processes involving humans regardless or whether an interview technique is employed.

An expert's perceptions of the physical space where the KA occurs may also impact the success of the session. For example, the expert may perceive the spatial character of the surroundings as being sterile, cramped, and uncomfortable. The noise of a tape recorder may distract his or her attention. The particular problem being addressed may require that the knowledge engineer reproduce the problem-solving environment so that it "feels right" to the expert. Such sensory aspects of the immediate context are important to consider both from the practical standpoint of performing KA and from the angle of designing empirical studies of KA.

The immediate context's temporal factor has received little explicit or direct consideration in the KA literature. This factor involves sequence and duration variables as contextual elements imposed on KA activities. For example, it has been recommended that a knowledge engineer should begin a session with an informal discussion and gradually work into using a more structured interviewing technique so that the source expert is made to feel comfortable. The implication is that the sequence of activities imposed by the immediate context on the KA process may affect the results obtained. Similarly, the duration of a session or subprocess can affect the process results. It can also be regarded as a constraint that influences the choice of a KA technique. For instance, unstructured interviewing may be more time consuming, but may yield more knowledge, while some of the highly structured techniques (e.g., repertory grids) may be much quicker and still yield satisfactory results.

C. Project Context Factors

The third major set of moderating factors that constrains KA activities revolves around what might be called the project context. This forms something of a middle ground between the broad organizational context and the immediate context of KA. A KA process occurs within the context of a project. It is but one of several efforts that occur within a project. Factors describing a project context are those that have the potential to shape or constrain the planned construction of a particular expert system. We can conceive of a project context in terms of two main groups of factors: project objectives and project constraints.

The objectives of an expert system development project can affect the KA process that occurs within

the scope of that project. At the project level, it has been recommended that specific objectives be defined prior to the KA phase of system development. These can involve objectives related to both technical performance and specific task performance of the resulting system. The technical and task performance standards embody the desired results of the project. Technical project objectives are concerned with aspects of the performance of the resulting expert system. They involve such variables as system response times and memory utilization. Time-related performance objectives may dictate both the size of the resulting system and the search strategies that are allowed. These, in turn, potentially affect the way in which a knowledge engineer elicits, structures, or represents knowledge to be used by the system. Task objectives are concerned with the substance of what an expert system does. Expert system tasks involve making recommendations, providing explanations, and offering assistance in the usage of the system.

Project constraints determine boundaries within which KA can occur. Kinds of variables that contribute to understanding the project constraint factor include those characterizing the financial, temporal, and participant resources available. While it may seem obvious to state that financial and temporal constraints are involved in an expert system project, these constraints have not often been expressly considered. The reality of such constraints certainly can affect how KA unfolds. Indeed, much of the motivation for recent interest in automating some or all of the KA process stems from the fact that expert system projects can be very resource intensive in terms of both time and finances.

Clearly, a project budget can have an impact on the choice of KA techniques used in a particular project. For instance, it constrains how much can be spent on special hardware and software to aid KA activities. In addition, the way a particular budget for a project is structured can directly impact the choice of KA strategies.

Similarly, the temporal notion of project context is realized as a schedule. It has been estimated that it takes a single knowledge engineer about 1 year to develop what is termed an average-sized expert system and that further refinements to a knowledge system can add another 1 or 2 years on average. Because expert system development projects are labor intensive and can be lengthy and complex, a schedule of the project activities is often warranted. Like budget constraints, project time may be allocated in a gross manner or by specifying more precisely how long each activity is to take. Either way, a project's schedule is a potential influence on the KA portion of expert system development. For instance, the choice of KA techniques can be based, in part, on the time allocated for KA.

IV. CHOOSING THE APPROPRIATE KNOWLEDGE ACQUISITION TOOL OR TECHNIQUE

Work on the KA problem currently follows along three major, interlocking lines. These are described as technique oriented research, empirical studies, and conceptual research. The primary emphasis of KA research to date has been on developing new KA tools and methods. A handful of experiments and case studies have focused on comparing and evaluating KA techniques. Many expert system case studies do not even describe the KA process in any detail.

A. Empirical Research on Knowledge Acquisition Techniques

There have been a few recent efforts to empirically test the usability of different KA tools and techniques. Previous researchers have recognized the need for sound empirical research to compare the effectiveness and efficiency of KA tools and methods. This initial research attempts to answer what is the best KA technique or combination of techniques and what are the best circumstances for particular techniques.

Grabowski designed a KA experiment to test the ability of three different KA methodologies to elicit different types of heuristics. The three methods tested were scenarios, simulated different tasks, and actual familiar tasks. Heuristics were divided into two categories: (1) those that all subjects identified regardless of knowledge acquisition method and (2) those that only individual subjects identified. These are further broken down as conceptual, operational, and logistical heuristics. Overall, a 30% overlap in the heuristics generated by each of the KA methods was found. Of the heuristics that did not overlap, conceptual, logistical, and operational heuristics were identified that were distinct to each method. But given that the task studied (piloting a boat in a harbor) was operational in nature, the results may not be surprising.

Adelman conducted a KA experiment which varied the domain experts, the elicitation methods, and the knowledge engineers in an attempt to see which, if any, had the greatest effect on the quality of the knowledge base. Five of the six knowledge engineers had Ph.D. degrees and one was ABD, but all had extensive training in both top-down and bottom-up elicitation techniques. The relative accuracy of each was compared to a "golden mean" rule set derived prior to the

elicitation sessions. Although a long line of psychological research has been devoted to describing interviewer effects that are analogous to the potential effects of a knowledge engineer, no significant effects were observed in this set of experiments. Interestingly, the only significant source of variation came from the domain experts themselves.

One of the most ambitious KA experiments varied the different KA techniques among different groups of experts. Each expert, in turn, was tested for cognitive style, and they discovered several specific things. Among their findings was that protocol analysis took the most time and elicited less knowledge than the other three techniques they tested (interviewing, card sorting, and goal decomposition). Not surprisingly, they also found that introverts needed longer interview sessions, but generated more knowledge than extroverts. Interestingly, the rarely used techniques of goal decomposition and card sorting proved to be as efficient as the more common interviewing technique and more efficient than the commonly used protocol analysis. One measure of technique efficiency was the time it took to code the transcripts into pseudo-rules, while the number of rules or clauses was taken as a measure of acquired knowledge.

These various experimental studies are symptomatic of a recognized need for empirical investigation of KA phenomena. The small number of such studies is, in part, indicative of the difficulty in conducting them. Other difficulties in pursuing this line of research are due to the confusing terminology, conflicting operationalizations, and the proliferation of ad hoc taxonomies. In addition, results are somewhat conflicting and no clear pattern has emerged. This may be because there are problems controlling for effects of moderator variables and in operationalizing the measurement of dependent variables.

V. CONCLUSIONS

KA is a complex set of processes that typically make up one of the most time-consuming phases of the development process for knowledge-based applications. This field combines a wide range of research from cognitive psychologists and computer scientists alike in designing new KA tools and techniques. This research has shown that there are many issues to consider when choosing a KA tool or technique, including contextual factors, characteristics of the expert and knowledge engineer, and also the problem domain being addressed.

From the results of recent empirical studies, some more specific conclusions can be made. First, though the problem domains used in the studies are generally drawn from problems in the classification or command and control type, it would appear that protocol analysis does not perform as well as other, more nontraditional techniques such as card sorting. Being data-driven tasks, the use of automated KA or inductive techniques seems more likely to perform well than manual interviewing techniques. Where induction cannot be used, KA techniques for organizing highly structured interviews, such as card sorting, seem to work better than interviewing. In either case, well-structured KA techniques seem to work best in analytic problem domains, and protocol analysis performs poorly in all of the comparative studies.

For the more difficult synthetic and combination problem domains, the evidence is not as clear. Some research indicates that problem complexity may be one determinant of the appropriate KA technique to choose. So if one were to develop a highly robust expert system for project management, then the research suggests that protocol analysis might be more efficient than interviewing. The fact that interviewing is more efficient for simple domains may imply that it is best used for initial KA sessions, when the problem complexity is not yet developed clearly.

It seems clear that no matter what the type of problem domain, developers of expert systems in all fields should consider the potential impact of some of the key contextual factors. The impact of the cognitive style of the expert and the domain complexity, along with other attributes of the domain expert, all seem to be important factors in the quality of an expert system regardless of the task involved. Further research will continue to clarify some of these issues with respect to the effect of moderator variables and problem domains in order to select the most appropriate KA techniques for the development of specific knowledge-based applications.

SEE ALSO THE FOLLOWING ARTICLES

Data, Information, and Knowledge • Industry, Artificial Intelligence in • Information Theory • Knowledge Management • Knowledge Representation • Machine Learning • Systems Science

BIBLIOGRAPHY

Adelman, L. (1989). Measurement issues in knowledge engineering. *IEEE Transactions on Systems, Man and Cybernetics,* Vol. 19, 483–488.

Boose, J. (1989). A survey of knowledge acquisition techniques and tools. *Knowledge Acquisition,* Vol. 1, No. 1, 3–38.

Burton, A. M., Schweickert, R., Taylor, N. K., Corlet, E. N., Shadbolt, N. R., and Hedgecock, A. P. (1990). Comparing knowledge elicitation techniques: A case study. *Artificial Intelligence,* Vol. 1, No. 4, 245–254.

Clancy, W. J. (1986). Heuristic classification. *Artificial Intelligence,* 27, 298–350.

Cullen, J., and Bryman, A. (1988)."The knowledge acquisition bottleneck: Time for a reassessment? *Expert Systems,* Vol. 5, No. 3, 216–224.

Dhaliwal, J. S., and Benbasat, I. (1990). A framework for the comparative evaluation of knowledge acquisition tools and techniques. *Knowledge Acquisition,* Vol. 2, No. 2, 145–166.

Forsythe, D., and Buchanon, J. (1989). Knowledge engineer as anthropologist. *IEEE Transactions on Systems, Man and Cybernetics,* 3, 19–26.

Hoffman, R. (1987). The problem of extracting the knowledge of experts from the perspective of experimental psychology. *AI Magazine,* Vol. 8, No. 2, 53–67.

Holsapple, C. and Raj, V. (1994). An exploratory study of two KA methods. *Expert Systems,* Vol. 11, No. 2, 77–87.

Holsapple, C., and Wagner, W. P. (1995). Contextual factors in knowledge acquisition. *International Journal of Expert Systems: Research and Applications,* Vol. 8, No. 4, 327–348.

Holsapple, C., and Wagner, W. P. (1996). Process factors of knowledge acquisition. *Expert Systems,* Vol. 13, No. 1, 55–62.

Holsapple, C., and Wagner, W. P. (Winter 1996). A knowledge acquisition frameworks: A case study of their development and assessment. *Journal of Computer Information Systems.*

McGraw, K. L., and Harbison-Briggs, K. (1989). *Knowledge Acquisition: Principles and Guidelines.* Englewood Cliffs, NJ: Prentice Hall.

Shpilberg, Graham, and Scatz. (July 1986). ExperTAX: An expert system for corporate tax planning. *Expert Systems.* Vol. 3, No. 2, pp. 53–65.

Wagner, W., and Holsapple, C. (February 1997). An analysis of knowledge acquisition roles and participants. *Expert Systems.*

Knowledge Management

John H. Heinrichs *Wayne State University*
Lonnie J. Hudspeth *Florida A&M University*
Jeen S. Lim *The University of Toledo*

GLOSSARY

absorptive capacity The ability of an organization to recognize the value of new, external information, assimilate it; and apply it.

abstract knowledge The form of knowledge that is conceptualized into essential features of meaning and cause-and-effect relationships that can be communicated and codified. Abstraction provides structure and meaning to phenomena. For example, an apparatus can detect the temperature of an environment when it is less than 32° F, abstract knowledge is derived from a human interpreting the data from the apparatus to conclude that the environmental conditions are "freezing." Abstraction provides structure by teasing out the phenomena's essential attributes that are necessary for describing it, while leaving out unnecessary details.

causal knowledge The know-why an event has occurred. Causal knowledge becomes the assumptions and theory in action that drives the formation of organizational strategies and practices.

chief knowledge officer (CKO) Although no commonly accepted definition of a CKO exists, the position's focus is on the leading efforts to assure the effectiveness of the organization's knowledge management strategies. The three key strategic functions served by the CKO include building a knowledge culture, building a knowledge management infrastructure, and leading efforts for the organization to extract value from its knowledge culture and knowledge management infrastructure.

concrete knowledge Knowledge can be made concrete when it is embedded in physical artifacts like products, production processes, equipment, and technology. Concrete knowledge can be perceived in terms of the particular experience or artifact in which it is embedded. This makes concrete knowledge localized to a specific context or object. For example, the know-how applicable to a particular circumstance, process, or product represents concrete knowledge that may or may not be transferable to any other circumstance, process or product. Such concrete knowledge is relevant to the local context, but not necessarily to any other. To the extent such knowledge can be codified and made more conceptual, i.e., divorced from its concrete context, it is transferable.

data Objective measurements of the attributes or characteristics of entities such as people, places, things, and events. Data are a set of discrete, objective facts about events. Data are symbols that have not been interpreted. Data may also be seen as the discrimination between states, such as on and off or high, medium, and low. Such discrimination of states can be represented as data, but may not necessarily convey information. Information is extracted from data via some type of filtering apparatus of an agent.

declarative knowledge The basis of a shared and explicit understanding of concepts, ideas, relationships, and categories that enables effective communication among people in organizations. Declarative knowledge is descriptive and is characterized by the know-what of an event or task.

enterprise knowledge portal A Web-based interface that provides users access to internal and external databases, tools, and services. Users can access

decision support systems, data mining, business intelligence tools, the Internet, the intranet, supplier extranets, customer extranets, operational and analytical databases, and business applications through the enterprise knowledge portals.

environmental scanning The acquisition and use of information about events and trends in an organization's external environment.

explicit knowledge Knowledge that is transmittable in formal, systematic language.

information Data that is placed in a meaningful and useful context for a user; is manipulated, presented, and interpreted; and has meaning. Data becomes information when it is filtered via an apparatus of an agent. Thus, information is extracted from data through the perceptual and conceptual filters of an agent. Some authors characterize information as "data that makes a difference" because it changes the way the receiver perceives something and has an impact upon subsequent judgment and action.

interpretation The process of translating events and developing concepts consistent with prior understanding of the environment.

knowledge Information that has been tested, validated, and codified. Knowledge consists of truths and beliefs, perspectives and concepts, judgements and expectations, insights, business models, methodologies, and know-how. In a sense, knowledge can be seen as information that matters to people, given their values, frames of reference, experiences, contexts, and expertise. Knowledge can build upon information that has been extracted from data. Knowledge becomes that basis or framework for perceiving new experiences and information. So, knowledge begins and ends in the minds of the "knower" as existing knowledge interprets experience and information to lead to new knowledge.

knowledge base A computer-accessible collection of knowledge about a subject in a variety of forms, such as facts, rules of inference, frames, and objects. A knowledge base can take the form of best practices, policies, and business solutions.

knowledge management The process of organizing and sharing the diverse forms of business information created within an organization. Knowledge management can include managing enterprise document libraries, discussion databases, intranet Web sites, and other types of knowledge bases. Knowledge management is the application of enterprise knowledge portals to organize, manage, and share the diverse forms of business information created by individuals and teams in an organization.

knowledge management systems Knowledge-based systems that support the creation, organization, and dissemination of business knowledge within the organization.

knowledge workers People in an organization whose primary work activities include creating, using, and distributing information and knowledge.

learning The process of acquiring knowledge about the interrelationship between the organization's actions and its environment.

organizational learning The process of improving actions through better knowledge and understanding and the process of detecting and correcting errors. It occurs through shared insights, knowledge, and mental models. It builds on past knowledge and experience. Organizational learning occurs if the organization encodes inferences from history into routines that guide its behavior and if through its processing of information the range of its potential behaviors is changed.

procedural knowledge Is how an activity is performed or happens. Procedural knowledge shared among people in organizations enables their actions to be coordinated smoothly.

tacit knowledge Is personalized, context specific, and hard to formalize and communicate.

I. INTRODUCTION TO KNOWLEDGE MANAGEMENT

The past decade has been earmarked by a significant transition into the "knowledge era." The major driving force underlying this significant transition has been the emergence and popularity of the Internet and the accompanying introduction of electronic commerce. The Internet has had an "e-everything" impact on individuals, organizations, cultures, societies, and the global economy. The dynamics of such a significant transition have driven to the forefront the awareness of the critical importance for creating and managing knowledge and then using that knowledge as a competitive resource or using it for competitive advantage.

According to scholars Davenport and Prusak (1998), the knowledge advantage is the only sustainable advantage for organizations in the information age. Due to the speed of innovation and knowledge transfer, conceivably, even knowledge-based competitive advantage is subject to erosion. In fact, an organization's capacity to improve existing competencies and learn new competencies that are of value to existing and new customers is the most defensible competitive advantage. Therefore, the increasing competitive pressures in the knowledge era, induced by rapid change,

increasing complexity, voluminous information, and increasing uncertainty, compel an organization's decision-makers to adhere to two related imperatives. The first imperative is to learn at a rate faster than current and unforeseen competitors and then to capture or incorporate that learned knowledge into the organization's knowledge bases. The second imperative is to manage the valuable knowledge gleaned from organizational learning into business models, products, services, and systems so that the organization's value propositions are more attractive to customers than the value propositions of competitors.

The processes of knowledge creation and knowledge management have become an essential component for organizations competing in today's turbulent and highly competitive environments. However, to fully utilize the knowledge management processes requires enabling technologies. These enabling technologies must support knowledge capture, organization, retrieval, distribution, and maintenance. The enabling technologies for knowledge management must also support the various forms (verbal, video, graphic, audio, and even virtual reality) in which knowledge can be represented and communicated.

The purpose of this article is to examine the concept of knowledge management from several views. First, knowledge management is described and its importance to the organization is discussed. Second, a knowledge management framework, the knowledge management life cycle, and the role of a chief knowledge officer (CKO) is examined and discussed. Third, how information technology acts as a necessary enabler for the knowledge management processes is discussed. Fourth, several successful organizations deploying knowledge management tools and technology are described. Finally, this article illustrates three prominent knowledge management technology vendors who offer unique knowledge management technology solutions.

A. What Is Knowledge Management?

In concept, knowledge creation and management has been a concern of humanity throughout our existence as civilized species. In the context of business and industry, the current focus on knowledge management stems from developments in information technology and from the realization of the strategic importance of knowledge to the success of the organization. Knowledge management is a set of organizational practices that combine the information processing capacity of information technology with the creative and innovative capacity of people to create, capture, organize, store and retrieve, diffuse, present, and maintain knowledge for the organization's benefit.

B. Why Is Knowledge Management Important?

An organization must capture and effectively manage the relevant knowledge encountered, created, and learned during the course of conducting business. Such captured knowledge must, in turn, be embodied in an organization's processes, products, services, and business models. Thus, firms must enact continuous organizational learning and knowledge management processes that enable timely and effective organizational responsiveness to customers, market opportunities and threats, and competitors.

During the more stable times of the 20th century, the primary responsibility for environmental scanning and customer information gathering belonged to senior management and the marketing function of an organization. Once interpreted, depending upon its nature and impact, this environmental and customer information was passed "over-the-wall" to other departments such as research and development, engineering, manufacturing operations, finance, and accounting. In the knowledge era's turbulent times, the responsibility for environmental scanning and customer information gathering must be distributed down the organizational levels and across the organizational functions and business units. Certainly, strategic planning and strategic decision-making need to remain the purview of senior management, but the environmental scanning and interpretation must be more widely shared. As such, organizations must possess the knowledge management processes to capture, organize, store, and share the knowledge obtained from such environmental scanning activities. As a result, knowledge management is a critically important necessity for conducting business in today's turbulent knowledge era.

In the knowledge era an organization must diligently integrate the perspectives and interpretations of its key stakeholders in evaluating the environment. These key stakeholders might include employees, customers, customers' customers, suppliers, suppliers' suppliers, industry associations, and political and civic groups. One reason for the required involvement of all the stakeholders is the limited information processing capability of humans. Such cognitive limits represent the "bounded rationality" of managers conceptualized by Nobel laureate Herbert Simon. Simon (1960) noted

that the more turbulent a manager's environment becomes, the sooner he or she is likely to confront his or her cognitive limits to process the myriad of information signals produced in the environment.

An analogy might further illustrate the idea behind bounded rationality and the need to involve all stakeholders in environmental scanning and interpretation. Imagine someone who can juggle two, three, or even four balls without dropping one. However, as the number of balls to be juggled increases to five, six, or seven balls, it becomes impossible for the juggler to handle them all. The juggler has reached the limits of his or her ability. Nevertheless, if instead of one juggler, additional jugglers joined in to share the juggling responsibility, many more balls could be handled without being dropped, assuming that their efforts were properly coordinated. By distributing the juggling responsibility, more balls can be juggled.

Futurists Alvin and Heidi Toffler have also noted cognitive limitations of managers in centralized organizations. They state that "while centralization in organizations is sometimes needed, today's lop-sided over-centralization puts too many decisional eggs in one basket. The result is 'decision overload'. Third wave organizations . . . push as many decisions as possible down from the top of the organization and out to the periphery." Such decentralization of decision making enhances the organization's ability to be flexible and agile in response to highly dynamic competitive forces and conditions.

In turbulent environments, such organizations should have an internal culture with knowledge management mechanisms and processes. These knowledge mechanisms and processes provide the flow of the right information to the right people at the right time for the right reasons. Thus, knowledge management is imperative for organizations in the information age. Environmental turbulence may impact both the external and the internal delivery systems. Decision makers must detect external environmental changes, as they may impact the organization's desired or expected states in terms of its products, markets, competitors, and legal and regulatory boundaries. For example, the environmental change resulting in the deregulation of the utilities industry has driven utility companies to drastically change their competitive postures. Their postures result in organizations that are more customer focused and possess a marketing orientation. Similarly, decision makers must also detect internal environmental changes, as they may directly impact the organization's desired or expected states internally, such as its people, resources, processes, and technology. Changes in the economy can translate into higher costs in the factors of production. Hence, environmental turbulence leads to changes that can impact an organization by influencing changes in its desired or expected states.

II. KNOWLEDGE MANAGEMENT FRAMEWORK

Before investigating knowledge management enablers such as information technology, it is important to understand the functions and requirements for knowledge management in an organization. Effective organizational decision making requires timely, critical, and relevant information about the changes in the external and internal environments. Decision makers must be able to detect emerging problems or potential opportunities that may impact the organization's competitive advantage. This must be accomplished in spite of the environmental realities of huge and enormously complex data volumes expanding at staggering rates.

The requirements for effective decision making have changed over time. This change is represented in Fig. 1. In this knowledge management framework, the knowledge management issues are shown as they have transitioned over time. In the industrial age, data-driven decision making was the key driving force for business decisions. In this stage, very limited computer processing capability and limited database structures supported the organization. As computer processing and data storage power increased dramatically, organizations shifted into the technology age. The more evolved underlying technology better enabled organizations to convert data to information for use in decision making and thereby better cope with the change. However, such a transition to the technology age required a different kind of management information system structure as well as a computer information structure to support this effort.

Now, as organizations transition into the knowledge era, the dynamics of business activity are being clocked in terms of "Internet speeds." This new environmental reality is not driven by technology in and of itself, but is driven by knowledge workers who use technology to enhance their decision making. The knowledge workers use knowledge, instead of relying solely on information or data, to guide their decision making. This orientation of knowledge-based decision making requires new tools and models to generate business insights for the organization. This basis of the new era of knowledge management utilizes knowledge workers to integrate knowledge generated from various disparate data sources into decision mak-

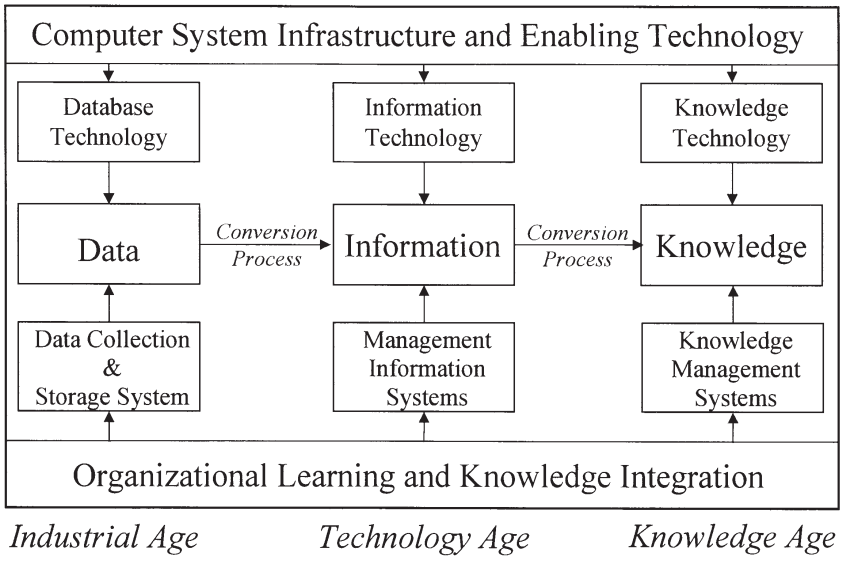


Figure 1 Knowledge management framework.

ing. Organizational learning and knowledge integration technology is at a level that enables real-time customer relationship management (CRM). For example, HelpoverIP offers a CRM product that enables real-time customer communication management for sales and customer support communication. HelpoverIP's product solution includes real-time video and voice over the Internet, live chat, whiteboard, and conferencing.

A. Knowledge Management Hierarchy

The transition from the technology era to the knowledge era implies a fundamentally different but complementary approach toward interrelating and integrating data, information, and knowledge. The technology era enabled organizations to build information systems that stored vast amounts of data that decision makers prespecified to be classified, formatted, and organized into various forms of information. Additionally, the current environment has provided the organization with a plethora of data and information available from data sources such as the Internet, organizational intranets, customer and supplier transactions systems, organizational data warehouses, and databases. As needs changed, decision makers specified new data requirements and additional formats and forms of information. From these data and information sources, decision makers, in turn, extracted knowledge with which they made decisions.

Figure 2A illustrates this "bottom-up" process of data and information specification, gathering, organization, and utilization from which knowledge is derived. This bottom-up orientation worked well during slow-changing times. Figure 2 highlights that data must first be gathered and then converted into relevant information in a timely manner for the knowledge process to work effectively. The reason is that there was enough time for organizations to obtain, store, and use such data and information to address changing business circumstances. Specified and stored data and information are still relevant for knowledge generation, but they now must be complemented by the "top-down" orientation illustrated in Fig. 2B.

However, the Internet speed of the knowledge era imposes smaller windows of opportunity and requires an additional but fundamentally different orientation toward data, information, and knowledge. First, knowledge has increasingly become a key basis for competitiveness and advantage. The form of knowledge that is most distinctively advantageous is that which is not easily codifiable, accessible, and understandable. Such a form of knowledge is less likely to be stored in databases and other information technologies, but more likely to be found in people and organizational culture. Therefore, this knowledge is likely to be in a form that is not stored in information technology. Second, for knowledge that might be derived from data and information, there may not be enough time for organizations to capture, codify, and store this data and information quickly enough for

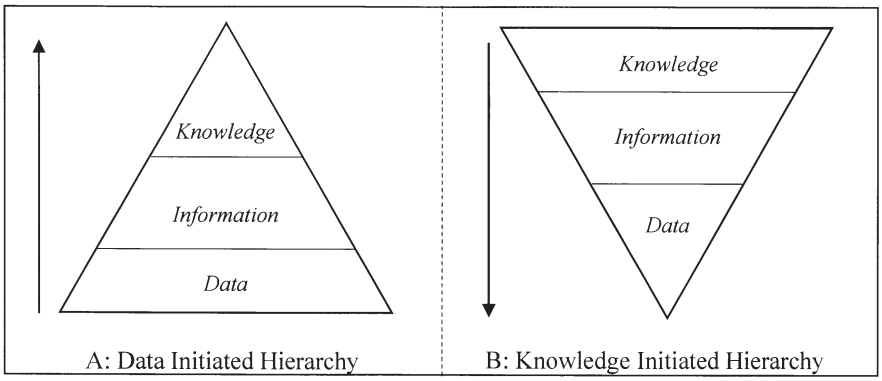


Figure 2 Data/information/knowledge hierarchy.

timely decision making. This paradox ensues because knowledge era circumstances are subject to change drastically at a moment's notice. Moreover, some of the data and information may even become obsolete before it is properly specified and stored.

Therefore, in the knowledge era it can no longer be assumed that the organization's information systems contain most of the current and relevant data and information necessary for deriving the knowledge needed for decision making. Chances are that some of the critical data, information, and knowledge required for decision making exists elsewhere in the organization or has to be created. Human beings and human interaction represent the most likely source for this knowledge. In this sense, human reasoning, based upon existing knowledge, drives the gathering of data, information, and new knowledge. Thus, in the knowledge era, data and information, in addition to that which may be stored in internal information systems, may need to be collected, and its collection will likely be knowledge driven. Hence, in this sense, the broader knowledge base that is mostly embodied in humans drives the discovery or acquisition of specific data and information. This is illustrated in Fig. 2B.

Organizations must have both formal and informal processes for sifting through the flowing river of global data and information scanning for relevant business intelligence. The technology era emphasized the prespecification of data and information needs that became embodied into the organization's formal information system and environmental scanning routines. In contrast, but also complementary to the technology era approach, the knowledge era emphasizes the dynamic creation and discovery of new knowledge through the knowledge-driven searches to specific data and information. Decision makers in the knowledge era realize that given a turbulent environment, it is impossible to prespecify all of the data and information that will be necessary to make a decision. Hence, to only emphasize the bottom-up approach illustrated in Fig. 2A would be remiss, since conditions change so rapidly. However, knowledgeable decision-makers, abreast of changes in their competitive environments, skilled at asking the right questions, and empowered with the right information technology tools, will also emphasize the top-down approach illustrated in Fig. 2B.

1. Converting Data into Information

Up to now, the terms "data," "information," and "knowledge" were used in a somewhat common-sense manner. It had not been necessary to define the terms properly, until now. In some situations, knowledge, information, and data are thought to mean the same thing. However, although these terms are related, data, information, and knowledge are distinct concepts.

Data represents a set of objective facts about events and transaction. For example, a sales transaction record may include data fields referencing the customer's name and account number, the goods purchased, the price paid, and the date purchased. Information is a message that in some way impacts its receiver's perception, judgment, or behavior. Information is data that is structured meaningfully by humans to be easily communicated and understood. Thus, information is purposefully and relevantly structured data. For example, an end-of-the-day summary of sales transactions represents data processed into information that is meaningful to managers interested in such a summary.

2. Converting Information into Knowledge

Defining knowledge is an age-old philosophical challenge that has yet to be completely resolved by epis-

temologists. However, among the scholars of knowledge management theory, several operational definitions of knowledge begin to converge around the notion that knowledge resides in the minds of people, unlike data or information. Hence, knowledge represents a combination of framed experiences, values, contextual information, and expert insight which people filter in order to act upon new experiences and information.

Thus, a major distinction can be made between the "objective" activities involved in the conversion of data into information vs. "subjective" processes of generating and utilizing this information to create knowledge. Whereas information is understood as purposefully and relevantly structured data, knowledge exists within people as a function of their values, framed experiences, contextual information, and expert insight. Thus, knowledge is information that has been enriched through human interpretation, analysis, and context. While information is relatively cheap, easy to obtain, and easy to duplicate, information's applicability, credibility, and validity cannot be taken for granted. Knowledge, however, is difficult to duplicate because its validation and applicability depends on the interpretation of a person based upon certain skills and experiences.

Noted systems theorist West Churchman made this point 27 years ago, when he stated that to conceive of knowledge as a collection of information seems to rob the concept of all of its life. Knowledge resides in the user. What matters is how the user reacts to the collection of information. Thus, knowledge can be seen as information that matters to people, given their values, frames of reference, experiences, contexts, and expertise.

Knowledge becomes something that is stored and manipulated and is also a process of enacting expertise. In this sense, knowledge has at least three different forms (see Table I): declarative, procedural, and causal. Declarative knowledge is descriptive and is the basis of a shared and explicit understanding of concepts, ideas, relationships, and categories that enables effective communication among people in organizations. Procedural knowledge refers to how an activity is performed or how it happens. Procedural knowledge shared among people in organizations enables their actions to be coordinated smoothly. Causal knowledge refers to why something occurs. As such, causal knowledge becomes the assumptions and theory in action that drives the formation of organizational strategies and practices.

Organizational knowledge, however, can exist in people in different forms ranging from "tacit/uncodified," which is difficult to control and communicate, to "explicit/codified," which is easy to control and communicate. Explicit knowledge is formal and systematic. Tacit knowledge is highly personal and is rooted in action. It is linked to an individual's commitment to a specific context or skill. Tacit knowledge consists of mental models, beliefs, and perspectives so ingrained in an individual that they are taken for granted. Indeed, it is in the process of moving from tacit knowledge to explicit knowledge that conscious and articulated understanding of something is ultimately accomplished.

Table I People-Oriented Knowledge Forms

Types of knowledge forms	
Terms	**Meaning**
Declarative knowledge	Know-what
Procedural knowledge	Know-how
Causal knowledge	Know-why

Using a store's daily sales as an example, the generation of knowledge is described. A given day's total sales might be compared to total sales from yesterday or a week ago and then compared to trends over time. Today's total sales may have shown a dramatic positive spike that prompts district sales management to inquire as to why such a positive difference exists. Although any number of factors might contribute to the spike in sales, it is only when the sales personnel reflect upon their experiences of today's events that a meaningful explanation surfaces. Perhaps a special one-day sales promotion including a tremendous discount was offered today to customers. Hence, the knowledge to interpret and understand the events had to originate from the people who knew about it.

This is not to say that organizational knowledge exists only in its people. Organizational knowledge exists in many different forms and levels (see Table II). Knowledge management scholars have specified at least six different domains where organizational knowledge can appear. Organizational knowledge appears in employees, machinery, plant or equipment, organizational systems, organizational processes, organizational cultures, and products and services.

3. Data Overload—Information and Knowledge Starvation

The difference between the amount of time required for humans to process information (minutes, hours, days, and weeks) compared to the amount of time

Table II Organizational-Oriented Knowledge Forms

Type of Knowledge Domains
Processes
Cultures
Systems
Employee
Machinery/plant/equipment
Products and services

needed for computers to generate data (nanoseconds) is enormous. Data must be gathered and then it must be understood, appreciated, assimilated, processed, formatted, and made available in order for it to become information. Information that takes on significance to the individual, given his or her values and experience, then becomes knowledge. This implies that organizations perhaps have more data available to them than they have people on hand to convert the data into information and knowledge (data indigestion). Data has to be organized and placed in a meaningful context and interpreted by individuals before it becomes information and knowledge.

Thus, although the plethora of data is a problem facing organizations, the need to convert this data into information and then into knowledge presents an even greater problem (e.g., in terms of accuracy, currency, and relevance). Without the proper information and knowledge management tools, as well as the proper use of these tools, the voluminous data cannot be converted into information and ultimately becomes useless (information and knowledge starvation). Some of this unconverted data may contain information that could lead to knowledge about critical problems and significant opportunities.

Organizations that are oriented toward continuous learning and knowledge management will likely have less unconverted data than their less learning-oriented competitors. To this extent, these learning-oriented organizations exploit their learning advantage to improve internally and to become of higher value to their customers. Therefore, these learning-oriented organizations will sustain a competitive advantage. Hence, a learning orientation is a critical competency for an organization to possess.

4. Explicating Knowledge

In addition to converting data into information and information into knowledge, organizations must convert some tacit knowledge into explicit knowledge. Making tacit or uncodified knowledge explicit enables such knowledge to be shared and reused. However, organizations are challenged to determine which tacit-level knowledge to make explicit and which knowledge to leave tacit.

The risks for not explicating tacit knowledge are the lost opportunity to share and leverage knowledge for competitive advantage. Also at stake is the threat that competitors will explicate similar knowledge for their competitive advantage. However, making the wrong tacit knowledge explicit may result in losing the essence of that knowledge and could cause performance to decline.

The conversion to and from explicit and tacit knowledge can occur in various combinations during the knowledge creation and sharing process. Nonaka and Takeuchi have theorized four different modes in the organizational knowledge creation process (see Fig. 3). Those conversion modes are externalization, socialization, combination, and internalization. Externalization involves converting existing tacit knowledge into new knowledge that is explicit. For example, through the metaphor of a "super highway" proponents conveyed to the public the importance of the Internet and World Wide Web as a vehicle of commerce called the "Information Super Highway." The tacit level common-sense understanding of a "highway" transportation system helped ordinary people to grasp the idea behind the Internet and World Wide Web. Internalization involves converting existing explicit knowledge into new knowledge that is tacit. An example of this is learning the know-how of any procedure, the instructions of which are originally in explicit form.

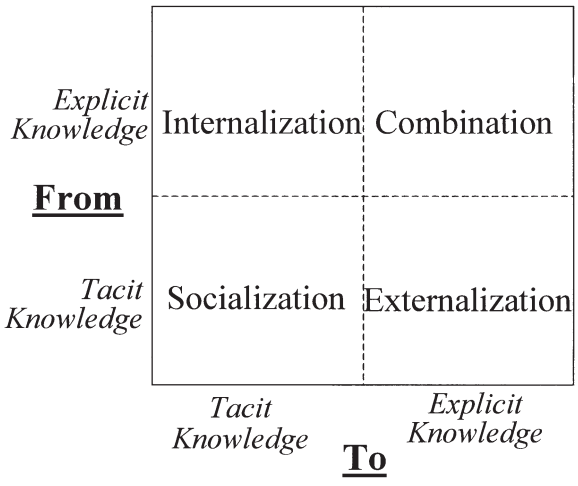


Figure 3 Organizational knowledge creation modes.

The procedure for formatting this paragraph to a certain font size and justification using Microsoft Word is an example of explicit procedure that exists in Microsoft Word Help. After following the explicit procedure a few times, the knowledge of how to justify and format the font size of a paragraph becomes internalized as tacit knowledge. Hence, explicit knowledge has been converted to tacit knowledge.

Socialization involves the conversion from existing tacit knowledge into new tacit knowledge. Such socialization is a social experience involving shared experiences and the creation of common mental models. The essence of socialization is in learning knowledge via experience. An apprenticeship is an example of such a mode of learning. Combination involves converting from existing explicit knowledge into new knowledge that is also explicit. Combination involves combining existing forms of explicit knowledge such as knowledge existing in documents, telephone conversations, or other forms to create new explicit knowledge such as would be in a new document.

B. Information Space for Knowledge Management

Theorist Max Boisot has established the information space through which organizations can understand and manage their knowledge assets. The information space comprises the three bipolar dimensions labeled codified-uncodified, concrete-abstract, and diffused-undiffused (see Fig. 4). Depending upon the how an organization evaluates a given knowledge element in terms of these three bipolar dimensions determines the extent to which information technology can enable knowledge management. The more abstract and codified the knowledge is, the more directly it can be stored, represented, and diffused via information technology. The more uncodified the knowledge is the less likely the knowledge can be stored in a computer. However, certain types of information technology can still facilitate the creation and/or transmission of some uncodifiable knowledge. For example, videoconferencing technology can enable the perception of some forms of nonverbal communication that signals tacit-level true feelings and opinions (see Fig. 5 for additional examples contrasting the three bipolar dimensions). Being able to see the facial expression of someone in real time facilitates the perception of the nonverbal communication. Additionally, certain information technology can enhance virtual collaboration and knowledge creation.

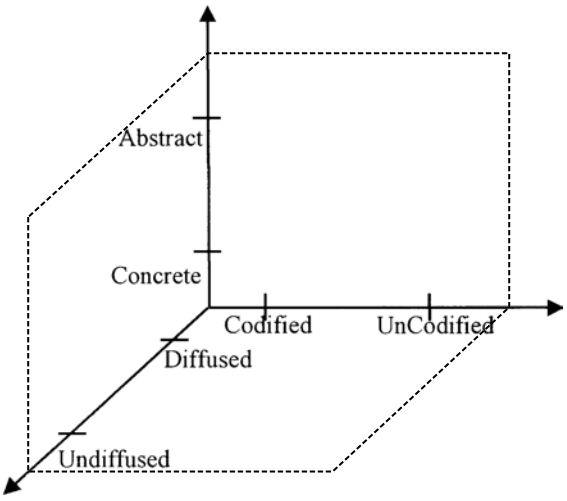


Figure 4 Information space.

1. Codified vs Uncodified

Codified knowledge can be represented in formal language such as mathematical, grammatical, digital, and symbolic codes. Codification involves the creation of perceptual and conceptual categories that facilitate the classification of various phenomena. Thus, codified organizational knowledge represents phenomena that have been classified into perceptual and conceptual categories meaningful to organizational members.

However, it is important to realize that the process of codification is not simple. Depending upon the complexity of the phenomena in question, codification can be fraught with potential problems. For example, codifying the sales performance of the store's salesperson could be viewed as a simple task of associating the number of dollars and/or the number of product units sold by the salesperson. However, sales performance could also be assessed in additional terms such as the amount of customer satisfaction generated with each sales transaction. Codifying customer satisfaction generated at the time of the sale would be much more difficult and more complicated to accomplish.

Thus, some knowledge is too rich, ambiguous, complex, and personal to be articulated or codified. Such knowledge remains uncodified and is often referred to as "tacit." Uncodified or tacit knowledge can take on two forms. Those forms are know-how and taken-for-granted beliefs. The know-how form of tacit knowledge is an embedded skill or ability acquired from birth or over time from experience. For example, the

	Examples		
Codified	•Sales in Dollars or Units	•Facial Expressions	*Uncodified*
Concrete	•Instructions for Parts Assembly	•Prototype of a New Product	*Abstract*
Diffused	•Unrestricted Access to Information	•Controlled Access to Information	*Undiffused*

Figure 5 Information space examples.

knowledge of how to ride a bicycle is probably tacit in most people who can ride a bicycle. These people can simply get on a bicycle and automatically begin riding it. However, for those who cannot ride a bicycle they need to have concrete experience on a bicycle before they can identify the knowledge needed to learn. The taken-for-granted beliefs embody what is and what should be. Taken-for-granted beliefs or notions of what is and what should be represent knowledge embedded in mental models and value systems that shape how one perceives and experiences the world. For example, one might take it for granted that everyone believes in God until he or she encounters an atheist for the first time.

2. Concrete vs Abstract

Some forms of knowledge can be made concrete or, in other words, embedded in physical artifacts such as products, production processes, equipment, and technology. This concrete knowledge that is given form must be codified in such a manner that systematic and repetitive processes based upon it can convert materials and parts into the physical artifacts. The knowledge necessary for machines to mold or assemble materials or parts into products has to be very precise and well structured into designs, specifications, and drawings. Given that such knowledge is sufficiently codifiable, it can be converted into a form such that it can be managed with information technology.

Conversely, knowledge in people's minds and knowledge conceptualized into essential features that can be communicated and codified represent abstract knowledge. Abstraction provides structure to phenomena. Abstraction provides structure by teasing out the phenomena's essential attributes that are necessary for describing it, while leaving out unnecessary details. For example, a model or prototype of a new product concept results from a process of abstracting the essential features of the new product.

Obviously, information technology cannot itself conceive or translate a concept that exists in a person's mind. However, as a tool, information technology can enable a person to represent his or her idea or concept in such a way that it can be communicated and shown to others. The actual translation of the concept into communicable form remains a human act.

Although complementary, there is a difference between codification, which gives form to phenomena, and abstraction, which gives phenomena structure. Codification assigns phenomena to categories, that is, to codes. Abstraction discriminates phenomena into distinct concepts that codification categorizes into meaningful groupings for a given purpose. For example, in a university setting, there exists a group of humans categorized as students, administrators, and professors. Abstraction enables people to differentiate the human population in the university setting into these three distinct categories. The university's information system certainly has distinct codes assigned to students, administrators, and professors listed in its databases. Thus, codification enables the formation of student lists, faculty lists, and administrator lists.

3. Diffused vs Undiffused

The diffusion of information concerns the extent to which it is available for others who want to use it. In an organizational context, the more diffused organizational information is, the more available it is to employees and other stakeholders. However, the more diffused the information is, the more it is potentially accessible to competitors. Depending upon the competitive and proprietary value of the information, the organization may have to limit and control its diffusion.

Current conceptions of knowledge management technology do not address the processing of uncodified or tacit knowledge very well. The reason for this is due to the highly personal and hard to codify nature of tacit knowledge. Because of the highly personal nature of uncodified or tacit knowledge, processes for managing it have to be people centered. The role of knowledge management technology could be as an enabler and as a facilitator of people-to-people interaction and knowledge sharing.

Knowledge management technology, however, should not be viewed as merely enhancing the technical capabilities of the existing management information systems. Successful knowledge management requires a change in attitudes, organizational culture, and processes in organizations. Technology, in and of itself, cannot improve knowledge management in organizations. People utilizing knowledge management enabling technology can improve knowledge management in organizations. Organizational knowledge management can be enabled and enhanced via information technology. Gaining the knowledge management benefits from the information technology depends upon the mindsets and behaviors of people in the organization being oriented properly. Therefore, the real challenge facing management is to change the organization's culture and the employees' attitudes regarding trust and sharing knowledge. When employees view themselves in competition with each other, they tend to resist sharing information. Often, the viewpoint is that "knowledge is power." This viewpoint and tactic may be beneficial to individuals, but it is usually destructive for the organization as a whole. In other words, while for one person knowledge may be powerful, sharing the right knowledge is more powerful to all involved.

In summary, knowledge management involves the combination of human intelligence (brainware) and technology (hardware and software). Toward this end, technology must support all components of the knowledge management life cycle.

C. Knowledge Management Life Cycle

Knowledge management technology should support the collaboration of people to create, capture, and share knowledge. It should also provide people with access to a variety of data, information, and knowledge. Knowledge management technology should support each of the phases of the knowledge management life cycle of knowledge creation, knowledge capture, knowledge organization, knowledge storage and retrieval, knowledge diffusion, and knowledge presentation and maintenance (see Fig. 6).

1. Knowledge Creation

Nonaka and Takeuchi use an excellent metaphor of "rugby" to characterize the knowledge creation process in organizations. Rugby players pass the ball nonsequentially forward, backward, and laterally as they move toward achieving their goal. This is unlike a track relay match where the baton is passed sequentially from the first runner to the second runner to the third runner and finally to the fourth runner for the completion of the race. This rugby metaphor provides an excellent description of how knowledge is created in organizations. In manipulating the rugby ball toward the goal, team members have to interact in a variety of combinations. This interactive process illustrates the knowledge creation process. As people interact, share their knowledge, and their perspectives, sometimes new knowledge is created in the process.

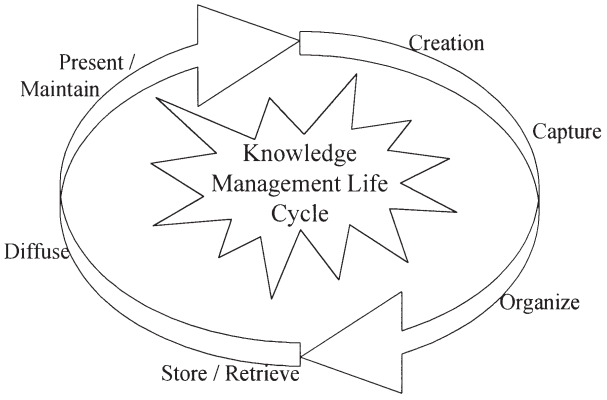


Figure 6 Knowledge management life cycle.

2. Knowledge Capture

Once valuable knowledge is created via an organization's people, the organization's challenge is to capture it for future use by others in the organization. Knowledge capture involves the deliberate intent to identify and use relevant knowledge created, discovered, or acquired. Organizations can capture knowledge through a variety of means. These capture methods include formal research and development, participation in strategic alliances, hiring new personnel or consultants, participating in various industry and other external knowledge networks, collaboration with suppliers or customers, and implementation of various process innovations. Other knowledge capture methods are achieved through the course of conducting business (learning by doing and learning by using), through the organization's deliberate continuous improvement efforts, through new product development efforts, through the diffusion of new technology within the organization, and through informal knowledge leaks.

However, capturing the right organizational knowledge at the right time is not as straightforward a process as one might think. There are many implicit

challenges and issues to be resolved. Since knowledge creation occurs via people, the first challenge is determining at what point in time does knowledge created in people become captured as organizational knowledge? Who decides this? Should knowledge capture occur informally through the organization's culture and social interactions? Should knowledge capture occur formally through specified routines, procedures, and protocols? The second challenge is once it has been determined that such knowledge is to become organizational knowledge, in what form should this organizational knowledge be represented? The third challenge is that the very nature of valued knowledge that needs to become organizational knowledge is rooted in paradox. Organizational knowledge that is to be communicated and diffused deliberately needs to be more codified/explicit than uncodified/tacit knowledge. The more explicit the knowledge is, the less likely misrepresentation or miscommunication will occur. However, paradoxically, the more valuable organizational knowledge tends to be, the more difficult it is to articulate and represent. Otherwise, competitors could easily obtain valuable organizational knowledge.

Scholars or practitioners have not resolved these challenges. Because knowledge is so rooted in people and subjective in its social context, managing knowledge requires tending to the human resources and organization culture, in addition to tending to technology and organizational structure. For example, Ernest & Young's 1997 survey of 431 U.S. and European organizations found that 54% believed that organizational culture was the biggest impediment to knowledge transfer.

Perhaps taking on the perspective of knowledge management from Nicolini and Meznar can help. Nicolini and Meznar emphasize the knowledge capture process, or the encoding of organizational learning, as a social construction process. During this social construction process, the organization deliberately stops to codify its valuable learned knowledge. Since learning occurs naturally in a continuous manner, from time to time the organization must stop and capture what it has learned.

3. Knowledge Organization

Knowledge that is formally captured by the organization has to be codified; that is, the captured knowledge should be defined, labeled, categorized, and indexed so that it can be stored and retrieved for later use. This facet of the knowledge management cycle is not as simple as it may appear. Implied in this process is a systemized set of routines for organizing an organization's knowledge. Someone in the organization has to decide what classifications and codes to associate with what knowledge objects that will be stored into the organization's knowledge base. Some forethought has to be given to how the organization will structure its knowledge and into what types of knowledge are important to capture and organize.

Scholars like Michael Zack have suggested that some organizations assign the knowledge management responsibilities to a CKO or to knowledge management centers. The CKOs or centers make the decisions that are necessary for the knowledge creation, capture, storage, and retrieval processes. The CKO or knowledge centers determine the knowledge structure and content which enables the classification, indexing, manipulation, linking, and cross-referencing of explicit knowledge. Therefore, CKOs or knowledge centers assume the responsibility for the cross-organizational process of creating, refining, organizing, and sharing the organization's explicit knowledge.

4. Knowledge Storage, Retrieval, and Diffusion

Once knowledge is stored in organizational knowledge repositories, users in the organization can search for the relevant knowledge content that they desire. Effective knowledge storage and retrieval hinges upon the establishment of entities responsible for knowledge refinement processes and knowledge repositories for the storage of refined knowledge. CKOs and knowledge centers must decide who in the organization gets access to what knowledge units. This decision determines the extent to which knowledge get diffused throughout the organization.

5. Knowledge Presentation and Maintenance

Decisions made by CKOs and knowledge centers will affect the extent to which the knowledge unit content is meaningful and applicable across multiple contexts of use. The interpretative context used by the CKOs and knowledge centers in refining and storing the knowledge must be the same context that the organization users of the knowledge repositories use. Otherwise, the knowledge will not be meaningful, useful, or relevant. The presentation of the knowledge must be in a form that is meaningful, useful, and relevant to users.

III. KNOWLEDGE MANAGEMENT TECHNOLOGY

One of knowledge management's major objectives is to connect people with people and facilitate their collaboration. Whereas data technologies are structured and typically numerically oriented, knowledge technologies deal most often with text, graphic, video, and audio formats. Knowledge management systems can be classified under collaborative work management tools. These tools help people accomplish or manage joint work activities. Examples include calendaring and scheduling tools, task and project management tools, and workflow systems.

Knowledge management requires a synthesis of several disciplines and business practices and assembles of a variety of technologies. According to the Gartner Group, these technologies interact in terms of three layers: data, process, and user interface. The data layer represents different types of data in different storage mechanisms such as relational databases, textual data, video, and audio. The process layer represents the logic that links data with the use people or systems make of it. People use is via user interface and system use is via program interface that is necessary for application integration. The user interface layer provides access for people to the information assets of the enterprise via logic incorporated in the process layer.

Technologies that enhance the ability of people to capture and manage human-added value are particularly suitable for knowledge management. For example, application sharing and video conferencing technology enable real-time communication between individuals, the shared creation of documents, group decision support, and networked virtual meetings.

Knowledge management technology supports the process of transitioning from tacit knowledge to explicit knowledge so that knowledge can be shared with others. The technology accomplishes this by providing appropriate language, formats, templates, and models to support the process of sharing explicit knowledge and explicating tacit knowledge. Technology also supports the collaborative process of new knowledge creation. Word processing, spreadsheets, e-mail, and presentation software represent traditional tools for capturing knowledge. Newer technologies such as voice recognition, biometrics, shared workspaces, and video conferencing are also useful in supporting the knowledge capture process.

Presentation technology ranges from traditional business intelligence products to automated discovery techniques. These technologies include data mining, skill mining, and text mining. Although automatic visualization of trends and patterns is still far from a mainstream knowledge management capability, technologies that support rich analytical processes will increasingly be part of the knowledge management environment.

A. Organizations Using Knowledge Management Technology

Many organizations are currently initiating knowledge management projects. Several organizations where knowledge management projects are being implemented include Arthur D. Little, British Petroleum, British Telecom Laboratories, Cisco, Context Integration, Dickstein Shapiro Morin & Oshinsky, Egon Zender International, Shiva Corporation, Storage Dimensions, The Tennessee Valley Authority (TVA), United Kingdom Post Office, and Xerox Corporation.

B. Organizations Supplying Knowledge Management Technology

Many firms provide knowledge management technology and consulting services, and the list of available software products and consulting services grows each day. The reader must be cautioned that this is a very dynamic and rapidly changing field. The reader must perform their own research to locate the information technology tools that are appropriate for their requirements. A suggested listing of various companies providing knowledge management technology is available from the KMWorld Buyers Guide. The buyers guide can act as an entry point for the search for the various information technologies that can be used in improving the knowledge management processes. Knowledge Management World also has produced a listing of what it has labeled "The 100 Companies That Matter." The reader may use it as a starting point for exploration of companies in the knowledge management field. Examples of several knowledge management initiatives sponsored by consulting organizations include Andersen's Knowledge Xchange and Ernst & Young's Center for Business Knowledge (see Table III to locate each organization's Web site address).

The knowledge products from three knowledge management software firms are further explored in this section. These firms vary in size and in the types of knowledge management software tools and products that they offer. Inxight Software provides unique knowledge management presentation tools. Microsoft

Table III Organizations Supplying Knowledge Management Software Tools

Companies of interest	Web location
KMWorld's Buyers Guide	www.kmworld.com/publications/buyersguide
KMWorld's 100 Companies	www.kmworld.com/100.cfm
Andersen Consulting	www.ac.com/services/knowledge/
Ernest & Young	www.businessinnovation.ey.com/
Inxight Software	www.inxight.com/
Microsoft	www.microsoft.com/
Verity	www.verity.com/products/

provides an array of products for content management, data analysis, information storage, and knowledge publication. Verity is a market leader in providing information retrieval tools. Each of these firms provides unique tools to enable the organization to develop unique and effective knowledge management processes. The information presented is an overview of several of the products that these firms offer. They are not meant to be complete descriptions of the capabilities of the software tools or representative of the offerings provided by the firm. They are meant to describe general capabilities of knowledge management tools and to provide a starting point in the investigation of various tools and techniques to enhance the organization's knowledge process.

1. Inxight Software

Inxight Software develops open portal software that permits Web users to navigate, preview, and analyze on-line information. Inxight's innovative knowledge extraction and information visualization tools are used in information-intensive applications of business intelligence and knowledge management.

Hyperbolic Tree Server 2.0 is a software development tool that enables organizations to integrate Hyperbolic Trees into their Internet, intranet, and extranet applications and Web sites. Hyperbolic Tree is a visual user interface component that assists knowledge workers in navigating Web sites and other information locations. By incorporating Hyperbolic Tree into portals and Web applications, organizations enable customers and knowledge workers to navigate and explore large amounts of data quickly and intuitively. It is well suited for organizing and displaying large amounts of interconnected information. Interconnected information is typically found in applications for knowledge management, business intelligence, e-commerce, and customer-relationship management. This and other products offered by Inxight are described in Table IV.

Inxight Tree Studio is a software tool that makes it easy for knowledge workers to present their Web site in a holistic and intuitive view called a "Star Tree." Navigating through the multitude of available Web pages on any Web site can be a difficult task. Most knowledge workers wander before getting to the information they want. Figure 7 illustrates the Star Tree

Table IV Products from Inxight Software

Product offering	Features/description
Hyperbolic Tree Server 2.0	Enables corporate portal, application builders, and e-publishers to integrate Hyperbolic Trees into Internet, intranet, and extranet applications and web sites
Tree Studio	Presents complex Web sites in a holistic and intuitive view called a Star Tree
Eureka	Simple user interface for exploratory data analysis
Summary Server	Extracts summaries from any electronic document
Categorizer	Automates the process of assigning electronic documents to a taxonomy of predefined subject categories
Thing Finder	Identifies the role of each word within a document & then indexes, highlights, dissects, and arranges them by key entities

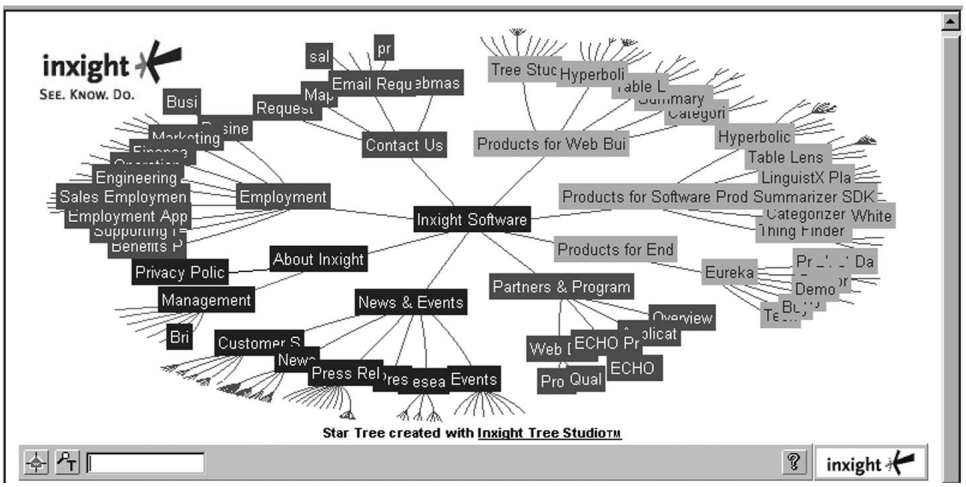


Figure 7 The Star Tree concept. Reproduced by permission of Inxight.

concept. Star Tree Web site maps help knowledge workers see at a glance all the information that is available on the site. Star Trees can give the organization's Web site an appearance that differentiates it from its competitors.

Inxight Eureka is a simple user interface for exploratory data analysis. Knowledge workers can interact with an entire set of data without scrolling through thousands of rows and columns. Eureka turns the massive amounts of tabular spreadsheet data into insightful and useful graphic presentations, enabling knowledge workers to immediately see patterns and outlines, explore correlations, and seek explanations. The "Focus+Context" feature of Eureka allows knowledge workers to view the entire set of data on a single screen. Filtering and grouping capabilities of Eureka enable knowledge workers to refine their analysis by narrowing large sets of data into relevant subsets. The spotlight feature of Eureka allows knowledge workers to mark selected data for benchmarking and spotting data anomalies.

Inxight Summary Server extracts summaries from electronic documents rapidly. Knowledge workers can view summaries of hyperlinks. Summary Server extracts key sentences, enabling knowledge workers to save time. Knowledge workers do not have to download, open, scan, and read volumes of text for relevant information. Summary Server ranks sentences by order of importance within a document and then determines the most relevant sentences to extract. Word location within a document, sentence length, and the number of thematic words all contribute to sentence identification and selection. Summaries can be presented to the knowledge workers in a variety of ways.

Inxight Categorizer is a knowledge management tool that automates the process of assigning electronic documents to a taxonomy of predefined subject categories. Documents that have been classified using codes that identify topical themes in the content enable knowledge workers to effectively browse, filter, and search for information. Traditionally, electronic documents were classified into these hierarchies by manually reading and coding each file. Classification approaches that sort documents based on rules lack scalability and flexibility and fail to provide a high degree of accuracy. Inxight Categorizer uses an advanced process known as "categorization by example" that begins by comparing a new document with a large collection of manually coded documents, which is the training set. Categorizer selects similar documents from the training set and infers the probable coding for the new document from these examples. In addition, Categorizer assigns a confidence value to each document it codes, depending on its ability to identify similar training documents.

With the incredible amount of data now available via the Internet, finding specific text, documents, or words has become more complicated and time consuming. Search engines can help, but if the knowledge workers enter the search word "car," they conceivably can get cartoons or Fords. Inxight Thing Finder identifies the role of each word within a document. It then indexes, highlights, dissects, and arranges them by key entities such as names, places, addresses, companies, dates, and others categories. Active annotation technology offers additional support materials and automatically links to other pages containing similar information. Figure 8 displays this

Figure 8 Active annotation technology.

technology. Once Thing Finder locates data, active annotation acts as a personal reference guide by finding, organizing, and presenting additional text and relevant links. By streamlining the search and organizational process, knowledge workers are able to find more information quickly and efficiently.

2. Microsoft

The Microsoft knowledge management platform has several components. Those software components include the knowledge desktop, collaboration, content management, analysis, and search and deliver. Table V details several of these products, their Web site address for further information, and a brief description of the features offered.

Microsoft Office 2000 has the capability to connect to data warehouses, collaborative messaging servers, and document systems. This creates the knowledge desktop. Microsoft Office and Microsoft Exchange Server include capabilities such as shared calendars, threaded discussions, and home pages to help groups collaborate. Microsoft Net Meeting conferencing software contains tools such as whiteboarding, video, and chat which allow workers to communicate and collaborate. The "Content Management" portion of Microsoft's knowledge management platform consists of Microsoft Exchange, Site Server, and Office. These software tools provide the ability to categorize, publish, and manage documents and content. They also support workflow around content, such as versioning, approvals, routing, and locking. The "Analysis" portion of the knowledge management platform consists of the data warehousing and business intelligence features in Office and Microsoft SQL Server. These tools enable knowledge workers to understand their markets. The "search and deliver" technologies of Site Server 3.0 search across databases, folders, and Web sites.

3. Verity

Verity is a market leader in text information retrieval. In 1997, the International Data Corporation (IDC) stated that Verity had a 28% revenue-based market share. Verity offers full text search product solutions

Table V Products from Microsoft

Product offering	Web site	Features/description
Site Server 3.0	www.microsoft.com/siteserver/site/	A powerful intranet tool for an organization to publish, find, and deliver information
Exchange 2000 Server	www.microsoft.com/exchange/	Provides e-mail, team folders, discussion groups, built-in content indexing, full text searches, workflow, real-time conferencing, text discussion, group scheduling, on-line forms, and tools for collaboration
Outlook 2000	www.microsoft.com/office/outlook/	A contact management, e-mail, and task lists software tool
SQL Server 2000	www.microsoft.com/sql/product/	A relational database management and analysis system used to integrate, consolidate, & summarize information from heterogeneous data sources and to connect to line-of-business data
Office 2000	www.microsoft.com/office/products/	A web collaboration and information sharing analysis tool consisting of Word, Excel, PowerPoint, Outlook, Publisher, FrontPage, and Photo

for intranet, e-commerce, and enterprise knowledge portals. A feature of their products is the ability for the knowledge worker to search and find information using misspelled product names or unfamiliar industry jargon. The focus of Verity's products is on solving complex organizational information retrieval problems. The Verity product family, Knowledge Suite, supports a full range of knowledge retrieval methods. The Verity product offerings are summarized in Table VI. These methods include search and retrieval, profile and disseminate, classify, organize and navigate, and publish.

Verity's Knowledge Suite includes Verity Developer. This software tool contains Verity's core search engine, its corporate search and retrieval information server, and its indexing tool. Other software products such as the Verity Profiling Tool Kit and Verity Agent Server can add custom document classification/profiling and intelligent user and group information dissemination capabilities to the Verity search engine. Verity's CD-Web Publisher enables off-line viewing and distribution of the contents of an intranet while maintaining live links for on-line use. Verity's Information Server can access a variety of information gateway products, enabling indexing and retrieval of relational databases and popular external repositories. With Verity KeyView filters and viewers, all Verity products can index and display document types ranging from simple text to Adobe Acrobat to dozens of specialized application file types.

Verity has been involved in the information retrieval market since the late 1980s. Verity's core search products include features such as advanced knowledge retrieval capabilities, custom thesaurus creation, natural language query input, and a system for building rules of evidence to classify documents. The advanced knowledge retrieval capabilities include combined metadata and full text search, advanced query navigation, and rich query language. Verity search products maintain user-definable metadata, such as author, title, or date fields. Verity provides advanced query navigation facilities to aid in further narrowing search results. These facilities include clustering, relevancy ranking, document summarization, and query by example. Topics is Verity's system for building rules of evidence to classify documents. It can define and retrieve an entire knowledge domain such as brands, product lines, products, partners, and competitors. Other vendors have produced Topics sets to classify documents according to the business characteristics of dozens of specific vertical markets.

Many solutions vendors have incorporated Verity's technology into their own products. These solution vendors include Adobe, Informix, Netscape, Nixdorf, SAP, Siemens, Sybase, and Xerox. On-line publishers also use Verity's technology as their underlying search, retrieval, and custom-subscription architecture. These on-line publishers include the *Financial Times,* Quote.com, Time/Warner, and the *Wall Street Journal.* Many organizations use Verity's products as a component in their knowledge management architecture. These organizations include Boeing, Cisco, Glaxo Wellcome, Johnson Controls, and KPMG. Examples of vendor product offerings using Verity's products include semantic networks from ERLI and multilingual search from Alis Technologies. ERLI's semantic network technology is used to build industry-specific lexicons to improve searching. Alis Technologies offers multilingual search capabilities where queries can be expressed in one of many languages and translated into the primary language of a document repository. The results lists and documents can then be returned to the users in the language of their choice.

Table VI **Products from Verity**

Product offering	Features/description
Verity Developer Kit	Capabilities enabling fault-tolerant operations and administrative tools for managing, administering, and optimizing high-performance search and retrieval operations
Agent Server/Agent Server Tool Kit	Adds custom document classification/profiling and intelligent user and group information dissemination capabilities
Information Server	Access information gateway products
Verity's CD-Web Publisher	Enables off-line viewing and distribution of the contents of an Intranet; maintains live links for on-line use
KeyView	Index and displays a broad range of document types
Topics	System for building rules of evidence to classify documents

C. Organizational Benefits from Knowledge Management Projects

Knowledge management systems are designed to provide rapid feedback to knowledge workers, to encourage behavior changes by employees, and to improve business performance (see Table VII). According to KPMG Consulting LLC, 60% of the organizations with knowledge management programs achieve faster response to key business problems or have delivered better customer service.

Shiva Corporation achieved a 22% drop in customer support calls. Staff at Shiva can locate and track common technical problems. Information is updated through the company's technical support and engineering departments. Those documents are converted to hypertext markup language (HTML) and uploaded to a server. Verity software products are used to index all available documents. Cisco achieved savings of $500 million a year from improved supply chain management, on-line technical support, software distribution via downloads, and other Internet-enabled processes. BP Amoco PLC saved $50 million in drilling costs at the Schlehallion oil field off the coast of Scotland by leveraging knowledge it had gained from developing prior oil fields.

IV. SUMMARY AND CONCLUSION

The 21st century has witnessed the expansion of the knowledge economy. The driving force of this knowledge economy has become information technology tools. By using Web-based application browsers to gain access to the Internet, intranet, and extranets, knowledge workers have virtually unlimited sources of data and information for analysis. This has enabled time and knowledge to become a new competitive lever for organizations. The knowledge workers are required to understand and access volumes of information, to develop insights using leading edge concepts and techniques, and to deal with highly uncertain and complex situations.

Knowledge management is a growing concern for organizations and is perceived as an opportunity to create a competitive advantage. A quasi-measurement that could be used to assess increases in information is Web pages. The IDC reports that over 2 billion Web pages have been created since 1995 and estimates that over 200 million Web pages are being added every month and 100 million Web pages are becoming obsolete during that same period. According to the IDC, organizations will spend over $15 billion managing their corporate knowledge. Software to automate the process of identifying and classifying electronic documents into a taxonomy of subject categories is a key requirement. The ability to extract precise information from a repository of millions of files is a very powerful tool.

Knowledge management in an organization has become a combination of social processes and information technology enablers. As such, it cannot be approached as solely a technological problem or as a method of capturing social processes. It is a combination of both elements. Organizations that are successful with knowledge management projects have realized this. Several lessons from these experiences can be shared and used by any organization. These lessons start with the "don't rely on technology too much." Technology is an enabler for change and for knowledge applications. The second lesson is to "rely on the people in the organization." People are the ultimate users of the information and are responsible for generating the insights used for competitive advantage. The third lesson is to "understand that just making data available to knowledge workers and just accumulating various pieces of information doesn't create knowledge." Using information technology tools to

Table VII Benefits from Knowledge Management Projects

Organization	Benefit	Reason
BP Amoco PLC	$50 million savings	Leveraging knowledge from developing prior oil fields
Cisco	$500 million savings	Improved supply chain management, on-line technical support, software, and other Internet-enabled processes
Shiva Corporation	22% decrease in customer support calls	Can locate and track common technical problems
Storage Dimension	Faster customer response	Due to the real-time currency of the knowledge base and rank ordering of solution documents, repetitive problems are solved correctly and at the first level every time

develop Web pages and to put that information on the Internet, intranet, or extranet does not create knowledge. Knowledge is created by individuals who have the ability to extract key data elements when required and then make sense of them.

SEE ALSO THE FOLLOWING ARTICLES

Data, Information, and Knowledge • Industry, Artificial Intelligence in • Information Theory • Knowledge Acquisition • Knowledge Representation • Machine Learning • Multimedia

BIBLIOGRAPHY

Boisot, M. H. (1998). *Knowledge assets.* New York: Oxford Univ. Press.

Churchman, C. W. (1971). *The design of inquiring systems: Basic concepts of systems and organization.* New York: Basic Books.

Coskun, S. A. (1996). *Information-driven marketing decisions: Development of strategic information systems.* Westport, CT: Quorum Books.

Davenport, T., and Prusak, L. (1998). *Working knowledge.* Boston, MA: Harvard Univ. Press.

El Sawy, O. A., and Bowles, G. (December 1997). Redesigning the customer support process for the electronic economy: Insights from storage dimension. *MIS Quarterly,* Vol. 21, No. 4, 457–483.

Garvin, D. A. (July/August 1993). Building a learning organization. *Harvard Business Review,* Vol. 71, No. 4, 78–91.

Glazer, R. (October 1991). Marketing in an information-intensive environment: Strategic implications of knowledge as an asset. *Journal of Marketing,* Vol. 55, No. 4, 1–19.

Nicolini, D., and Meznar, M. B. (July 1995). The social construction of organizational learning: Conceptual and practical issues in the field. *Human Relations,* Vol. 48, No. 7, 727–746.

Nonaka, I. (November/December 1991). The knowledge-creating company. *Harvard Business Review,* Vol. 69, No. 6, 96–104.

Nonaka, I., and Takeuchi, H. (1995). *The knowledge-creating company.* New York: Oxford Univ. Press.

O'Brien, J. A. (2000). *Introduction to information systems: Essentials for the internetworked E-business enterprise.* New York: McGraw-Hill Irwin.

Prokesch, S. E. (September/October 1997). Unleashing the power of learning: An interview with British petroleum's John Browne. *Harvard Business Review,* Vol. 75, No. 5, 146–168.

Simon, H. A. (1960). *The new science of management decision.* New York: Harper & Row; quoted in Nylen, D. W. (1990). *Marketing decision-making handbook.* Englewood Cliffs, NJ: Prentice Hall.

Starbuck, W. H. (1993). *Knowledge-intensive firms: Learning to survive in strange environments.; quoted by Bonora E. A., and Revang, O. A Framework For Analyzing The Storage and Protection of Knowledge In Organizations: Implementing Strategic Process* (Lorange *et al.,* Cambridge, MA: Blackwell Publishers.

Toffler, A., and Toffler, H. (1995). *Creating a new civilization: The politics of the third wave.* Atlanta, GA: Turner Publications.

Tuomi, I. (Winter 2000). Data is more than knowledge: Implications of the reversed knowledge management hierarchy for knowledge management and organizational memory. *Journal of Management Information Systems,* Vol. 16, No. 3, 103–117.

Zack, M. (Summer 1999). Managing codified knowledge. *Sloan Management Review,* Vol. 40, No. 4, 45–62.

Knowledge Representation

Amit Das
Nanyang Technological University, Singapore

I. INTRODUCTION

Knowledge representation deals with the encoding of knowledge in a form that can be used for computer-based problem solving. This article describes a number of approaches that have been developed to represent knowledge in software, as well as refinements to these approaches and areas of ongoing research.

A. Separating Knowledge and Inference

Most computer programs encode knowledge about the domains they work in, but the separation between domain knowledge and reasoning procedures that operate on this knowledge is clearest in the area of artificial intelligence (AI). In part, this is a consequence of a basic axiom of traditional AI that intelligent behavior arises from the combination of two conceptually distinct resources: knowledge and reasoning (also called inference). In practice, too, the separation of knowledge and inference provides an important benefit: in the knowledge-intensive domains that AI programs are applied to, the ability to add to or modify the knowledge base in an incremental manner to improve the performance of programs (without having to rebuild the entire program) is very useful. Thus in most AI programs, especially those categorized as knowledge-based or expert systems, domain knowledge resides in well-defined *knowledge* bases, separate from the inference engine.

Once the decision to represent domain knowledge as a distinct entity is made, the issue of *how* to represent such knowledge becomes salient. Early AI programs tackled the knowledge representation issue in problem-specific (i.e., ad hoc) ways, and it was not until the 1980s that attempts to organize the diverse approaches into a few coherent streams gained momentum.

B. Desirable Properties of Knowledge Representations

The central problem of knowledge representation is to encode domain knowledge into a form that supports reasoning by computers. Among the desirable attributes of any system of knowledge representation are:

- The ability to represent all types of knowledge used to solve problems in a particular domain
- The ability to derive most, if not all, conclusions that follow from a body of knowledge using the principles of reasoning sanctioned by the domain
- The efficiency of the process of generating conclusions from available knowledge
- The ability to add to and modify the knowledge already stored in a program

In addition to the above criteria, since most of the knowledge represented in AI programs is acquired from human experts, some degree of fit between the way domain experts organize their own knowledge and the way domain knowledge is structured in its computer embodiment facilitates the knowledge acquisition process. This is often the basis for choice between two knowledge representation systems when both can represent the knowledge of the domain

adequately and are equally adept at inference; the one that "feels more natural" to domain experts is usually preferred.

II. KNOWLEDGE REPRESENTATION SYSTEMS

The simplest knowledge representation system is perhaps the relational database, where knowledge is stored in the forms of relations between entities and implemented as a set of tables. Given

- the strict limitations on the type of knowledge that relational databases accommodate (entities, attributes, and relationships), and
- the very rudimentary inference capabilities provided (primarily, the ability to select records satisfying clearly-stated constraints)

most discussions of knowledge representation in AI simply overlook relational databases as a knowledge representation system.

A. Logic

Formal logic (dating back to the ancient Greeks) is perhaps the most widely discussed system of knowledge representation. Philosophers have long been fascinated by the prospect of describing (i.e., representing) the material world with logical propositions, and then manipulating these propositions to generate knowledge that would be valid in the material world. Instead of engaging the material world at each step and learning purely through trial-and-error, a logical "simulation" of the material world would be interrogated first to identify promising courses of action, and only these actions would then be instantiated in the material world.

Once the knowledge of a domain can be captured in a set of logical propositions, the universally accepted rules of formal logic can be deployed to perform sound inference. All knowledge domains modeled using logic would, therefore, use the same domain-independent inference procedures, and the only difference between different applications would lie in the content of the respective knowledge bases (sets of logical propositions).

1. Propositional Logic

The simplest form of logic that has been investigated for knowledge representation is propositional logic. Real-world facts such as "it is raining" can be represented by well-formed formulas (wff) in propositional logic; the wff denoting "it is raining" might be the single term *RAINING*. Similarly "it is sunny" may be denoted by *SUNNY,* and the relation between it raining and it being sunny may be captured by the formula $RAINING \rightarrow \neg SUNNY$ (rain implies that it is not sunny).

In propositional logic, it is always possible to determine the conditions under which a proposition is true, but this attractive feature is offset by the difficulty of expressing relations between propositions. For instance, propositional logic can represent the fact that Socrates and Plato were men (*SOCRATESMAN* and *PLATOMAN,* respectively), as well as the fact that all men are mortal *(MORTALMAN),* but it does not represent the relation among these propositions. Consequently, it does not provide a way to conclude that Socrates and Plato, being men, would necessarily have been mortal, unless they are specifically declared to be so.

2. First-Order Predicate Logic

The limitations of propositional logic as a knowledge representation system focus attention on first-order predicate logic. In predicate logic, the facts that Socrates and Plato were men are represented by the statements *man(Socrates)* and *man(Plato),* respectively. Since predicate logic permits variables and quantification, the fact that all men are mortal can be expressed as $\forall x : man(x) \rightarrow mortal(x)$ (for all x, if x is a man, then x is mortal). Using a logical procedure known as resolution, the three statements above can be combined to prove that Socrates and Plato were mortal.

The greater expressive power of predicate logic (i.e., its ability to represent knowledge that cannot be expressed in propositional logic) comes at some cost. First-order predicate logic does not have a decision procedure: though proofs of all theorems can be found, such proof procedures are not guaranteed to halt in the event of trying to find a proof for a nontheorem. However, this shortcoming of predicate logic (called semidecidability) is not of much concern in the design of working programs where halting rules can be extraneously imposed in terms of elapsed time or the computational resources consumed. Restrictions on the language can also achieve decidability at the expense of some expressive power. First-order predicate logic continues to be widely used as a knowledge representation system. Coupled with a general-purpose theorem prover as its inference engine, predicate logic

representation has been applied to a variety of tasks such as mathematics, design, and planning.

B. Production Rules

Production rules or *IF–THEN* rules, as they are better known, are most preferred for the procedural representation of knowledge. Production rules relate the *conditions* under which the rule can be applied (traditionally, written on the left or IF side of the rule) to the *action* to be performed when the rule is applied (written on the right or THEN side of the rule). Production rules have well-understood logical properties as systems of computation, but for the purpose of knowledge representation, their main benefits are

- their "modularity" (each rule represents one "unit" of knowledge, simplifying knowledge acquisition and maintenance of the knowledge base), and
- the ability of domain experts to articulate their knowledge in the form of rules.

Production rules are perhaps the most widely used form of knowledge representation. Some of the most successful programs in AI, such as the MYCIN, XCON, and PROSPECTOR expert systems, use production rules as their underlying knowledge representation system. Commercially available production rule interpreters include production system languages such as OPS5, ACT*, and CLIPS for programming, as well as a wide variety of expert system "shells" into which end-users can add their knowledge in the form of rules.

1. Forward and Backward Reasoning

Production rules can be used for reasoning either *forward* in response to given facts, or *backward* in pursuit of specified goals. In forward reasoning, the set of facts asserted to be true at any time is matched to the left (i.e., condition) sides of the all rules in the knowledge base. If only one rule is matched successfully, it is "fired," i.e., the action specified on its right side is performed. The application of the rule causes some new fact(s) to be asserted, and the process of matching currently valid facts to the conditions of available rules begins again. Since more than one rule may have its conditions satisfied in any matching cycle, a conflict-resolution procedure is required to select one rule to fire out of a set of contenders. Conflict resolution may be achieved by

- giving priority to rules that are more specific (i.e., in terms of conditions for application) over those that are more general; for instance, rules specific to egg-laying mammals take precedence over the more general rule that mammals give birth to live young when the animal in question is a platypus or echidna,
- giving priority to rules that have not been fired recently over those that have been fired recently, and
- consulting explicit control knowledge (which may also be stored in the form of rules) about which rules are more promising in which state of problem solving.

In forward reasoning, the process of matching and rule application is repeated until a configuration that matches the goal state is generated.

Production rules can also be used to reason backward. In this case, the rule R1 whose right side matches a desired goal state is identified, and an attempt is made to achieve the conditions (the left side) needed for the application of this rule. If another rule R2 is found whose right side matches the left side of R1, then the problem reduces to achieving the left side of R2. In this manner, the achievement of the final goal state is decomposed recursively into a series of subgoals until a subgoal that matches the initial state is generated.

The choice of whether to reason forward or backward is determined by a few factors:

- the relative numbers of initial and goal states—if there are more goal states, reasoning forward is viable, but if there are few goal states, backward reasoning avoids the generation of many dead ends,
- the branching factor in each direction—we prefer to reason in the direction with the lower branching factor,
- the need for explanation to the user—the direction that fits better with the user's direction of thinking is preferable, and
- the nature of the application—if the application needs to generate the consequences of a new fact, forward reasoning is useful, but backward reasoning is better suited to answering specific questions.

Forward and backward reasoning can be combined opportunistically, until the two reasoning paths—one forward from the initial state, and the other backward from the goal state—meet somewhere in between. Of course, we need to guard against the possibility of the

two search paths passing each other (without meeting) which would only increase the amount of problem-solving work.

2. Logic Programming

Logic programming, as popularized by the programming language PROLOG, may be viewed as a special case of pure backward reasoning (using the sequential order of rules for conflict resolution). The popularity of PROLOG in AI programming can be traced to:

- the fact the inference engine (a simple backward-chaining rule interpreter) is built into the language itself, eliminating the need to develop one for each application program, and
- the efficiency of implementation arising from the constraint that only one type of logical expression, Horn clauses, is allowed in the language—this facilitates the design of efficient interpreters and compilers.

C. Slot-and-Filler Structures

Slot-and-filler structures represent knowledge in the form of interconnected concepts, implemented as nodes interconnected by arcs in a network. This is consistent with the philosophical notion that the meaning of a concept comes from the way in which it is connected to other concepts. The nodes of the network may denote objects or collections of objects, while the arcs denote relations (usually binary) between the objects. Objects may also have attributes or characteristics. While this general scheme can be used to represent almost all logical knowledge, such networks (commonly called semantic nets) may also be deliberately structured to support two important logical predicates:

- *isa,* denoting class inclusion—A *isa* B denotes that the class of objects A is a subset of the broader class B, and
- *instance,* denoting class membership—C is an *instance* of D implies that the object C is a member of the class D of objects that are similar to C in some respect.

Using these relations of class inclusion and membership, it is possible to *inherit* knowledge from one part of a semantic network to another. Even though the value of an attribute for a particular object is not explicitly specified in a knowledge base, the fact that it is a member of some class of objects, which is in turn included within other broader classes allows us to infer the value of the attribute (for the particular object) on the basis of knowledge about the classes it belongs to. Of course, if an instance has the value of its attributes explicitly stated, these values override any inherited values.

1. Semantic Networks

Semantic networks started from the simple idea of interconnected concepts described above, though they have sometimes been augmented with additional features (such as partitioning) to help represent complex knowledge. In general, however, semantic nets are "free-form" in their approach to knowledge representation, and systems that impose greater structure on the representation of knowledge are usually referred to as *frame* systems.

2. Frames

Frames are representations of objects in terms of attributes (called *slots*) and the values associated with these slots. Values may be specified as logical/numeric constants, procedures for computation, range constraints, default values, and so on. A distinguishing characteristic of frame systems is that the slots of one frame may be frames themselves, and knowledge is represented in a cascading set of frames (obviously, a single frame would not be very useful). Since frame systems support inheritance, there are often conflicts between different values of the same attribute of the same object inherited along different paths. It is then necessary to resolve such conflicts and decide which inherited value dominates others—resolution strategies are usually based on some measure of network distance (the length of the path over which an attribute value is inherited).

3. "Strong" Slot-and-Filler Structures

The slot-and-filler structures described above (semantic nets and frames) place few, if any restrictions on the meaning of objects and links modeled using the formalism; hence they are sometimes referred to as "weak" (in the sense of "knowledge-poor"). Consequently, they are applicable to a wide range of problem domains as long as the programmer takes the responsibility for matching the semantics of the problem to the structure of the knowledge representation system. In contrast to

semantic nets and frames that impose few restrictions on their content, there exist a class of representations that embody specific notions about the primitive objects and relations to be modeled. Such slot-and-filler structures are said to be "strong" (i.e., "knowledge-rich") in that they constrain the semantics of the knowledge represented in addition to its structure. Two examples of such knowledge representation systems are *conceptual dependency* (CD) theory from Schank in 1975, and *scripts* from Schank and Abelson in 1977.

Conceptual dependency theory specifies a set of primitive actions to describe the interaction among entities. If the problem domain can be satisfactorily expressed using the limited set of primitives provided, inference is significantly accelerated (relative to content-free, general-purpose representational structures). Of course, only a small number of problem domains (such as natural language understanding and simple analyses of social interactions) have been shown to be amenable to modeling with conceptual dependency theory.

Scripts are structures that describe generic templates of events in a particular context (such as eating at a restaurant). Particular events, once mapped into these generic structures, can be expressed as instantiations of these scripts (with the variable elements of the script bound to specific values for the particular event). The structure inherent in a script (how different elements relate to one another) enables inference about the particular event even if some aspects of the event are not explicitly recorded. For instance, a restaurant script can tell us that we must pay for the food we buy, even if the description of a particular visit to a restaurant contains no specific mention of payment.

Many of the key ideas of frame-based systems and semantic networks have been captured in *description logics,* a family of knowledge representation languages that combine a high degree of expressiveness with nice computational properties. Description logics have been successfully applied to knowledge representation tasks in the areas of database management, conceptual modeling, and configuration problems.

D. Model-Based Reasoning

For complex domains such as electrical circuit troubleshooting or medical diagnosis, it is often necessary to represent knowledge about the structure and/or function of a specific electrical device (or human organ, in medical diagnosis) explicitly in a model. The model is then used as a basis for reasoning, using logical principles that are applicable to the class of devices under examination. At first sight, model-based reasoning, as this form of inference is usually called, seems to use a knowledge representation quite different from general-purpose formalisms such as logic or rules. However, it is possible to view the knowledge items even in a logical representation or a rule base as adding up to a "model" (usually qualitative) of the phenomenon to which the representation refers. The reframing of conventional knowledge representations as models also recasts the knowledge acquisition process as one of modeling, resolving some concerns about the philosophical status of knowledge as "extracted" from experts. Specifically, it does away with the "conduit" metaphor of knowledge transfer from the expert to the knowledge engineer, which many found to be naive and simplistic.

III. REPRESENTING UNCERTAIN KNOWLEDGE

Most real-world problem domains are characterized by uncertainty. Uncertainty may arise from features of the problem domain such as

- Genuine stochasticity—the processes underlying the phenomenon are partially random (e.g., the climatic processes responsible for the weather)
- Incomplete information—the information required for problem solving is not fully available at the outset, but becomes progressively more complete (e.g., a judicial trial where evidence is presented sequentially, piece by piece). In these cases, the absence of key pieces of information limits the ability to draw definitive conclusions. It also admits the possibility that some inferences that are valid at one stage of problem solving may be invalidated (and therefore have to be retracted) later on as new information arrives.

A related source of uncertainty arises from the overwhelming complexity of reasoning in certain domains—exhaustive enumeration of the search space is possible in theory, but combinatorial explosion renders such enumeration impossible (e.g., the set of possible board positions in the game of chess). Pruning the set of alternatives considered renders the outcomes of actions uncertain, as some possible outcomes are overlooked.

The presence of uncertainty necessitates refinements in the knowledge representation systems described above. At least two main approaches to handling can be

distinguished. One approach deals with uncertainty by making (reasonable) assumptions about information not currently available. If these assumptions are subsequently found to be invalid, the problem solver has the capability to rectify all erroneous inferences resulting from the invalid assumptions. This approach to dealing with uncertainty may be broadly characterized as *nonmonotonic reasoning*. The label "nonmonotonic" refers to the characteristic that facts once asserted to be true might sometimes need to be retracted later, a possibility that never arises in deterministic uncertainty-free logic.

The second approach to representing uncertain knowledge attempts to quantify the degree of uncertainty associated with particular conclusions. As problem solving proceeds and more evidence accumulates, the measure of support for each conclusion may grow or decline. Adopting this quantitative approach to handling uncertainty brings AI in contact with statistics, and the quantitative approach may be broadly called *statistical reasoning*. A quantitative approach to uncertainty makes it necessary to specify clear policies about

- how the uncertainty associated with individual items of knowledge is assessed and expressed, and
- how the uncertainty of the multiple premises used to arrive at a conclusion should be reflected in the uncertainty measure of the conclusion itself.

At least four distinct streams (each with many variants) can be discerned within the statistical reasoning approach to handling uncertainty. They are *certainty factors, probability and Bayesian networks,* the *Dempster–Shafer calculus,* and *fuzzy set theory.*

A. Nonmonotonic Reasoning

Nonmonotonic reasoning enables conclusions to be drawn from the absence of particular facts as well as their presence. In doing so, it opens up the possibility that some inferences enabled by the absence of related facts may need to be retracted when these facts become available subsequently. It also raises the issue of maintaining consistency in the knowledge base when new facts are added or old ones are removed. Finally, incomplete information is often unable to rule out divergent (and potentially conflicting) inferences, raising the question of which path(s) of reasoning should be pursued and which ignored.

Since first-order predicate logic is itself monotonic, a variety of alternative logics have been put forward as candidates for implementing nonmonotonic reasoning. These include modal logics, default logics, support for abductive reasoning, modified inheritance schemes, and minimalist reasoning assumptions. At the implementation level, each faces similar problems:

(a) how to restrict the nonmonotonic inferences to those of relevance to the problem (since a large number of "useless" inferences could potentially be generated),
(b) how to incrementally update the knowledge base as new facts are added and old ones retracted—having to restart at the beginning each time would achieve consistency but would be computationally inefficient, and
(c) how to prioritize the multiple paths of reasoning that are consistent with the current state of knowledge, given that many such paths will turn out to be dead ends as new facts arrive.

While issues (a) and (c) above continue to be difficult problems, some progress has been made in the area of (b). A class of programs called *truth maintenance systems* (TMS) has been developed to perform the bookkeeping tasks associated with maintaining the consistency of the knowledge base as new facts are added or old ones removed. The details of truth maintenance systems are outside the scope of this article, but, simply speaking, they keep track of the assumptions used to make each inference and are able to locate and "roll back" all inferences relying upon a particular assumption when that assumption is found to be invalid.

B. Statistical Reasoning

We use *statistical reasoning* as an all-encompassing term to denote all approaches to uncertainty that involve quantification. Some of these approaches were developed outside of traditional statistics (e.g., certainty factors), while others (e.g., Bayesian networks) were informed by relevant statistical theories. Among the desirable features of any quantitative approach to uncertainty are expressive power (to represent the types of knowledge used in the problem domain), formal correctness (freedom from conceptual anomalies), efficiency of computation, and agreement with human intuitions about uncertainty (to facilitate knowledge acquisition as well as explanation of conclusions).

1. Certainty Factors

Developers of early knowledge-based programs such as MYCIN (which performed medical diagnosis)

adopted a pragmatic approach to the representation of uncertain knowledge. MYCIN, in particular, incorporated a calculus for uncertain reasoning called *certainty factors* in which

- each rule was associated with a certainty factor—a quantitative measure of how strongly the antecedent (left side) of the rule supports the conclusion (right side) of the rule. The certainty factor for each rule in MYCIN was assessed by the experts who provided the rules.
- the certainty factor of a conclusion derived from the application of multiple rules was computed as a simple arithmetic function of the certainty factors of the underlying rules.

The certainty factors in MYCIN provided efficient inference in the face of uncertainty. The certainty factor approach also mimics fairly well the way people manipulate numeric measures of uncertainty. However, on certain occasions, the certainty factor attached to a conclusion derived from several rules in MYCIN would diverge significantly from the direct assessment of the uncertainty of the conclusion by a domain expert. Upon closer examination, this problem was traced to the fact that MYCIN's certainty factor approach treats all evidence as independent (in a statistical sense). When this is not true, and the pieces of evidence are related to one another, MYCIN tended to overstate their joint contribution. In MYCIN's problem domain, medical diagnosis, decisions are relatively insensitive to the precise values of the uncertainty estimates, so the application performed relatively well in spite of these occasional aberrations in the certainty factor calculus.

2. Probability and Bayesian Networks

Statistical theory in general, and Bayesian inference in particular, provides a theoretically coherent system for uncertain reasoning. In the Bayesian view, probability is a measure of *subjective belief* in a proposition (in contrast to the definition of probability as a long-run relative frequency in traditional statistics). The revision of belief in a conclusion with the arrival of new evidence is achieved through Bayes' rule that relates the probability of a conclusion given some evidence, to the probability of observing the evidence if the conclusion were indeed true.

Bayesian inference provides the underpinning of economic decision making. The use of Bayesian statistics as a calculus for representing uncertain knowledge enables the implementation of AI programs that embody economic rationality. Unfortunately, pure Bayesian inference tends to be computationally intractable. All pieces of evidence, singly and jointly, potentially bear upon all conclusions; this gives rise to two difficulties:

- calculating the strength of a conclusion from this dense network of statistical dependencies is computationally hard, and
- the impact of each item of evidence on each conclusion needs to be assessed during the process of knowledge acquisition, a combinatorially insurmountable task (especially in view of people's inability to assess probabilities well).

In spite of its computational complexity, the theoretical elegance of Bayesian statistics has encouraged the search for implementations that place principled constraints to enable efficient computation. One such approach to representing uncertainty is called Bayesian networks (sometimes called "causal" networks). Bayesian networks (described in Judea Pearl's 1988 book *Probabilistic Reasoning in Intelligent Systems*) place links between nodes representing propositions only if the propositions are causally related. Except for these explicitly connected propositions, all others are assumed to be statistically independent of one another. As long as cycles are avoided in the graphs that represent Bayesian networks, a number of efficient algorithms exist to propagate uncertainty from evidence to conclusions.

3. Dempster–Shafer Calculus

Dempster–Shafer theory (described in Glenn Shafer's 1976 book *A Mathematical Theory of Evidence*) is a generalized scheme for expressing uncertainty. It considers sets of propositions (instead of just single propositions) and assigns to each set an interval within which the degree of belief for the set must lie. This is especially useful in situations where each piece of evidence implicates multiple candidate conclusions, and the support for each individual conclusion is computed from the overlapping contributions of diverse pieces of evidence. Unlike classical probability theory, Dempster–Shafer theory also enables some portion of the belief to be kept "unassigned" to any of the candidate conclusions (to reflect the relative state of ignorance in the face of incomplete information). These features of the theory make it very suitable for knowledge representation in certain domains, notably legal reasoning.

4. Fuzzy Sets

The motivation for fuzzy sets as a system for representing uncertain knowledge arises from the observation that the *membership* of many classes (such as that of "tall people") is not binary (as is usually implemented in most computer logics) but rather a continuous measure (called a possibility function). Different people are members of the class of "tall people" to different degrees. This is qualitatively a different type of uncertainty than a "strength of belief." To see the difference between the two types of uncertainty, consider the statement, "John was pretty sure that Mary was seriously ill." The statistical measures of uncertainty described earlier capture John's state of belief quite well, but fuzzy logic appears to have the edge in describing the state of Mary's health. This suggests that fuzzy logic plays a role complementary to that of statistical measures in representing uncertain knowledge.

Many proponents of fuzzy logic use it as the sole mechanism for representing uncertainty in knowledge bases. Reasoning is achieved by combining distributions that achieve a trade-off between expressing uncertainty accurately and supporting efficient inference. Not only have fuzzy knowledge representations been implemented in a variety of applications (ranging from train traffic control to domestic appliances), a number of older knowledge-based programs have been reimplemented in "fuzzified" form.

IV. RESEARCH IN KNOWLEDGE REPRESENTATION

Knowledge representation was always critical to the advancement of AI systems, but the first generation of such systems adopted ad hoc, problem-specific approaches to knowledge representation, obscuring some of the issues common to all representations. It was only in the 1980s that the topic of knowledge representation as a research area in its own right gained momentum. Since then, an appreciable amount of effort in AI has been dedicated to the development of "portable" representation systems (such as logic, rules, and slot-and-filler structures) that apply across multiple problem domains, while retaining the mix of expressive power and efficiency demanded by each domain. Insights gained in knowledge representation research have also advanced the state of the art in other areas such as deductive databases and object-oriented programming.

A. The Centrality of Knowledge Representation in AI

The topic of knowledge representation is central to the field of artificial intelligence. Not only are appropriate knowledge representations critical to the design and performance of commercially valuable software programs, our choice of knowledge representation systems also surfaces our (often) implicit theories about the very nature of machine and human intelligence. For instance, the proponents of rule-based representations and those of semantic nets may hold somewhat different views of how knowledge is organized in human minds. While some AI practitioners restrict their attention to the development of software systems that perform particular tasks, the broader field of cognitive science asks questions about what constitutes knowledge and how it is organized in humans and machines. For such inquiry, knowledge representation systems provide a vehicle for expressing and testing theoretical claims and hypotheses.

B. Explicit and Implicit Representations of Knowledge

We should note that the explicit representation of knowledge using the techniques described in this article does not apply to connectionist systems (e.g., neural networks) that store knowledge in the configuration of networks (specifically, link connectivity and strengths in the case of neural nets) rather than in any explicit form. Such systems acquire knowledge through a process of "training" wherein network parameters are continually adjusted to improve the performance of the network at a particular task. To the extent that the performance of a connectionist system can be considered intelligent (in the conventional AI sense), a trained network may be said to store knowledge in its configuration information. To distinguish connectionist networks from conventional ("symbolic") AI programs, such networks are sometimes termed *subsymbolic* or even nonsymbolic to denote their radically different approach to knowledge representation.

SEE ALSO THE FOLLOWING ARTICLES

Artificial Intelligence Programming • Data, Information, and Knowledge • Hybrid Systems • Industry, Artificial Intelligence

in • Information Theory • Knowledge Acquisition • Knowledge Management • Machine Learning

BIBLIOGRAPHY

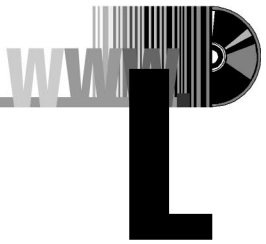
Brachman, R. J., Levesque, H. J., and Reiter, R. (Eds.) (1992). *Knowledge representation.* Cambridge, MA/Amsterdam: MIT Press/Elsevier.

Davis, R., Shrobe, H., and Szolovits, P. (1993). What is a knowledge representation? *AI Magazine,* Vol. 14, No. 1, 17–33.

Rich, E., and Knight, K. (1991). *Artificial intelligence,* 2nd ed. New York: McGraw–Hill.

Russell, S., and Norvig, P. (1995). *Artificial intelligence: A modern approach.* Englewood Cliffs, NJ: Prentice Hall.

Winston, P. H. (1992). *Artificial intelligence,* 3rd ed. Reading, MA: Addison–Wesley.



Law Firms

Brian McNamara
California State University, Bakersfield

William Go
Go & Associates

GLOSSARY

attorney Person admitted to practice law in their respective state and authorized to perform both civil and criminal legal functions for clients, including drafting of legal documents; giving of legal advice; and representing such before courts, administrative agencies, boards, etc.
courtroom The portion of a courthouse in which the actual proceedings take place.
deposition A pretrial discovery device by which one party (through his or her attorney) asks oral questions of the other party or a witness for the other party.
evidence Any species of proof, or probative matter, legally presented at the trial of an issue.
interrogatories A pretrial discovery device consisting of written questions about the case submitted by one party to the other party or witness.
jury A certain number of men and women selected to inquire of certain matters of fact and to declare the truth upon evidence to be laid before them.
trial A judicial examination and determination of issues between parties to action.

I. INTRODUCTION

Vince Lombardi stated that "Winning is not everything—it is the only thing." Many law firms use this motto. Their ability to win ultimately determines how successful they are in financial terms. Today, information systems play a strategic and important part in the goal of law firms to win. Not all law firms deal with litigation. Many successful law firms play a more transactional role in the legal profession (preparing the needed paperwork to document business dealings). Information systems can once again play an important role.

The use of information systems by law firms, in general, can be subdivided into three areas:

1. Business operations
2. Pretrial
3. Trial

The business operations support and underscore all the operations of the law firm, whether the firm is involved in litigation or not (this includes mergers, securities law, etc.). *Pretrial* is the stage used to determine whether the law firm has a good case or not. An honest assessment at this stage can make or break many a law firm. The *trial* stage is the final area where the issues in dispute are decided.

Managing the delivery of legal services is where the rubber meets the road. Information technology will get a law firm maximum mileage and performance. A law firm succeeds or fails on the quality of its legal services and the efficiency in its delivery. This success includes responding to client concerns as well as needs, understanding the client's business, paying careful attention to fee arrangements, and effectively managing the client. But "quality" only begins with substantive legal and technical competence in today's environment.

This article takes a close look at each of the three areas of information systems used by law firms, particularly what is new in information systems and the

successful law firms. It also takes a "looking glass" look at the future and makes predictions about future use of information systems in the legal profession.

The contents of this article will provide a solid foundation in information systems on which readers can gain insights into the interplay between a successful law firm and information systems. The essence of this article is that information systems underscore all operations of a law firm. This is the case because information is the single most powerful resource in every business function of a successful law firm. In addition, this article will provide the correct balance between technical information and real-world applications. The text will provide the reader with enough technical background to see the benefits of information systems immediately. The text will not be so technically based as to cloud the important issues in a successful law firm today. Real-world discussion will be presented throughout the article to provide the law firm context in which the information system will operate. In short, the article separates the wheat from the chaff.

After reading this article the reader will understand how successful law firms get information to work for them. Moreover, enough information will be provided to demonstrate how one can work productively in a successful law firm. Furthermore, the applications will be presented in such a manner to demonstrate how a law firm can use information systems to support its work in the best possible way.

Business practices are always changing. The information systems field changes daily. To remain current the research for this article actively sought all new and exciting information systems and their applications to be presented. The constant used throughout the research was that it focused on emerging applications for the successful law firm.

A. Information Systems

Today, information systems are thought of as computer-based information systems. With the hype surrounding computers, it can blind people to the simple use of their five senses to gain information on which to make decisions and improve ones' law firm. For example, a client says that when he or she called the office the person on the other end was rude. A computer information system is not needed to tell the law firm that this situation has to be dealt with quickly and effectively.

This article will deal exclusively with computer- and technical-based information systems. So as a disclaimer it is not an oversight by the authors but one of the best information systems on the market is in your own back yard so to speak—your five senses. First, the article will examine the role of information systems in a successful law firm. Second, the article will review the unique aspects of the law firm as a business. This will include an introduction to the main focus of the chapter: business operation, pretrial, and trial. Third, the article will deal with each of three areas in depth with regards to information systems. The article will finish with a look at the future and information systems in the legal profession.

B. The Law Firm

The law firm has many unique aspects that must be addressed to give the reader a foundation to what drives a law firm beyond the traditional modes of operation for a business. This foundation provides the reader with greater insights as to why certain information systems are required while others may not get the same emphasis which would not be the case in a traditional business operation. For example, a tickler file in a marketing system alerts a salesperson as to whom one should call or make contact. A tickler system in a successful law firm alerts an attorney as to the statute of limitations regarding the filing of a case. If the salesperson misses the call, it can be done the next day. However, if the statute of limitations is passed the lawsuit is moot and the law firm could be subject to a malpractice suit and could potentially cost the client their day in court.

1. The Law Firm as a Business

A law firm is a business. A successful law firm is a business that is profitable. A law firm is a service business, essentially it solves problems. Clients come to a law firm to have their problems solved. Theoretically, if a person goes to work and pays their bills on time and does not get run into the back-off, they will never need the assistance of a law firm in their lifetime. Therefore, an information system must be geared to serve the clients. Like any business, satisfied clients refer other clients. It is imperative that any information system that supports the operation of a successful law firm must have this goal in mind. Other intangible goals would only detract from this aim.

2. Being an Attorney

Attorneys are required to take legal ethics courses in law school. Once graduated and after passing the Bar,

most states require that attorneys as part of their continuing education take further courses in legal ethics. These ethical rules are codified in most states. Attorneys can, in extreme cases, be disbarred for violation of these rules. To reduce the likelihood of being disbarred, it is necessary for an attorney to have an information system that would reduce all likelihood of ethical violations arising.

3. Pretrial

Most disputes and problems that attorneys deal with are resolved without a trial. If initial contacts between attorneys do not settle the dispute, then both sides begin pretrial activities. Basically, pretrial activities such as serving the other side with interrogatories and conducting depositions are all done for the sole purpose of negating the need for a trial. During these activities both sides gain information that allows them to assess the strengths and weaknesses of their case. Following full discovery, both sides can meet and resolve an outcome or they can use an arbitrator. Today, many disputes are encouraged to use mediation as an avenue to resolve the dispute. Any of these options will hopefully resolve the situation.

4. Trial

If the dispute is not resolved at this stage, then a trial is needed to settle the dispute. A trial, at best, can be described as rolling the dice. This is where an information system can be used to "stack" the deck in favor of the client. Many new methods are being used in trial to "put one's best case forward" so that the jury will find for your client. Here, an information system can give a successful law firm a competitive edge. Remember being a "winner" as a law firm makes a law firm successful.

II. BUSINESS OPERATIONS

A. Standard Business Practices

A successful law firm in today's business environment has to contend with an unceasing flow of information. However, for many years attorneys have had a bad reputation as business people. Today, attorneys can no longer be undisciplined about their business practices (as "fat profit margins" of the past no longer exist). This has come about due to increased economic pressures, court-mandated time limitations under "fast track" case processing rules, and malpractice concerns. In a survey of claims handled by one legal malpractice insurer, the four most common causes of claims were:

- Failure to calendar a matter properly
- Failure to be clear about the attorney's intention to take or withdraw from a case
- Failure to adequately monitor client's payments, resulting in a suit for fees that prompts a client's cross-complaint for malpractice
- Inadequate knowledge of the law or investigation of the facts

Moreover, an attorney as a business has become more competitive. This has forced the successful law firm to consider their business practices. Cole further stated that a good case management system must perform the following:

- Organize client information for effective filing, retrieval, and use.
- Facilitate workflow by providing sufficient notice to accomplish specific tasks in a timely manner.
- Generate information for billing, budgeting, and other financial purposes.
- Provide data for decision making about growth and marketing.

Even in a small law office, information must be collected, classified, sorted, recorded, filed, retrieved, and communicated. Much of the information collected relates to a law firm's routine business functions. To underscore these operations a legal firm needs good network design and implementation, document management, custom application integration, Internet/intranet development, education, and technical support. These elements of an information system are beyond the scope of this chapter; however, they will be considered to be in place and fully operational. The following areas discussed represent features of systems integration and office automation solutions to the legal community.

B. Word Processing

Suffice to say that all the information in other chapters on this subject can be incorporated here by reference. In term of the successful law firm a word processor dramatically reduces the time one spends preparing routine documents. This is done through the edit function. A word processor allows an attorney to review complex documents. The word processor creates professional

looking documents which help create a good relationship with the client. All modern word processors allow one to add graphs, charts, and tables with ease, and they can be used to quickly find and mark legal citations and generate tables of authorities. Developments in word processors related specifically to the legal profession let attorneys organize and streamline their practice by client matter. The big breakthrough has come with the use of voice recognition software. This software allows an attorney or an office staff member to speak to one's computer. The software using speech recognition translates one's voice into text. The main development in this area of information systems has been the software's ability to deal with dialects and accents and individual pronunciations.

The Internet has had a big influence on the features now available with word processors. Software has been developed which allows an attorney or office staff member to organize documents and publish them on the Web. This software creates hypertext markup language (HTML) code automatically and does not require the user to learn a "programming" language. It also can be used for the creation and updates of a successful law firm's Web site.

C. Presentation Software

Again, by incorporation by reference, information on this topic in other chapters can be included here. Presentation software allows a law firm to create captivating, high-quality slide shows and drawings that leave a lasting impression with the viewer. More use of this software will be discussed in Section IV.

D. Time Managers

Time management software allows attorneys to manage time and tasks effectively. With overcrowded courts and long delays expected in courtrooms this software becomes a vital tool to organize what time attorneys have to maximize client contact, work done on cases, etc. The attorney can organize, store, and access communication and scheduling information with ease. The software is now advanced to the point where an attorney can customize personal journals. Also, it allows the attorney to create user-defined timetables, real-time calendars, "to-do" reminders, and alarm alerts. This can all be done with e-mail notification. From the law firms' perspective, these time managers can be used to support the billing function by keeping track and to bill activities. Again, the attorney must focus on the service side of the business and customer satisfaction. The use of this type of software is a vital player in both these goals.

E. Office and Internet E-Mail

This type of information system has been a great step forward for the successful law firm. Within an office each attorney and staff member is exposed to real-time internal messaging. E-mail can be used to contact clients and other attorneys. This method of communication has replaced, to some degree, the use of the telephone. It theoretically reduces "social talk," while at the same time gets to the "heart of the matter" effectively. Recent problems with internal e-mail, however, tend to discount the elimination of "social talk with verbal chat" being replaced with electronic chat. This communication method also creates a paper trail and allows one to check on e-mail sent and received. The use of an address book to send e-mail to multiple parties at the same time is a very time-saving device. The communication afforded by such a feature would be hard to quantify.

F. Databases and Spreadsheets

The benefits detailed in other articles in this Encyclopedia regarding the use of databases and spreadsheets can be applied to the successful law firm.

G. Case Management

Case management software allows the successful law firm to be more efficient and better organized. This type of software provides a convenient method of tracking all client and case information. The information input could, in-turn, be used to automatically prepare documents and, on an as-needed basis, management and status reports. Its features included an automatic document tracking and locating system and an automatic check for conflicts of interest and statute of limitations deadlines.

A case management system can organize all information on each client file, generating appointments and tasks. Delegation of these tasks could also be accomplished. This information could be automatically cross-referenced to calendar activities and autoscheduling. With the proper equipment connected to the information system, it would be possible to contact a client with one click of a mouse. From this activity

phone histories could be logged and time entries could be integrated with the billing system. The system could, in turn, assemble all the documents needed for a client conference or interview. If the attorney needed to attend court on a case, then the appropriate documents could be assembled for that activity. From a strictly management perspective, this information system could print warnings on files that were inactive or whose time exceeds targets.

Case management would be a great tool to improve client contact and satisfaction. In today's competitive business world a client does not like being put on hold. This comes about when the attorney has to dig through a file or files to answer the client's question. Case management software will provide the information attorneys need at their fingertips. It will provide all of the information on a certain client—notes, documents, time spent, and more. With this information at hand, a successful law firm can impress its clients with quick and accurate answers.

H. Billing System

In 1997 the following was stated in the *California Lawyer:* "For years, time and billing chores were dreaded in most law firms. Lawyers were forced to fill out forms long after they performed work. Then the forms were entered onto the mainframe before being mailed to the client. The system often led to lost billable hours and mistakes in final bills. Computer software revolutionized the process, allowing lawyers to enter their billing as they performed tasks and cut out the middleman who had to enter the data into the system. In addition, some applications automatically prepared the formatted bills ready to be mailed to the client."

The most important function from a business perspective is keeping track of billable and non-billable time, recording client transactions, preparing statements, managing accounts receivable, and the like. If a law firm is still in the position of relying on manual systems or information systems that are outdated, it is almost impossible to be competitive.

In choosing a billing system one must be careful to avoid a system that is so complicated that it discourages its own use. By analogy, the early word processing programs were shunned by secretaries as "too complicated." Simplicity encourages use, with an intuitive program being the "ideal." If the program does not make sense to the user, it probably does not make sense to attempt to use the program.

A lawyer hates nothing more than having to fill out billing sheets. Busy attorneys who spend most days in court and meeting clients often forget to generate time entries. The features in a billing system will prompt the busy attorney to make a time entry after every billable action, with the result being no loss of any billable hours. This type system reduces the time and saves significant costs associated with preparing and reviewing time sheets. Most lawyers do not immediately recognize the problem with failing to fill out completely or timely billable time. When these tasks are not completed, cash flow becomes a problem and ultimately profits suffer.

A billing system can hold in one place all the information concerning a client, the latest activity on a client, and what has to be done next for that client. It would be a simple matter of calling up a client's file and entering the work done and billable hours. From this entry alone the information system could generate the status of a client's bill before the amount was in excess of another attorney's retainer! The information systems would allow the successful law firm to print billing statements. It could also be used to produce an attorney profitability report for each attorney in the law firm. The report could be broken down by billed fees, overhead, and profit. Also, another version of the same theme would be an attorney productivity report, i.e., amount of hours billed per month and hours worked monthly, yearly, etc. It could also be used to breakdown the law firm's area of practice and give the law firm insight into what areas of the law are more profitable than others. Moreover, from a general partner's perspective, the information system could generate a work-in-progress report to provide a "bird's eye view" of what the law firm is doing at any point in time.

I. Calendar

A calendaring information system allows the office staff and attorney to view their schedules and appointments by day, week, and year. Important features for the less organized attorney are the to-do lists, deadlines, phone calls to be made, and notes. One of the most common client complaints lodged against attorneys is that they lose touch with their clients even though their case is ongoing. Promised follow-up calls and activities are not accomplished. Promised letters are late, and important details are forgotten. The calendaring system can help solve this problem with a wide variety of tools. Callback reminders alert the attorney to make a promised call. Overdue items are clearly marked in the attorney's calendar. Notes made on a client can instantly be incorporated into the

attorney's reminder. These activities will certainly go a long way to satisfying clients' expectations. In a large law firm such a system can be used to coordinate multiple people or groups required to attend an activity. This information could be used to make sure vacations and statutory holidays were protected. Attorneys like to free up holidays, and there is nothing worse from a customer-service perspective than to have to reschedule appointments. The motto here is to schedule once. Also, from a law firm perspective it can mark employees' and clients' birthdays. These events can be used to affect morale in the law firm and to be a good customer relation's activity with the client.

J. Conclusion

Areas such as accounting and human resource management, which are important areas to law firms as these topics, will be covered in depth in other chapters in this book. As stated previously, one of the top goals of a successful law firm is its concern about the client's needs. Also, the level of service provided must be a high priority as well. It is tantamount for a successful law firm to continually reassess and insure the health and efficiency of its practice. Given the areas discussed previouly and their uses in a successful law firm, an investment in information systems will make a world of difference in responding to client's needs and in the level of service provided the client. Moreover, an efficient and pinpointed information system will provide numerous benefits that will allow a successful law firm to bill more and reduce costs. The math speaks for itself, the result is increased profitability. The proper purchase and use of an information system will often pay for itself within weeks of acquisition.

The direct result of using an information system efficiently and effectively is that the attorneys at the firm will spend more time practicing law. In turn, they will spend less time administering the running of the law firm. Another and clear benefit of using an information system is that the productivity of the staff will be improved dramatically.

Employing the use of an information system will reduce the risk and cost associated with errors and omissions. As stated previously, an information system used properly will ensure that deadlines are met, clients are called, phone messages are returned, and a record is kept of all activities on a file. Some insurance companies reduce their malpractice insurance if a law firm can demonstrate an efficient and effective information system.

"Utilizing the business systems" will allow attorneys to stay on top of their practice and hopefully as a by-product give them peace of mind. Utilizing the system is more than ordering staff to use it. With any new procedure requiring training and lead time, "utilization" is not instant, so one must factor in the cost of disruption with present systems, the cost of the program, and the cost of training.

The bottom line to any business is the fact that it must stay profitable to remain in business. In the law firm environment there is always pressure to increase billable hours. The information provided above demonstrates that this goal can be achieved.

With greater profits the successful law will expand. The demands on one's time will increase. However, with the correct information systems in place, the infrastructure is already available to accommodate this growth. As such, even with growth the information system will become a critical ally in battling the increased demands. The information system will allow the firm to continue to work efficiently and effectively. In short, utilizing an information system is a wise investment.

Training is an often-overlooked part of the making an information system effective. Money spent on training is often considered "soft money" which could easily be put to use elsewhere. This is not so. The staff who make up a successful law firm are the number one competitive advantage. They deserved to be trained because they make the successful law firm work. A firm can "sink or swim" by its staff. The staff are the ones who use the business function more than the attorneys. They keep track of attorneys and clients. This allows the attorneys to keep track of the law. Therefore, the better the information system provided for them, the better the whole practice will be. To empower staff, make sure they get the proper training on the information systems used by the law firm. Sometimes training is not best conducted "in house"; use outside firms specializing in afternoon seminars. Knowing how to use a program does not guarantee a competent trainer. Remember a satisfied staff produces a higher quality of work. This fact alone benefits a successful law firm and ultimately affects the bottom line. With an efficiently running and effective information system, the partners of a successful law firm can sleep at night.

III. PRETRIAL

At this juncture it must be stated that very few disputes go to trial. The law itself favors settlements. Parties have every right to go to trial if their dispute cannot be settled. Judges cannot require parties to settle

a dispute. However, the system is "set up" to encourage settlements. A settlement is an agreement between the parties involved in the dispute to terminate the lawsuit.

A law firm can gain the competitive edge in pretrial to influence a settlement without involving the cost of a trial. Jack Slobodin urges attorneys to consider the following:

1. **Reasons to settle**
   - Save time and expense of pursuing or defending litigation
   - Avoid risks of unexpected high or low verdicts or judgments
   - Finality (no posttrial motions or appeals)
   - Confidentiality (can be a term of the settlement agreement)
   - Spare client emotional stress and interruption of business and personal life
   - Certainty (assure an outcome via settlement)
2. **Factors influencing amount of settlement**
   - Respective evaluations of liability
   - Respective evaluations of damages
   - Extent to which each side is prepared for and willing to go to trial
   - Degree of overlap or difference in estimates of verdict range
   - Credibility and attractiveness of parties and their positions
   - Skill, experience, reputation, and personality of counsel
   - Status of litigation (suit filed, trial approaching)
   - Ability of each side to see benefit in proposed settlement

Suffice to say, the whole goal of pretrial is to gain information about the dispute; evaluate the pros and cons of the dispute; and, if not resolved, then decide whether to go to trial. The following section reviews how information systems can be utilized in this process.

A. Legal Research

With each new dispute the attorney has to assess the facts of the case. The attorney finds the law that best supports his or her client's position. At that point the attorney must consider areas of the law that contradict or at least challenge that position. Moreover, an attorney must do research to determine if the current law is still "good law."

The traditional way of attacking these problems was going to the library and doing research through books. Some attorneys still take this approach. In essence, they are still using an information system, just one that is not computerized. There are benefits to this approach. A person doing research in the library can have a question that needs to be answered. A similar question using a computer search requires that the search engine be geared to accommodate that query. Certain attorneys state that they do better by themselves in the library than by using the benefits of a search engine to find answers to their research questions.

The Internet has spurred the development of many companies offering research to law firms. They provide instant access to court forms, decision, statutes, legislation, cases with histories, and codes annotated with full-text citations. These systems can be loaded on the desktop machine and accessed at any time. The excitement of this research is that it accommodates the attorney on the road. An attorney can be in a motel or at home or anywhere, and as long as they can log on to the Web they can do research.

Most of these companies promote the ease of use, faster access, multidatabase searches, etc. as part of their sales pitch. Moreover, the cost of this method of research can be as low as $30 a month.

Accordingly, a law firm's efforts to determine the strength of its case can be utilized at such a low cost and reap tremendous information. The attorney can keep current virtually on a daily basis so as not to get outmaneuvered by the other side. They keep current with the law and have any forms they need with the click of a mouse. This development in information systems and the use of the Web has now become part of the fundamental arsenal that a successful law firm must have.

1. Interrogatories

Interrogatories are sent from one law firm to the opposition's law firm. These interrogatories contain questions geared to gain information about the dispute. The law firm gets a certain time to respond. If they do not respond within a certain time frame then the other law firm can go to court and compel the other side to respond. Both sides generally send interrogatories to each other. Once they are completed and returned, both sides evaluate their case.

Boilerplate interrogatories are reviewed and appropriate questions are selected. A word processor is used predominately in this area to accommodate this function. The word processors' function as an information

system in this area has come under much attack. Some detractors claim that attorneys get lazy and, utilizing the benefits of word processing, send out interrogatories more for the sake of it rather than trying to tailor questions to extract the facts they really need clarified. Also, large law firms have been known to "drown" smaller law firms with interrogatories to stretch to the limits the resources of the smaller operation (these are often known as "punishment interrogatories"). Both tactics are used to gain a competitive edge and to see who can stay in the fight longest. The ethical rules in this area are scarce to none.

2. Depositions

A deposition involves a meeting between the sides accompanied by their respective attorneys. The opposing attorney can ask the parties questions. A stenographer records all the questions and answers that take place during the deposition. Depositions can be on videotape and sometimes can be used at trial for impeachment purposes.

3. Videoconferencing

Videoconferencing is a simple method to cut down the costs of traveling to various sites, including international travel, to take depositions. Moreover, it can be used for client meetings. The technology is in place for such events. With the advent of affordable integrated services digital network (ISDN) lines, videoconferencing now delivers television-quality picture and audio. From an information systems standpoint, attorneys argue that they gain a lot of information by "reading" the person being deposed through their body language, reaction to the questions, and level of movement during the proceedings. The question then becomes can the same be achieved by using videoconferencing. If experienced attorneys embrace this use of technology as a cost-saving device, then it will filter to all firms given that the cost for the set up is reducing every day.

4. Using the Web to Search Public Information

The Internet as an information resource has become an invaluable tool. Most public agencies either have or are in the process of putting public information on the Web. It is easy for an attorney or members of their staff to log on to the Internet and seek information. Below is a list of some areas that can be found.

- Public records
- Media searches
- Criminal history
- Litigation history
- Public record research and retrieval
- Property search
- Ownership form
- Background information
- Liens and judgments
- Business associations
- Financial condition

The information found can be used to strengthen a client's position with regards to settlement. Remember information is power and the Internet is currently an unfathomable medium for this type of information gathering. Knowledge-based systems are used to integrate on-line public record. These systems search and produce intelligent document assembly. More discussion of knowledge-based systems can be found in other chapters. Suffice to say, this is a growing area in the successful law firm. Once again, the competitive edge can be gained by due diligent search on the Web.

5. Electronic Evidence and Discovery

More and more businesses are digitizing their business activities. Most commercial information is now digitally coded in some manner. Thus, electronic data has become a crucial source of discoverable evidence. Parties to litigation are now targeting computer systems. Accordingly, electronic discovery has become a crucial element of the litigation jungle.

Once coded, information can then be sorted and transmitted with the click of a mouse. In response, attorneys must be prepared to accept evidence in this manner. Also, they must be prepared to council their clients on how that information should be stored and managed in the first place with the knowledge that the information maybe subpoenaed at some point in the future.

An article in *Computer Security* stated that it was inherent in computer information systems by their very nature to open Pandora's box with litigation liability. For example, the information that could be found in e-mail backup systems that store a company's e-mail; how does a company upon request produce the discovery requested form the vast libraries of information which is now no longer available; and what could be done to limit the need to accumulate uncontrollable amounts of data?

With these statements in mind, it is important for successful law firms to retain expertise in technology, law,

and litigation strategy when they are in the discovery stage of a dispute. This will ensure that the successful law firm maintains control of the discovery process from a defensive stance and attains its discovery goals from a proactive stance. From a malpractice perspective, by retaining the correct skills the firm can avoid expensive mistakes and reduce the cost of electronic discovery.

Electronic Evidence Discovery, Inc. presented the following as to what is involved in electronic discovery.

- Identifying, locating, retrieving, and reviewing potentially relevant data in both client and opposing party systems
- Identifying the most cost-effective means for responding to electronic discovery requirements with minimum disruption to business operations
- Developing and implementing strategic electronic discovery plans and assisting with the development of production requests and deposition outlines
- Formulating accurate and proper responses and objections to requests for production and interrogatories
- Extracting relevant information from electronic mail, various desktop applications, and other electronic sources in a proper, timely, and cost-effective manner
- Reading and recovering data from obsolete tapes and disks for which hardware and software is no longer available
- Conducting high-speed searches of electronic data sets, including e-mail, and extracting relevant information for litigation at speeds ranging up to many gigabytes per hour
- Reducing huge data sets to a manageable size for analysis
- Recovering data that has been deleted, tampered with, damaged, or hidden
- Accessing password-protected and encrypted data
- Verifying dates and other file attributes and tracing user activity
- Providing data in printed or electronic formats that meet counsel's needs

Entrepreneur magazine in April 1996 wrote, "Unless your business has established systems for managing electronic data, one lawsuit could mean disaster." In summation, electronic evidence and discovery is a double-edged sword. In the dispute one party is trying to get information applicable to the dispute and the other side has to produce the evidence. The main areas for investigation are the companies' records management systems, e-mail, backup systems, and historic data archives. Therefore, it is necessary for corporate counsel or a companies' outside counsel to produce electronic data in the most cost-effective manner possible and use the pursuit of electronic discovery as a strategic litigation tool.

IV. TRIAL

Failing all attempts to resolve the dispute, the successful law firm now has to prepare for and conduct a trial. The arena moves from the conferences rooms of the opposing law offices to a courtroom. The courtroom has its own aura. Whether it is a jury trial or a bench trial a result will be reached as to the dispute. Fifty percent of the parties involved will be happy. The preparation and presentation of information at this stage becomes crucial. Accordingly, information systems play an important part. The first step is getting all the information coordinated for the trial.

A. Litigation Management Using the Internet

The Internet in a very short time has become the next industrial revolution. It has or will very shortly become intricate in the lives of every person on the planet. However, attorneys have been slow to use its potential for supporting litigation. In a complex case the amount of information to be managed and coordinated can run into millions of pages of transcripts. Law firms have used traditional software packages such as word processing, spreadsheets, and databases to achieve a level of management and coordination. However, Craig Freeman found that these traditional approaches have inherent weaknesses:

- Inconsistent, difficult software application interfaces
- Performance problems and difficult remote access
- Difficulties in sharing information among distributed counsel
- A large number of applications to control all case information

An Internet- or extranet-based solution can resolve the weaknesses in this traditional approach. An Internet browser coupled with a designed, easy-to-use interface can provide a comfortable "look and feel" interaction with the Internet. The processing can be accomplished by using server-side processing of information. This type of approach allows the successful law firm to retrieve information over the Internet

utilizing their current browser. The big bonus of this approach is that this can be achieved without requiring any additional software on the office computer. Moreover, utilizing the Internet through a browser means that an attorney can work on the case from any location that has access to the Internet.

The Internet by its very nature allows attorneys to share information from anywhere. The Internet allows legal teams the ability to instantly search and retrieve case-related documents from anywhere at any time using a simple Web browser. E-mail is the traditional method in this area. E-mail allows attorneys to notify each other of changes to the case, new information, or a recent court decision that will directly impact the dispute. In essence, this creates a seamless method of sharing information. The capabilities of one's e-mail system and its ability to interact with database systems allow all attorneys associated with the dispute to be automatically notified of updates when new or changed information becomes available.

Beyond e-mail, all information involved in the case can be stored and accessed through the Internet- or extranet-based solution. This means that information can be updated from anywhere an attorney can get access to the Internet. This allows for immediacy and current real-time information being at the fingertips of those who need access to it. Decision-making and coordination of the information needed to win the dispute can be now directed to this goal. In the past this was a rather cumbersome endeavor.

As information overload has become a tactical weapon in many large litigation disputes, it makes a strong case for storing case-critical data on the Internet, allowing access by the attorneys involved in a case that involves multiple law firms. This method allows equal and multiple access to the information in real time. As to the future, it is the only way to go.

In summary, the Internet is an important part of a successful law firm's arsenal in litigation. A successful law firm can harness the convenience and power that the Internet provides. According to Craig Freeman, the fundamental areas in using the Internet in this manner are:

- A browser-based system utilizing Microsoft's Internet Explorer or Netscape's Navigator
- No additional software requirements on client (i.e., attorneys) workstations
- A programmed interface to case information that is familiar yet powerful, such as software that mimics Yahoo
- A standards-based system allowing the use of e-mail and other information sharing capabilities
- A secure system with significant controls

1. The Courtroom and the Jury

The courtroom is a daunting place. The jury is now empanelled and the evidence is coordinated to present a good case. It is fundamental to recognize at the outset in this setting that the *jury will remember only 30% of what they hear. They will comprehend 80% of what they see and hear and the average reading level of juries has been claimed to be fourth grade in some areas.* Parallel to this fact is the development of information technology. Most courtrooms have been around a long time and therefore are not wired to accommodate the new technology that is currently available.

The first electronic courtroom was designed and equipped in Tucson, AZ in 1984 and cost $250,000. More technology courtrooms have been installed in Virginia and the latest is in Kansa City, MO. The cost to the courtroom in Kansas City was $62,000. The hardest part of the installation was raising the floor to accommodate the wiring needed to support the information technology. The digital courtroom creates a highly visual environment for trial presentation. Moreover, it reduces the amount of paperwork carried into the courtroom, reduces the time required to present information, and allows for fast retrieval.

In this courtroom display monitors are located on the bench, the podium, and both counsel tables. Ten-inch monitors are provided for the court reporter and the witness. A 37- in. monitor is used for the jury.

The technology courtroom has the following information systems:

- Audio visual systems
- Power podiums used to house the overhead projector, video camera, and VCR
- High speed ISDN lines
- Video phone
- Ability to teleconference

With the above equipment in place, the courtroom is now graphically and visually stimulating. This equipment allows the successful law firm the ability to demonstrate evidence in a more persuasive manner and to effectively present its case to the jury.

The ability to teleconference allows attorneys to present testimony of people who are not available to be in the courtroom. This saves the travel expense and the waiting time for expert witnesses and also saves time and money for all concerned. Teleconferencing allows the jury to instantly see and discuss progressively transmitted images without waiting for the full image to be received. It can also import images of documents, graphics, spreadsheets, or entire files to

share and mark up with others involved in the teleconference.

The VCR allows the jury to view depositions taken of witnesses. This is very useful for the attorney when the defendant says one thing at trial and another in the deposition. The tape can be set up to play from preset places on command. What a dramatic way to impeach a witness!!

Given the presentation of the information, the juries' attention to the information and retention for later use is greatly increased. RSI, Inc. found that this comes about due to the following:

- *Immediacy*—the information is quickly and dramatically presented to the jury.
- *Emphasis*—attorneys have the ability to underscore the message they want to present at a peak moment.
- *Organization*—With organization comes speed of access. With the click of a button the information is there for the jury to see.
- *Ease of modification*—The information can be modified as needed to illustrate one's point or change as the testimony is presented. The jury will be spellbound.
- *Permanency*—The ability to use the information presented at trial at a latter date.

2. Digital Advocacy

It is fundamental that attorneys be aware of the local rules so that they do not turn up to trial and find that the courtroom cannot accommodate what they want to present. The question is can the courtroom accommodate what they want to achieve?

As the successful law firm prepares for trial, it must think ahead with the technology courtroom in mind. It needs to be emphasized that it is important that the attorney does not ONLY think of technology in the courtroom, but the attorney must think about technology in his or her daily practice. The potential for technology and electronic presentation can be used in arbitration, mediation, settlement, and negotiation. The attorney must utilize a mixture of traditional demonstrative evidence with exhibits and new technologies for maximum impact and persuasion power. Well before trial the attorney must determine technology needs based on presentation type, size and layout of the courtroom, size of the jury, and requirements of the judge and court system.

If the courtroom supports multimedia presentations, the successful law firm can then store information on CD-ROM. Attorneys can also use digital cameras to take photos. These images can be edited and stored on CDs or laser disks for presentation with the click of a mouse. The laptop computer becomes the command center and fundamental to organizing information to be presented in this manner. It makes the multimedia connectivity more versatile.

Once an image is stored digitally, it can be increased in size to help visually impact one's point. Animation can be added to the presentation to illustrate a significant point or can be used to draw the juries' attention back to the task at hand. It provides a change of tempo to engage the juries' attention.

Do not present the technology; let the technology present the attorney When used at its best, technology is almost invisible. The attorney is the master of the power of persuasion. Technology holds the power to captivate and to enable the attorney to make quick, concise points for maximum retention.

3. The "John Madden"

In the first version of PowerPoint the user could use the mouse to work interactively on the screen with the presentation. The name John Madden comes from the football broadcaster for Fox Television who first used this idea to communicate to the audience how plays are run by the offense and defense during televised football games. It proved to be very popular. Electronic annotation such as a light pen can be used upon images on the screen to highlight points. This is interactive and is a very powerful way to reinforce a piece of information the attorney wants the jury to focus on. Also it can be used by a witness to demonstrate, on the image, where they were or what they did next. Both these activities give the jury the "feel" of being there.

4. Evidentiary Foundation

Multimedia presentations by their very nature and peoples' natural distrust of what they do not understand make it paramount that the attorney demonstrate effectively evidentiary foundation. In essence, what is presented can be trusted to represent what it is supposed to. To guarantee this, the presentation must have a reliable basis and must use scientifically approved and accepted expertise. Regarding photos, animation, and documents presented in a multimedia format, the attorney still has to have witnesses available to testify to their reliability and to lay the necessary foundation as to their authenticity.

When the jury retires to the jury room, the information presented through the multimedia format must now be reduced to a paper format for the jury to be able to request information they wish to review.

This fact must always be kept in mind, and an attorney must be prepared to have these documents available.

If the case goes to appeal, then the evidence still has to be available for review and a record of all images must be kept of what the jury saw during the trial. These are all considerations an attorney has to recognize in order to maintain the integrity of the new multimedia courtroom.

In summation, a multimedia courtroom empowers an attorney to enhance evidence and empowers their ability to communicate. With that power comes the responsibility to control it. It is one thing to present a case, it is another thing to overwhelm the jury.

B. The Future

Information systems will become more and more a part of the attorneys "briefcase," particularly in the following areas:

1. Internet

As the Internet develops and software for the Internet is produced for attorneys directly, its use from anywhere will be the main attraction.

2. Expert Systems and Decisions Support Systems

By its nature the legal profession should lend itself to expert and decision support systems given that it is rule based. In the past this observation has been overoptimistic. Leo van der Wees found that previous attempts have brought attorneys together for weekends to produce the rules for such a system. In the future, with developments in software in this area and as the more areas of the laws become codified, this area should expand. Future software will allow an attorney to type in the facts of the case; in turn, the software will render the judges decision. Currently, bankruptcy in Chapter 7 cases uses software to produce the documents to file with the court. As long as the information is true and correct, the trustee will send the debtor a discharge notice after the formalities of a creditor meeting and judges are observed.

3. Digital Jury Rooms

Given the digital courtroom, the next step is the digital jury room to allow the jury to review the information presented in trial in the same format as presented in the courtroom.

V. CONCLUSIONS

Regardless of all the new information systems, the legal profession is still about people—the litigants, the attorneys, the judge, and the jury. In spite of all the work that can be done in all the states of litigation, there is a saying in the United Kingdom, "There is nowt queer like folk." Literally translated, it means that when we try to understand people we are often wrong. With such sage advice in mind, information systems can still give the successful law firm the competitive advantage. New information systems compliment good legal skills, but were never meant to replace these skills. Look at the latest hi-tech army rifle equipped with a state-of-the-art laser sight designed for "fire and forget shooting." At the end of the weapon is a bayonet lug to attach a bayonet. In the end technology helps the person; if technology helps the person and if technology fails, it goes back to the beginning, i.e., "hand to hand" combat and the bayonet.

SEE ALSO THE FOLLOWING ARTICLES

Computer Viruses • Copyright Laws • Crime, Use of Computers in • Ethical Issues • Forensics • Privacy • Public Accounting Firms • Service Industry • Software Piracy • Year 2000 (Y2K) Bug Problems

BIBLIOGRAPHY

Alter, S. (1996). *Information systems.* Reading, MA: Benjamin Cummings.
Amicus Attorney, Gavel & Gown Software, Inc., Toronto, Ontario, Canada M5H 1L5
Casecentral.com, San Francisco, CA 94102
Compulit, Grand Rapids, MI 49512
Corel Corporation, Ottawa, Ontario, Canada K1S 1V7
DSI, LA Regional Office, Los Angeles, CA 90071
Electronic Evidence Discovery, Inc., Seattle, WA 98161
Oz, E. (1998). *Management information systems.* Cambridge, MA: Course Technology One Main Street.
RSI, Inc., Kansas City, MO 64108
ST Software Technology, Inc., Lincoln, NE 68512
Time Matters Software, Cary, NC 27511-4474

Library Applications

Johanna Olson Alexander
California State University, Bakersfield

GLOSSARY

aggregators Information intermediaries that partner with content providers to provide user search interfaces/systems and/or other content linkages or support.

bibliographic utilities Companies that, for a fee, facilitate capturing, creating, and accessing cataloging records for libraries.

Boolean searching Based on Boolean algebra developed by George Boole in the mid 1800s, a search method using structured logical search queries using the operators OR, AND, and NOT.

digital object identifier (DOI) "A system for identifying and exchanging intellectual property in the digital environment. It provides a framework for managing intellectual content, for linking customers with content suppliers, for facilitating electronic commerce, and enabling automated copyright management for all types of media." See http://www.doi.org/

document type definition (DTD) Defines, identifies, and specifies the processes of markup language coded documents. There are various DTDs.

Dublin Core An international and broadly based information-sector effort to develop a new, standard, but flexible metadata system to support interoperability among various information systems (NISO/ANSI Standard Z39.85-2001).

e-content Content or information in an electronic or digital format.

information content distribution chain All participants/steps in providing information including the information creator/author, publisher, programmer, distributor/vendor, database producer, search engine/information retrieval system, library or information service, portal, and communication provider.

integrated library systems (ILS) Library information systems or database management systems that integrate library process and service modules including public access, reference, circulation, billing, interlibrary lending, acquisitions, serials, and cataloging.

machine readable cataloging (MARC) Metadata of library resources in a form understood by computers.

metadata Information about information.

National Information Standards Organization (NISO) One of several standards organizations creating standards for the library and information industry.

natural language processing (NLP) Based on artificial intelligence (AI); computers are programmed to understand natural language rather than artificial query language.

portable document format (PDF) A file format used by Adobe Systems and libraries to download and transfer e-content journal articles, documents, etc.

relevancy ranking Ranking results of a database search by using algorithms and word/phrase weighting based on word location and occurrence.

syntax Database search protocols which dictate the forms of search queries including punctuation,

grammar, and the optimal way to structure a search within a database.

Z39.50 Information Retrieval Protocol (NISO Z39.50/ISO 23950) "Defines a standard way for two computers to communicate and share information. Designed to support searching and information retrieval—full-text documents, bibliographic data, images, and multimedia—this standard is based on client-server architecture and is fully operational over the Internet. Specifies access control, resource control, extended services, and a 'help' facility and addresses communication between the client and server." See http://www.niso.org/standards/standard_detail.cfm?std_id=465 [December 17, 2001].

ALL TYPES OF LIBRARIES, whether academic, medical, public, school, or special, provide information and services to assist users in accessing information. Information systems (IS) are integral to organizing, indexing, delivering, and managing library resources. Libraries archive, preserve, acquire, subscribe, catalog, index, store, and facilitate access to information. Libraries provide user services such as access, instruction, and reference assistance. Library IS applications are used throughout these processes, from information creation to information utilization.

I. INFORMATION SYSTEMS AND LIBRARY MANAGEMENT

A. History and Information Delivery

In 1962, Schultheiss, Culbertson, and Heiliger authored one of the first books on advanced library automation. *Advanced Data Processing in the University Library* describes *electronic* computers, punched cards, card readers, punched paper, and magnetic tape used for library processes. Much has changed since that time including hardware, software, infrastructure, and the information content distribution chain itself. Information creation, distribution, and access methods have been radically transformed by IS. Figure 1 illustrates current information infrastructure, access points, and delivery. While IS provide information more quickly, with increased access and less user effort, the process has become more complex, relying heavily on networked IS.

B. Digital and Virtual Library Environments

The phrase digital library generally refers to collections of materials in a digitized or electronic form. Today, new information is first created in a digital

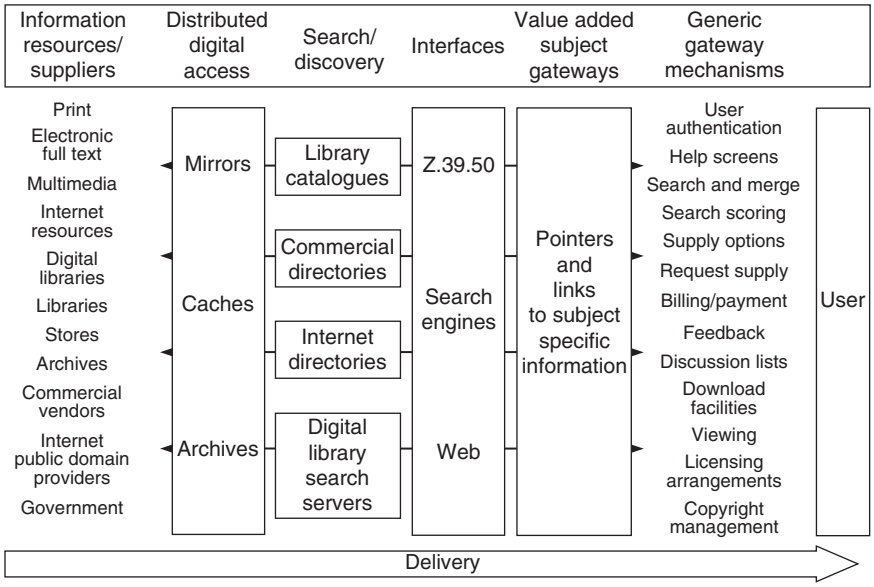


Figure 1 Infrastructure and access points. [Gartner, Inc., Gartner figure used in Neil McLean, "The Global Scholarly Information Infrastructure: The Quest for Sustainable Solutions" (Washington, DC: Coalition for Networked Information, January 19, 1998), Fig. 1. Available at http://www.cni.org/regconfs/1997/ukoln-content/repor~10.html [February 10, 2001]. Reprinted by permission of Gartner.]

format and then distributed in electronic, print, and media forms. Various programs have been implemented to retrospectively convert archival information to digital form for improved preservation, distribution, and space and cost savings. Digital information, or e-content, and the development of Internet and Web technologies allow information to be created, captured, archived, distributed, searched, and retrieved anywhere with the Web as portal. This has produced the *virtual library* or the *library without walls*. Libraries continue to purchase physical formats such as books, compact discs, DVDs, etc., and also acquire and/or provide access to licensed electronic information sources. Users have more options. They can access and use resources inside the library, check out information in physical packaged formats, and, increasingly, access and use library information from remote locations outside the "brick and mortar" library. It is in this digital/virtual library environment that IS are used to manage libraries and deliver resources.

II. LIBRARY INFORMATION SYSTEM BASICS

A. Bibliographic Structures and Systems

Information regarding library holdings and the actual information accessed from a library take on different but interrelated forms including:

- Bibliographic and retrievable field elements which describe a particular item or source and allow enhanced search capabilities
- Information formats including text, graphics, sound, video, computer software, maps, etc.
- Databases such as integrated IS, web-based information retrieval systems, and bibliographic and full text systems which provide control and access to these various elements, formats, and library collections

The foundation of these structures and systems is often based on industry or library/information science standards. Some of these elements and structures were used in a paper card catalog/print index era but have been adapted or improved in library IS. These standardized forms improve search capabilities among different vendor products and system platforms, making sharing and accessing information more efficient and compatible.

B. Bibliographic Records

Historically, the bibliographic record served as the basis for library information catalogs, identifying an information source by author, title, publication information, date, assigned subject headings describing the item, and possibly other fields such as a summary abstract, an acquisition number, or other standard number. While the bibliographic record is still important, new documentation standards and more sophisticated and/or unstructured information retrieval systems have enhanced searching and access capabilities.

1. MARC Record Basics

In order to use the bibliographic record in an electronic information system, the machine readable cataloging (MARC) record was developed in the 1960s. The MARC record is based on library cataloging conventions. In the United States, Canada, and Britain, these conventions are compiled in the *American Anglo Cataloging Rules* (presently AACR2R revised) and through the CONSER (Cooperative Online SERials) Program. The current U.S. and Canadian MARC format is called MARC21 and is an implementation of ANSI Standard Z39.2 (1994, revised 2001), the Information Interchange Format. There are various MARC standards throughout the world, but most have similar structures. MARC record files can be electronically transferred in mass, generally using tape loads, file transfer protocol (FTP), or a standard information retrieval protocol called NISO Z39.50. MARC contains data fields for retrieving and accessing pertinent bibliographic information about library materials including descriptions, content, locations, specific local annotations, formats, and numbering schemes. Computers read, translate, manipulate, and access MARC. The MARC record describes and identifies physical library holdings and increasingly, what is digitally accessible to library users.

Each MARC record has specific coded fields and headers that identify the information contained in particular fields. The MARC record structure has 10 basic field groups and contains 3 major sections including the *leader, directory,* and *variable fields*. Figure 2 illustrates a sample coded MARC record for the book *Make the Team*.

MARC coded or abbreviated fields facilitate effective use of computer disc storage space, provide consistency in record format, facilitate the sharing of MARC records among individual libraries, decrease duplication of effort, facilitate record access for

01041cam 2200265 a 450000100200000000300040002000500170002400800410004101000240008202000250010602000440013104000180017505000240019308200180021710000320023524500870026724600360035425000120039026000370040230000290043950000420046852002200051065000330073065000120076 3^###89048230#/AC/r91^DLC^19911106082810.9^891101s1990####maua###j######000#0#eng##^##$a###89048230#/AC/r91^##$a0316107514 :c12.95^##$a0316107506 (pbk.) :$c$5.95 ($6.95Can.)^##$aDLC$cDLC$dDLC^00$aGV943.25$b.B741990^00$a796.334/2$220^10$aBrenner, Richard J.,$d1941-^10$a Make the team.$pSoccer :$ba heads up guide to super soccer! /$cRichard J. Brenner. ^30$a Heads up guide to super soccer.^##$a1st ed.^##$aBoston :$bLittle, Brown,$cc1990.^##$a127 p. :$bill. ;$c19 cm.^##$a"A Sports illustrated for kids book."^##$aInstructions for improving soccer skills. Discusses dribbling, heading, playmaking, defense, conditioning, mental attitude, how to handle problems with coaches, parents, and other players, and the history of soccer. ^#0$a Soccer $v Juvenile literature.^#1$aSoccer.^\

Figure 2 Sample coded MARC record [Furrie, B. Part XI: A sample record in various formats. Available at http://lcweb.loc.gov/ marc/umb. Reprinted by permission of the Library of Congress].

interlibrary lending networks, and provide compatible and standard record formats and protocol among various library systems and vendor products.

2. MARC Fields and Electronic Resources

MARC fields not only describe information items but now also have the capability of linking bibliographic records to the entire document or information source. The 856 MARC field provides access to an electronic or digital source by direct Internet link. Other MARC fields of particular significance to electronic library systems are presented in work by Bluh (pp. 134–136).

Field	Purpose
506	Restrictions on access note
516	Type of computer file or data note
530	Other physical formats available
538	Mode of access and system requirement note
740	Added entry for electronic version of print source
776	International Standard Serial Number (ISSN) for electronic serial
856	Electronic location and access (URL: Universal Resource Locator)

3. Controlled Descriptors, Thesauri, and Authority Files

Predetermined subject descriptors, headings, and/or names describe the content of information items. Such descriptors, if used consistently within a database, provide controlled subject and name indexing and access. Descriptors, in a database thesaurus file, show subject relationships through conventions such as "see" and "see also" references and, in a hierarchical arrangement, indicate broader and narrower terms. Electronic thesaurus files are useful in searching databases, allowing direct search capabilities from thesaurus links. The *Thesaurus of ERIC Descriptors, Thesaurus of Psychological Index Terms,* and *Medical Subject Headings* (MeSH) of the National Library of Medicine are examples of controlled database thesauri.

Similar to database thesauri, cataloging authority control files and records are used to control and provide consistent forms for subjects, titles, series, authors, geographic locations, and proper names. There are several standard library authority sources including the *Library of Congress Name Authority file, Library of Congress Subject Headings,* and *Sears List of Subject Headings.* These authority sources provide standards, thereby eliminating or decreasing variant forms. This improves the quality and control of individual records and the database.

4. Classification and Identification Systems

Classification systems are organizational tools used to organize materials and collections. Alphanumeric, numeric, or alphabetical classification systems bring like subject materials together on the shelf—a sort of "parking" system. Classification systems are based on an organized or classified body of knowledge. Classification systems have been used most often with physical materials such as books and periodicals in print or microform formats. At first glance, in a virtual collection, classification seems less important. However, classification systems are not format dependent. A

classification is more than a shelf location designation. Classifications provide a means to browse by categorized subject, as with a "shelflist" both in physical and virtual formats, and to provide a basis for data analysis. Most libraries in the U.S. use one of the following systems: Library of Congress Classification, Dewey Decimal Classification, National Library of Medicine Classification, or other more specialized systems.

5. Barcodes and Identification Systems

While there is more emphasis on access vs ownership in the virtual library, inventory, control, and data management are still important issues. Library users continue to need and use print and other traditional formats. To assist in inventory control, barcodes are used to track the location of an item and its use status; to maintain a database of materials, user data, borrowing privileges, materials currently in a user's possession, and outstanding billing or material obligations.

Additionally, *identification systems* (not classification) are commonly used to identify information resource records. Identification number examples include the Library of Congress Control Number (LCCN), International Standard Serial Number (ISSN) for periodicals, and the International Standard Book Number (ISBN) for monographs.

C. Cataloging Bibliographic Utilities

Cataloging records are generally created by national and/or large libraries within a country. In the United States, most cataloging is created by the Library of Congress Cataloging Division and other major contributing libraries. Sharing contributed cataloging with other libraries saves time, money, and personnel costs. These few libraries create individual (but not necessarily unique) cataloging records for almost every published, media, or electronic work. Some libraries follow nonstandard and/or locally developed cataloging practices that may require alterations to shared cataloging records or creation of unique records. Individual libraries or library systems attach specific holdings, barcodes, and location/collection information to records. Companies that facilitate capturing, creating, and accessing cataloging records for a fee are called bibliographic utilities. Examples include the Online Computer Library Center (OCLC) and the Research Libraries Group (RLIN). These utilities provide specific functionality including the ability to alter and add holding codes to records, load large numbers of records into a database, and facilitate resource sharing (interlibrary lending) among library networks.

D. Integrated Library Systems (ILS)

Integrated library systems (ILS) are database management systems (DBMS) for libraries. ILS help manage library functions including acquiring, processing, cataloging, locating, linking, and distributing information and collections. ILS generally provide integrated modules for these various functional areas. Such function integration requires library organizations to rethink organizational structures, workflows, procedures, and policies related to acquisitions, bibliographic records, and public catalog screens. While basic ILS products are vendor driven, industry innovation, requirements for interoperability with other systems, and library/customer input help add functionality.

Periodic reviews of vendors and ILS are available from the "Automated System Marketplace" of the *Library Journal* (http://www.libraryjournal.com), *Library Technology Guides* (http://staffweb.library.vanderbilt.edu/breeding/ltg.html), *Integrated Library System Reports* (http://www.ilsr.com/index.htm), Project LIS (http://www.coe.missouri.edu/~is334/projects/Project_LIS), and issues of *Information Today*. Vendors also demonstrate products at major library and information system meetings. While systems vary, presently ILS/DBMS features typically include:

- Open system design
- Interoperability
- Client/server architecture
- Distributed networking
- Graphical user interfaces
- Compliance with such industry standards as relational DBMS, ANSI/NISO Z39.50, and MARC record importing and exporting protocols
- Report creation capability for assessment and tracking purposes, using relational DBMS and Structured Query Language (SQL)

Newer functionality can include:

- Enhanced electronic content linking
- Content enrichment such as book jacket images, tables of contents, book summaries, and reviews

- Multilanguage adaptability
- Interlibrary loan management modules
- Modules for other library collections and services

Translating coded MARC information into understandable and clear information is reliant on ILS online public access catalog (OPAC) database modules which search and display specified field information for public use. Most OPAC systems are web-based, provide customizable displays, and are one component of integrated library systems.

Purchasing an ILS usually requires a request for proposal (RFP) process. In addition to cost, reliability, and effectiveness factors, specific considerations regarding ILS selection include:

- Searching mechanisms and protocols and the ability to refine searches
- In-house control of screens and data
- Collection development support functions including available report generation and budget control
- Display, print, and download system functions
- Security and user authentication practices
- Integration of all library modules including public access, circulation or check out services, technical processing, acquisitions, cataloging, and serials
- Special areas such as archive, image, interlibrary lending, and reserves modules
- Interoperability with other information standards and platforms
- Type, size, and purpose of library

E. Metadata

Metadata are the data collected to describe other data or information describing information. It is a system of providing standard and common descriptions for diverse data and information, both structured and unstructured. MARC records are used to describe structured and standard record formats. In the Internet and Web environment, new metadata systems are being developed and defined for structured, less structured, and, more often, for unstructured forms of information. One such system is the Dublin Core.

The Dublin Core (named for the first organizational meeting in Dublin, Ohio, in 1995) is an international and broadly based information-sector effort to provide a flexible metadata standard, to support interoperability among various information systems. The Dublin Core standard was approved in 2001 by the American National Standards Institute (ANSI Z39.85-2001) and consists of fifteen elements.

Title	Contributor	Source
Creator	Date	Language
Subject	Type	Relation
Description	Format	Coverage
Publisher	Identifier	Rights

More detailed information and developments regarding the Dublin Core are available at http://www.niso.org/standards/standard_detail.cfm?std_id=725 and http://dublincore.org/.

Another metadata system example is the Government Information Locator Service (GILS) used by the U. S. Government Printing Office. The International Federation of Library Associations and Institutions (IFLA) web site (http://www.ifla.org/II/metadata.htm) presents a list of links to other metadata systems in use or in development. Specifications, called *crosswalks,* have been developed to map one metadata system to another.

F. Standards and Development

Moving experimental systems and technologies from testing to commercial use has been enhanced by standards. Players in the information distribution chain, including creators, industry, vendors, and libraries cooperatively working together, have set standards that use viable and cutting-edge systems to deliver information. Electronic information standards of data interchange, formatting, programming and markup languages have increased the interoperability among library databases. Many of these standards and specifications are now in use or are being adapted for commercial library products.

In the field of library IS, standards organizations include the National Information Standards Organization (NISO), American National Standards Institute (ANSI), Library of Congress (LC), World Wide Web Consortium (W3C), Internet Engineering Task Force (IETF), Book Industry Study Group (BISG), Open e-book Forum (OeBF), EDItEUR, and International Standards Organization (ISO), among others.

Some libraries and library consortia have developed in-house, large- and small-scale information systems. Generally, however, in-house information system development and maintenance is too costly for most mid- to small-sized libraries. Commercial enterprises or cooperative library and commercial projects are able to produce large-scale systems at lower unit costs with a large customer base, greater capital, server and personnel resources for product development, and research and technical support.

III. INFRASTRUCTURE

A. General Factors

Library IS infrastructure requires the same considerations as other IS applications including:

- Compatibility, reliability, resiliency, scalability, speed, memory, capacity
- Quality, customization, and flexible screen design
- Technical support availability
- Funding for software and database site licenses, equipment purchases and replacement, and training
- Vendor stability

Regarding library specifications, Mayo and Nelson, in *Wired for the Future: Developing Your Library Technology Plan* (pp. 57–58), list the following factors to be considered.

What Do You Look for When Evaluating Possible Products and Services?

- Number of sites needing access
- Number of users
- Location of the users: in-library only, access from off-site, or both
- Licensing restrictions on the use of the data
- Frequency of updates of the data contained in the resource and the currency of the data
- Authentication of users
- Indexing and retrieval features
- User interface to the product, including the special interface needs of persons with disabilities
- Costs to support the product
- Training requirements to support usage of the materials, including training the staff and training the public

B. Other Considerations

The compatibility and integration of library and larger institutional systems should also be considered—hardware, software, operating, and network systems. Although these technologies change, currently, most library systems are networked, server-based systems. Barber's "Networking Checklists for Library Managers" is a useful tool in analyzing and planning infrastructure.

IV. DATABASE STORAGE AND ACCESS

"Library databases" can refer to ILS, online public access catalogs, other electronically searchable/accessible resources, and/or licensed databases. Licensed databases provide indexing, abstracting, and/or electronic journals and information resources, usually for a subscription licensing fee. Library database discussion involves database storage and impacts of Web/Internet access.

A. CD-ROM

In general, libraries have moved away from databases requiring standalone workstations and single use CD-ROM database technology. Web and in-house server databases offer simultaneous user access, greater interactivity, integration, and currency. While information is still distributed in CD-ROM format (software, media, geographical spatial data, data, and text sources), this format is decreasing in popularity as a library database storage format due to its limitations. Some smaller libraries may find CD-ROM source formats and microcomputer applications more affordable than licensed networked resources. Libraries selecting single user CD-ROM formats must provide standalone workstations with appropriate software to support these various products.

B. Web/Internet

Internet and Web technologies have revolutionized libraries by allowing linkages and access from a graphical user interface (GUI) or browser. This allows simultaneous network access to varied types of information and sources, providing data and information transfer, manipulation, and linkages to other resources. As an example, Java (a computer language that runs on any computer and is used in a network/web environment) allows the use of library system program applications that are compatible across platforms. The ability to transfer information is also a great user convenience. Users can print materials, download information, and/or send the material electronically to an e-mail address. Libraries' home pages provide links to both print, in-house, and remote electronic database and Internet sources from one central access point. Alliances among information creators, publishers, aggregators, portal vendors, and Internet service providers through the web have increasingly blurred the content distribution chain. However, it has also made the process more seamless to library users. This has increased both the use and utility of information resources.

V. DATABASE SEARCH AND RETRIEVAL SYSTEMS

A. Search Systems, Syntax, and Information Retrieval

All library databases have search rules dictating what is searchable and the search syntax—meaning the form of a search query and the optimal way to structure a search. Two concepts pertinent to information searching and retrieval are *recall* and *relevancy*. *Recall* refers to the number of relevant references, articles, books, or other information items retrieved from a search. *Relevancy,* or *precision,* refers to the degree to which items retrieved are pertinent and germane to the user's request. Search systems should provide a balance of recall and relevancy. The focus of this section is on licensed or traditional library database search systems rather than Internet search engines. However, many of these conventions are used in both arenas and, in some cases, are used in combination. Most context-based and structured information systems include the following elements.

1. Common Search Elements

Boolean logic and operators (based on Boolean algebra) are used in most information databases, providing the ability to combine synonyms and variant concepts together to access relevant items. AND, OR, and NOT are the basic Boolean connectors.

Nesting is used to show search logic and the order in which Boolean commands will be performed. Nesting is commonly indicated using parentheses.

Example

((higher education OR universities OR colleges) AND students AND costs) NOT (community colleges or junior colleges).

Adjacency and proximity operators narrow or broaden a search by indicating word relationships in context—words' proximity to each other in the reference or text. Use of adjacency and proximity operators should improve search results. Words found closer together in context usually indicate increased relevancy.

Example

Higher education NEAR student costs would be a more specific search than *higher education AND student costs.*

Truncation and wild card or universal character searching allow the user to search word roots or portions of words with variant endings or variant spellings. A symbol, such as an asterisk, signifies missing characters that should be searched. Some databases automatically search for root word stems.

Example

Spous* would retrieve spouse, spouses, or spousal.

Key word searching allows searching by any word within searchable database fields. Generally, a list of common and incidental "stop words" are excluded from database searches.

Related-term searching uses a built in thesaurus to cross reference search terms. For instance, a searcher might input the word spouse and, in related-term searching, a term such as marriage would also be searched.

Field searching allows the searcher to confine search terms and phrases to particular fields, providing more concise and relevant search results.

Phrase searching allows searching for bound phrases, signaled by symbols (such as quotes) or by automatically searching adjacent words as bound phrases.

Search limits allow the user to limit a query by any number of parameters such as date, language, source document, and/or information type (scholarly or popular, full text or bibliographic reference, abstract or citation).

Search histories allow a user to revise or edit previous searches based on a review of the search results.

2. Special Elements

Examples of more specialized search retrieval systems follow.

Search mapping connects or maps variant concepts or resources together. *Chemical Abstracts' Chemname,* the *Unified Medical Language System* (UMLS), and Institute for Scientific Information (ISI) databases are examples of mapping database systems. The *Chemical Abstract* mapping system allows a user to search a known chemical name and all other related substance and brand names. The UMLS provides a similar function by pulling together multiple thesauri of differing medical databases and indicating all related terms, again, mapping one term to the others. ISI products are based on "citation indexing" where searchers look for relevant and related research (clusters) based or mapped on a primary source's citations and other authors' citing the primary source.

Multilanguage information systems and translation are provided in some database search systems. These systems handle diacritics and various languages.

Natural language searching is based on artificial intelligence (AI) where computers are programmed to

understand natural language queries. Natural language searching allows the user to input a question or phrase. This is simpler for the user because the computer and search system do the work behind the scenes, using the common search elements listed in the previous section and relevancy ranking based on word location and occurrence. Natural language results may not be as precise as searcher controlled Boolean searches, etc. Natural language searching is available in many Internet search engines and in some more traditional library databases.

Relevancy sorting or ranking is based on statistical weighting and algorithms. The purpose of this mechanism is to draw directly from the materials that appear to be the most relevant based on the user's request. The U. S. Congressional *THOMAS* system provides access to congressional texts and summaries of bills, public laws, votes, the *Congressional Record,* and committee reports. This is a large text system that provides searchers with options of Boolean searching or relevancy ranking based on weighted searches using an algorithm. Algorithms and weighting vary among database systems but generally, term frequency, location, and adjacency are used in the formula.

Fuzzy set searching recognizes that although a search query has resulted in a set of retrieved items, some of the items are more relevant to the actual query than others. Fuzzy logic assists users by finding and ranking results and then identifying how close a match the item is to the original query—"closest matches," "close matches," "not as close matches," and catching incorrect or misspelled terms. In terms of incorrect input, the system will indicate that something close was found, although not an exact match. This provides the user an opportunity to confirm or edit the search query. The U. S. *National Criminal Justice Reference Service* using Convera Retrieval Ware uses fuzzy logic in its search system. The retrieved set is then ranked by relevancy.

3. Intelligent Technologies

Lancaster and Warner note that AI, expert system (ES), decision support system (DSS), and knowledge based system (KBS) applications in libraries have often only been prototypes. However, in addition to these technologies being integrated in some database search engines, there are operational library applications for database/source selection, collection decision control systems, and adaptive technologies. Source selection systems assist users in identifying the best group of sources or databases using successive menus or question-answer methods to narrow the topic (see http://www.ala.org/rusa/mars/expertsys.html). Collection decision support systems may be a component of a larger system, built with an off-the-shelf program, or a specially designed program such as PAD (Periodical Analysis Database) (http://home.kscable.com/bobweeks/pad/pad.html).

Intelligent search agents are most often used in current awareness, selective dissemination of information (SDI), or alerting service tools (push technology). The "agent," operating on selected parameters or profiles set by a user, at regular intervals searches selected databases and/or Web sites and transfers the information to the user's e-mail or other profile area. *Elsevier ScienceDirect* has such a program titled, "My Alerts." Aaron (in Ensor, pp. 249–250) provides examples of library intelligent search agent applications that include:

- New-title alerts
- Subject alerts from online public access catalogs (OPACs)
- Overdue and availability notices from circulation systems
- Articles of topical interest from a database
- Publisher dispatch data about the status of issues
- Tables of contents of selected journals
- Title changes and publication status changes

ES searching relies on a developed or developing knowledge base (KBS). ES are computer systems that perform functions normally handled by humans and that learn from these functions and transactions. AskJeeves (http://askjeeves.com/) does this. While AskJeeves is a commercial search engine/Internet resource and is not strictly a library database, it is a good example of ES searching. When a question is asked, relevant sites that have proven useful to previous users are retrieved. The search system is therefore relying on a developing knowledge base. ES searching is not widely used in library information retrieval systems, but there are instances of its possible application in live real-time reference interaction over the Internet. Data from query responses received can be stored for later use.

B. Structured vs Unstructured Information

Information can be divided into two broad categories: structured and unstructured. Structured information is usually classified and indexed based on standardized methods. Metadata systems are used to access structured information databases. Unstructured information is found on the Internet, in Web documents and sites, and in graphic, sound, or media files.

Unstructured files, until recently, often had no set pattern of indexing or classification. Instead, information extraction systems such as natural language processing (NLP) "seek" the material (not index it) or rank materials by relevancy based on algorithms, word location, and occurrence.

Structured and unstructured information databases and access systems are increasingly converging. Structured information database searching is improved by using newer and popular techniques such as NLP, results ranking, weighted term searching, using sample results to "find more like this," and search query assistance such as searching additional terms. These techniques are used widely in Internet search engines. Unstructured information searches are also improved by the introduction of structured access points based on metadata such as the Dublin Core.

VI. ELECTRONIC CONTENT (E-CONTENT)

Content creation, publishing, distribution, and transfer have changed dramatically because of text, image, and sound digitization. There are distinct formats of electronic content (e-content) but the linkages among these different formats are rapidly increasing. The resources discussed in the following section are available through licensed, pay-per-use, and/or free databases usually accessible via the Internet/Web.

A. e-Periodicals

Full text periodicals (including e-journals or e-zines, short for e-magazines) are usually text or HTML (HyperText Markup Language) files. These files are cheaper to produce, more easily downloaded, take less computer disc space than graphics and image formats, and are more easily manipulated and transferred. Full text periodicals usually lack graphics and are more affordable for the producer, but are less helpful to the user.

Full image periodical articles give the reader text and graphics such as charts, illustrations, etc. Generally, these image files, accessible from the Internet/Web through a browser, are dependent upon external software programs (plug-ins) for creating, transferring, and reading files. Such programs as *Adobe Acrobat* are often used in conjunction with full image files but require the user to have access to the free reader program. These graphic files are in a transferable form called portable document format (PDF).

Increasingly, e-content providers are converting to XML formats (eXtensible Markup Language) to provide content *repurposing* or cross-media publishing—that is, the creation of content that can be distributed in multiple formats rather than creating content for one particular media. Content can then be made available through various means such as the Web, personal digital assistants, print, etc.

There are a variety of e-periodical "packages" or products. These packages may be provided by publishers (or alliances of publishers) such as *Project Muse* from Johns Hopkins University Press or Elsevier's *ScienceDirect;* library community retrospective and archival backfile initiatives such as *JSTOR* (journal storage project); publisher and scholarly society partnerships such as Stanford University's *HighWire Press* which focuses on value-added electronic functionality and graphics in science, technology, and medical journals; aggregator services such as EBSCOHost, which negotiate full text/image access to periodicals; commercial publishers that provide paper subscriptions and limited electronic access; and electronic only journals. These packages may be free but are more often fee-based licensed resources.

Other initiatives have sought to speed, expand, and archive scholarly communication within the research community and then provide access over the Internet. One example is *arXiv,* a free electronic preprint or e-print archive of scientific research in physics, mathematics, and other scientific areas. Authors can submit, replace, and/or remove papers from the *arXiv* collection and researchers can access the materials.

B. Indexing and Abstracting Databases

Databases that provide indexing, bibliographic records, and abstracts (not full text/image) are another type of e-content form. These types of basic bibliographic/abstracting databases are enhanced by linkages to digital full text/image materials. Some of these databases provide links to selected full text/image materials but increasingly full e-content inclusion in the database or linkages to full e-content will be mandatory for a vendor to stay competitive.

C. Linking Software and Services

E-journals/periodicals are most useful when the title has a stable URL and/or a persistent URL (PURL) allowing a direct and permanent link that is maintained in metadata search information. Some aggregators provide only e-journal access through their search interface and do not provide stable URLs. To improve electronic resource access, some aggregators have ne-

gotiated stable URL connections for customers and developed methods to load 856 MARC field information into library databases and ILS. This allows users to access e-periodicals directly from library online catalogs. New aggregator *linking software and services* are cropping up. These services may have variant ways of processing or even slightly different products, but all are based on OpenURL specifications. Linking companies that have access to multiple vendors and database producers are now acting as link aggregators. Examples of these linking services include *jake: Jointly Administered Knowledge Environment* (http://jakedb.org/) and Openly Informatics Inc. (http://www.openly.com/link.openly/).

CrossRef is also based on OpenURL specifications and is a cooperative initiative of publishers. Implementation of *CrossRef* means a "researcher can click on a reference citation in a journal and immediately access the cited [reference, abstract], or article . . . [using] Digital Object Identifiers (DOI), which are tagged to article metadata supplied by the participating publishers." (Publishers International Linking Association)

Linking technologies are now being integrated into ILS products. SFX is an example of a citation linking system based on OpenURL. SFX is being used by the ILS company Ex Libris, allowing customizable searching and access by the library " . . . to dynamically create [context-sensitive] links that fully integrate . . . [a library's] information resources regardless of who hosts them—the library itself or external information providers." (http://www.sfxit.com/)

D. Electronic Books (e-Books)

Electronic books, or e-books, are becoming more prevalent but have not had the popularity once forecast. Nevertheless, e-books are finding their way into academic and commercial markets. E-books come in forms including: (1) e-book apparatus that hold downloaded books, (2) entire book collections accessible via the Web, or (3) electronic reference sources. E-books may be freely available, as in the *Gutenberg Project* (http://promo.net/pg/), the *National Academy Press* (http://www.nap.edu), or *xrefer* (http://xrefer.com/), or for purchase or subscription, as is the case with *netLibrary* (http://www.netlibrary.com). These systems allow the user to search the book text, review and read, possibly highlight sections and/or take notes online, print and download selections, and, depending on the pricing and licensing model, "check out" the e-book for in-depth reading and research. Other e-book reference sources may provide searching but not browsing. If there is a stable URL link, the e-book can be cataloged in a library's ILS providing a direct link to the electronic book. Some of these systems are self-contained systems that do not require any special software while others require plug-ins for specific access levels. Due to copyright protection issues and pricing model experimentation, there is a good deal of variation in the e-book industry. The National Institute of Standards and Technology, Open eBook Forum (OeBF), and the American Association of Publishers are some of the e-book standards setting organizations.

E. Other Formats Including Data, Maps, and GIS

Statistical databases present data in HTML, PDF, or other easily readable formats. Many of these databases also allow data extraction for customizable reports and/or downloading in multiple formats such as comma-delimited, HTML, or spreadsheet formats. These variations are convenient, allowing the user to view and/or download data in a textual format or a spreadsheet format for calculation purposes. Appropriate plug-ins or data extraction programs are sometimes required. XML formats used with various tools will allow additional manipulation capabilities of research data. "XML Cover Pages" (http://xml.coverpages.org/) provides a useful list of XML applications in library, archival, and museum settings.

Cartographic and geospatial data, digital maps, and individual print maps are available in libraries. Map products range from easily accessed and updated digital maps and mapping systems available through the Web to highly specialized and technical geographical information systems (GIS). General issues of concern to libraries include:

- Indexing and accessing both print and digital map collections
- Ensuring quality relating to accuracy, print quality, and scale
- Providing appropriate hardware, software, and trained personnel to provide quality maps and services

VII. DISTRIBUTED ACCESS AND SEARCHING

A. Remote Access

Library IS in an Internet environment allow access from remote locations. Internet service providers are the portal from which users can access library

resources, but because of licensing agreements and user restrictions, users must first be authenticated before gaining remote online access.

B. User Authentication and Security

User authentication and security are important issues for libraries and information system managers. Database license agreements prescribe the user population authorized to access particular databases. Library systems must have some means of securely authenticating authorized users. A number of methods are used for authentication: IP (Internet Protocol Address) authorization, use of a proxy server that checks and verifies authorized users before allowing the user access to databases, individual database passwords, and/or digital certificates/signatures which use encryption and are used in e-commerce. Important issues to consider in selecting an authentication method include the number of users and user groups, number of access locations, limitations, flexibility in allowing remote use of resources, user-ease, infrastructure, resources, and staffing available to implement and control systems.

C. Z39.50 Information Retrieval Protocol

The ANSI/NISO Z39.50 Information Retrieval Protocol "provides a generalized mechanism [model] for transmitting and managing queries and the result sets of records created by those queries. . . . [is] capable of working with many data types and in many application contexts." (Needleman, pp. 160-161). Specifically, the Z39.50 protocol specifies:

- A client/server-based protocol for information retrieval
- Procedures and structures for a client to (a) search a database provided by a server, (b) retrieve database records identified by a search, (c) scan a term list, and (d) sort a result set
- Access control, resource control, extended services, and a "help" facility (ANSI/NISO Z39.50-1995, Abstract)

Z39.50 has been touted as a means of performing one-stop and multiple database searching. Used in conjunction with application profiles and integrated library systems and/or Z39.50 compliant database systems, Z39.50 has been used as a mechanism to create virtual union (cooperative) catalogs from various libraries' catalogs and to facilitate distributed searching of various databases. There are problems with implementation and Lynch outlines some of the possible difficulties. Inconsistent search results are possible due to (1) databases' indexing inconsistencies and variations, and (2) reliance on the "lowest common denominator" in terms of supported query language usable among various databases. Reliability and response times can also be impacted by the slowest part of the system and/or network such as an individual library's online catalog system, network problems, and/or load factors involved in processing and aggregating retrieved records at smaller installations. Nonetheless, a fairly large number of ILS and library database systems are Z39.50 compliant, allowing users to search multiple databases simultaneously. Examples include NLM Gateway (http://gateway.nlm.nih.gov/gw/Cmd) and Searchlight (http://searchlight.cdlib.org/cgi-bin/searchlight). Other Z39.50 compliant vendors are listed at http://lcweb.loc.gov/z3950/agency/register/entries.html.

New products are being introduced which provide multiple database searching capabilities without databases having to be Z39.50 compliant. An example is *WebFeat*, which provides a single search interface for all Web-based in-house, ILS, and outside databases.

VIII. APPLICATIONS

The following section describes specific applications of IS in particular library service areas. Table I summarizes information systems and application areas.

A. Acquisitions and Collection Development

Collection development is the systematic assessment, selection, and deselection of library resources. Librarian selectors/bibliographers use vendor information systems in conjunction with ILS to build library collections. Automated vendor collection development and acquisition functions can include:

- Approval programs (automatically receiving books for a library based on a preselected profile)
- Electronic selection programs providing online reviews and selection bibliographies of books and other information resources and notification services
- Electronic ordering, payment, and claims through Electronic Data Interchange (EDI)
- Online transaction security

Table I Information Systems and Library Application Examples

Information system							
Electronic data interchange (EDI)	Decision support systems	Collaborative computing	Geographical information systems (GIS)	Media and virtual information systems	Expert systems & knowledge-based systems	Metadata systems & retrieval	Natural language systems
Applications							
Library acquisitions	Electronic measure, assessment, and report writing systems	Library instruction & interactive reference services	Map and spatial data collections	Internet, Web, bibliographic records, & electronic text systems	SDI or current awareness systems	Integrated library systems and library catalogs	Bibliographic & electronic text systems

- Cataloging interface and shelf-ready processing
- Report generating capabilities

B. Archives and Special Collections in a Digital Environment

Archival and special collections include text, art, objects, photographs, geographical data, maps, sound, and other media. These collections present special issues because of unusual and nonstandard formats, conservation/preservation requirements, and sometimes the costly and fragile nature of materials. There are a number of digital archive programs in large institutions, but increasingly, smaller libraries are also participating.

While archive digitization is not directly within the scope of this text, systems used to access collections are relevant. An important standard is the Library of Congress' Encoded Archival Description Document Type Definition (EAD DTD) used in formatting archival finding aids and indexes. (A finding aid is a detailed descriptive guide or inventory of archive or museum collections.) Metadata systems are used to define, describe, and access digital archival collections. A helpful source for archival metadata and digital imaging projects is the Cornell University Library, Department of Preservation and Conservation's Web site, *Moving Theory into Practice: Digital Imaging Tutorial* available at http://www.library.cornell.edu/preservation/tutorial/index.html. The site outlines three types of digital image metadata: descriptive, structural, and administrative. Rothenberg's *Avoiding Technological Quicksand: Finding a Viable Technical Foundation for Digital Preservation* provides an overview of the issues and problems involved with electronic preservation projects.

C. Reference Services

1. Library Users

Reference services are designed to help users access and use library resources. Reference services are delivered in-person, by phone, electronically through Web sites, by e-mail, or through collaborative computing. Library personnel help users select appropriate databases and resources, structure effective search strategies, identify pertinent print, electronic, and media resources, and find additional assistance by referral to other resources.

User needs are changing. While IS make research more effective and efficient, users now face information overload. Users must evaluate the reliability and authority of multiple information sources. Users may be using systems and sources remotely, where on-site assistance is not readily available. New ways have been implemented to provide remote assistance online, make databases more intuitive, and assist the user interactively with online technologies and other information assistance enhancements.

2. Reference Collections

Reference sources can be accessed electronically through the Web, providing improved search capabilities, simultaneous multiuser access, and downloading, printing, and/or e-mailing capabilities. Online sources can be updated more frequently; combine text, statistical data, sound, and media; offer links to outside source materials; and in some cases, offer interactive functionality.

The advent of information systems has leveled the playing field for distant education and remote library

services. Access to electronic sources through IS allows improved access to rural or isolated areas and to students on residential campuses as well as students at off-campus sites or through distance/online programs.

3. Virtual Reference Services

Virtual reference services currently include virtual reference collections found on web sites, e-mail reference transactions provided synchronously, and live/real-time online reference programs providing 24-hour, 7-days-a-week (24/7) or specified time block service.

Live online reference programs and systems are similar to customer service or help-desk software programs used in commercial enterprises. These systems allow real-time interaction between the user and library personnel and service at varied locations and time zones around the world. Some of these systems have the ability to capture a knowledge base to assist the librarian and user with future questions.

4. Web-Based Databases and Library Home Pages

Internet/Web-based databases provide authorized users direct access to varied search capabilities from basic to advanced, links to other resources, and simultaneous use of a database with other users. Web-based databases loaded on a local server allow the local institution in-house control over public access screens, local holdings, and information linked to databases.

Library Web pages are usually the gateway to library resource access, the OPAC, and databases, making Web page design important. Garlock and Piontek, in *Designing Web Interfaces to Library Services and Resources* (pp. 4–8) provide nine principles to aid library personnel in Web and interface design.

- Plan your structure carefully.
- Let content inform design.
- Be consistent.
- Create an intuitive Web site.
- Make sure your interface is compatible and accessible.
- Provide a solid navigational base.
- Design your interface for your audience.
- Put user input into perspective.
- Be aware of the dynamic nature of the Web.

D. Adaptive Technology, Universal Design, and Disabilities Legislation

The Americans with Disabilities Act (ADA) of 1990, The Rehabilitation Act of 1973, and various revisions, specifically mandate library and information environment access for persons with disabilities. In addition to legislative direction, making information systems accessible and useable by all individuals should be a goal of information providers. This entails providing accessible library Web sites, selecting library databases that are compatible with adaptive technologies, and providing adaptive equipment and software to access library systems. Many of these adaptive technologies use voice and visual recognition patterns based on artificial intelligence programs. Mates describes IS adaptive technologies including screen reader/synthesizers, Braille software translators, speech recognition software, and text browser software.

E. Information Competency and Instruction

Library information competency means teaching users the best way to frame research questions, search, access, evaluate, and transfer pertinent information. IS are more sophisticated and complicated but often provide at least *some* measure of output. While a search may produce good recall, relevancy may suffer without practical search limits. Resultant content may be less than adequate due to the ease of putting *any* information on the Web. For these reasons, libraries provide reference and instructional services. User needs and behavior have brought about a number of initiatives including:

- Information competency components in library/user instruction
- Use of collaborative computing and chat systems for library/information instruction
- Improved Web and database designs that are more intuitive to the user
- Intelligent technologies that assist users in selecting databases and/or information sources and that perform enhanced searches

F. Plagiarism Detection

Educators and librarians involved with teaching information competency are concerned about plagia-

rism, particularly in academic settings. The Internet has exacerbated the problem because it is easier to copy source material, find term papers on the Web, and cut and paste material. On the other hand, IS also make it easier to detect plagiarized work. An example is illustrated in turnitin.com's plagiarism.org service (http://plagiarism.org/solution.html). The system uses information extraction and machine learning methods. Patterns or "digital fingerprints" are made of the content/paper and are compared with Internet documents and the plagiarism.org database. This process is used to identify similar patterns through machine learning.

G. Access and Circulation Services

ILS have improved management information systems by providing canned and customizable reports concerning collection and database use; circulation, workflow, and workload statistics; and automatically generating periodic notices for fines, overdue, lost, and recalled materials.

In addition to barcode systems discussed earlier, other information tracking and inventory systems have been introduced. The digital smart chip is being used by some libraries. A radio frequency identification (RFID) smart chip is inserted into library materials. The smart chip actually stores bibliographic data about the item. The system assists in inventory and shelf-reading, providing security against material theft and streamlining checkout processes. Smart cards are used for library user identification, to access databases, to check out materials, and as debit cards for fines, printing, etc.

H. Reserves

Reserve collections are high use collections and materials that have special check out limitations and are usually reserved for students enrolled in specific courses. While libraries still maintain physical reserve collections many ILS and/or academic libraries have electronic reserve collection capabilities. These electronic reserve collection functions can provide electronic indexing of the collection, simultaneous multiple user and remote access, archival functions, and copyright tracking. Most of these systems use Web technologies and ILS modules for cataloging and describing materials by course, instructor, and bibliographic information.

I. Automated Storage and Retrieval

Robotics and AI have been applied in a limited way to library collections. In the 1990s there was interest in library automated storage and retrieval systems (AS/RS). Decreased budgets for new buildings and a continued need for compact storage for older, less used collections were the impetus for using AS/R systems. This technology, used in manufacturing and inventory dependent businesses, was adapted for library storage and retrieval, linking information and robotic systems that retrieve materials from compact storage. A few libraries have adapted this technology; but, with growing emphasis on digitized information rather than storage of print materials, interest in AR/S seems to have waned.

J. Interlibrary Loan (ILL)

Before there were adequate information systems available, interlibrary loan (ILL) processes—one library borrowing an item from another—were tedious and lengthy tasks. Information systems have revolutionized this process. Interlibrary lending modules and bibliographic utilities have improved both service quality and speed. Library personnel can request materials—books, journal articles, and media—from a list of libraries known to have the material. Once the request is made, the system allows the lending library time to fill it. If the request goes unfilled it is automatically passed to the next lending library until it is filled. While books and media still must be shipped, articles, graphics, and portions of books can be electronically delivered over the Internet with document delivery software such as *Ariel for Windows* from the Research Libraries Group, and delivered to the user as a PDF file.

Statistical packages for monitoring and assessing interlibrary loan services are available and provide cost information, number of borrowing and lending transactions, copyright control and payment, transaction turnaround times, nonfill rates, and information useful for collection development purposes.

There are some software programs and systems that provide user initiated ILL service with limited library mediation. Users can search consortia library catalogs with Z39.50 broadcast searching, submit requests without mediation from the borrowing library, be notified of request status, and have materials delivered to the borrowing library for user access. One example of this is the "Borrow Direct Project" of Columbia University, University of Pennsylvania, and Yale University.

K. Application Outsourcing

Some library IS services can be outsourced to commercial entities. There is debate concerning the cost-effectiveness, quality received, and relevancy of outsourcing versus using in-house personnel and resources. Improved information systems create new system needs and, at the same time, allow commercial companies to provide solutions using additional, cost-effective systems. Sometimes commercial companies can expedite jobs faster or at lower unit costs. To see whether outsourcing certain system functions is viable, library personnel must weigh efficiency, cost, time, privacy and confidentiality issues, and knowledge required of any particular job. In many ways, the efficiency or cost-effectiveness depends on the situation, the library, and the project size and parameters. There are a variety of outsourcing companies and products.

- Rather than libraries having to contract with multiple publishers, information jobbers or aggregators provide acquisition services and/or serial and book title acquisition packages. These types of services assist library personnel in acquiring both print and electronic source materials.
- Digital resource control/management services have evolved with increased electronic resources. As an example, electronic journals are available from many different vendors. Some require licensed access while others are free. Some have stable URLs and others are only accessible through an aggregator. Some of these e-journal aggregators have changing e-journal title collections and various date availabilities due to publisher contractual agreements. In addition to electronic access to some journals, libraries may also subscribe to other journals in paper or microform formats. Electronic journal management systems have been developed to assist libraries manage, track, and provide records of these various electronic and paper journal holdings. One example is *SerialSSolutions* (http://www.serialssolutions.com/Home.asp) which tracks electronically available periodicals and provides records and links to libraries.
- Some vendors provide shelf-ready books and/or cataloging records transferred to a library's catalog system.
- Conversion of a Library's MARC records from one system to another can be problematic due to varying record formats. Outsourcing companies often handle conversion projects, database cleanup, and cataloging authority file updates more cost-effectively and efficiently than an individual library.
- Some libraries are using an *application service provider* (ASP) model for their systems. This means that the library leases access to application software hosted on the vendor's remote site rather than maintaining library servers, running applications, and storing ILS data on-site.
- Designed to provide electronic live backup reference service for libraries, a fee-based 24/7 electronic reference Web-based center is provided by Library Systems and Services, LLC (http://www.virtualreference.net/virtual/03e.html).

More information regarding library outsourcing is available from the American Library Association's Web site available at http://www.ala.org/alaorg/oif/outsourcing.html.

IX. BUDGET CONTROL AND ASSESSMENT

A. Budgeting and Control

Information systems provide effective, improved, and faster access to vast information resources. Of course, there is a price. In addition to network, hardware, and software costs, a larger percentage of total library budgets now goes towards expenditures for systems and e-content. As reported by Sewell and the Association of Research Libraries (ARL), 27 large research libraries each expended on average $1,163,100 in 1999/2000 on electronic resources (13.5% of each library's total materials budget). Each library spent an annual average of over $126,000 on bibliographic utility costs alone. Barry reports ILS vendors' total revenues for 2000 as $440 million from library markets.

It is also true that library IS can provide better management information, expenditure control, and resource assessment. Library IS provide faster, more detailed, and cost-effective means of collecting and analyzing information needed for management decisions. Management information is improved with IS by providing current data and report analysis of library, collection, network, and instructional usage and services. Budgeting control is vastly improved with IS, providing easily monitored purchase accounts, secure electronic data interchange (EDI), and programmed reporting of accounts and expenditures by various categories, order status, funds encumbered, and purchases made. System interoperability with var-

ious accounting, budgeting, and ordering systems are important factors to consider when acquiring an ILS. This allows for better management of budgetary resources and integration with larger scale institutional accounting systems.

B. Assessment

A good integrated library information system is essential to assessing collections and use. Systems provide report-writing capabilities that allow individual libraries and library managers to assess collections by subject, budget account, date, price, selector, and title.

The amount of use particular sources, databases, or collections receive is important in reviewing resources and making purchase decisions. Library organizations such as the International Coalition of Library Consortia (ICOLC) and the ARL have guidelines for measuring electronic source use. Table II displays suggested measures.

Additionally, the NISO standard on *Library Statistics* is now in its 5-year review cycle (ANSI/NISO Standard Z39.7-1995 available at http://www.niso.org/standards). These standards specify data categories collected for academic, public, school, and special libraries including information regarding unit and target population, personnel, collections, facilities, finances, and service and activity measures.

X. LEGAL, ETHICAL, AND SOCIAL ISSUES

E-content and database licensing, digital copyright, universal access, and online protection and privacy will be ongoing concerns for the foreseeable future.

Table II Suggested Measures for Assessing Electronic Resource Use[a]

Suggested measures and statistics for research library networked services	ICOLC guidelines for statistical measures of usage of Web-based indexed, abstracted, and full text resources
Patron accessible electronic resources Number of electronic full text journals Number of electronic reference sources Number of electronic books	Statistical measurement elements Queries (searches) Menu selections Sessions or logins Turnaways due to simultaneous user limits Items examined User, institutional, and consortium confidentiality Comparative statistics Secure access to statistical reports via the WWW
Use of networked resources and services Number of electronic reference transactions Number of logins (sessions) to electronic databases Number of queries (searches) in electronic databases Items requested in electronic databases Virtual visits to library's Web site and catalog	Vendor data provided For each specific database of the provider For each institutionally-defined set of IP addresses/ locators to subnet level By total consortium By special data element passed by subscriber (e.g., account or ID number) By time period, Vendor's system should minimally report by month. For each month, each type of use should be reported by hour of the day, and vendor should maintain 24 months of historical data.
Expenditures for networked resources and related infrastructure Cost of electronic full text journals Cost of electronic reference sources Cost of electronic books Library expenditures for bibliographic utilities, networks, and consortia External expenditures for bibliographic utilities, networks, and consortia	
Library digitization activities Size of library digital collection Use of library digital collection Cost of digital collection construction and management	
Performance measures Percentage of electronic reference transactions of total reference Percentage of virtual library visits of all library visits Percentage of electronic books to all monographs	

[a]International Coalition of Library Consortia. November 1998. "Guidelines for Statistical Measures of Usage of Web-based Indexed, Abstracted, and Full Text Resources." Available at http://www.library.yale.edu/consortia/webstats.html [November 2001]; Association of Research Libraries. October 2001. "Measures and Statistics for Research Library Networked Services: Procedures and Issues (ARL E-Metrics Phase II Report)." p. 56. Available at http://www.arl.org/stats/newmeas/emetrics/ [November 2001].

A brief outline of the issues and pertinent legislation follow.

A. Licensing and Contracts

Expectations and standards for e-content and database licensing are fairly standard compared to several years ago. Davis (in Ensor, pp. 368–372) discusses concerns that should be addressed in IS contracts and license agreements including:

- Parties in the agreement
- Ownership rights
- Lease rights
- Users
- Access
- Databases
- Search results
- Software
- Product maintenance
- Access control
- Hidden costs and penalties
- Contract terms and termination
- Warranty, liability, and indemnity clauses.

The Council on Library and Information Resources sponsors a web site, LIBLICENSE, providing model contract language and information about licensing electronic resources (http://www.library.yale.edu/~llicense/index.shtml). The ICOLC has developed "Guidelines for Technical Issues in Request for Proposal (RFP) Requirements and Contract Negotiations" (http://www.library.yale.edu/consortia/techissuespr.html). These guidelines are particularly useful in reviewing the technical requirements, software, hardware, and network configurations of an electronic database or system.

B. Copyright

Copyright laws protect content creators, publishers, producers, and distributors' rights to information and payment for content use. New laws have been created to address digital copyright. The 1998 U. S. Digital Millennium Copyright Act (DMCA), copyright laws of the U. S. and other countries, and world copyright treaties (World Intellectual Property Organization at http://wipo.org/) are relevant sources of information regarding copyright restrictions.

Information systems that control and facilitate user access to copyrighted material, track usage, and provide payment and licensing authority are called digital rights management or copyright management technologies. These systems are often integrated into publishers' or sources' web sites. The Copyright Clearance Center and its web site at http://www.copyright.com/ provide information and services regarding copyright payment.

C. Universal Access

Library legislation provides for networked universal access and service to libraries and library users, regardless of geographical location. The U. S. Telecommunications Act of 1996 (Pub. L 104-104) set up the Universal Service Program for Schools and Libraries providing discounted rates (E-rate). The Library Services and Technology Act (LSTA) established library telecommunication connections and services. *Access to* versus *control of* licensed databases, software, and online information continues to be debated by way of the Uniform Computer Information Transactions Act (UCITA) now being proposed in various states.

D. Privacy and Confidentiality

Privacy and user confidentiality are ongoing issues for libraries due to the confidential nature of library users' records and the importance of preserving intellectual freedom. New information systems provide more sophisticated ways of collecting data. However, the sole purpose for collecting data is to better manage libraries and to assess and improve collections, resources, and services. It is not intended for monitoring an individual's use of information. Safeguards and attention to system security are vital issues. The American Library Association (ALA) provides a Web site on "Privacy Resources for Librarians, Library Users, and Families" available at http://www.ala.org/alaorg/oif/privacyresources.html.

E. Child Safety, Censorship, and Filtering

Internet access and user protection is a controversial issue among libraries, families, and schools. Recent United States statutes include the Children's Internet Protection Act (CIPA, 2001), Neighborhood Children's Internet Protection Act (NCIPA, 2001), and Children's Online Privacy Protection Act (COPPA, 2000). The CIPA and NCIPA make selected federal funding for some libraries dependent on usage of "technology protection measures" such as filtering

programs. Thus, the issues of censorship and intellectual freedom have been raised.

The issues of filtering, intellectual freedom, censorship, and child safety raise questions of who decides what is filtered, what criteria are used, and the effectiveness of filtering programs. Filtering and controlling Web/Internet access and child safety are primarily issues for public and school libraries and are handled differently at various libraries. However, all libraries should have a written Internet access policy and guidelines. Shuler (in Ensor, pp. 202–203) provides an overview of options for controlling web access in libraries including (1) do nothing, (2) restrict minors' use of unfiltered access, (3) provide filtered access in areas where minors' collections are located, and/or (4) use filtering throughout the library and ask adults to request exceptions.

Various filtering/control technologies include:

- Filtering by Web site location or by subject using a separate filtering/blocking program
- Using an Internet service provider that offers filtered ISP service(s)
- Using URL rating or labeling technology standards as devised by the Platform for Internet Content Selection (PICS) (these technical standards allow a Web creator or a third party to rate Web site content)

Various viewpoints and information on intellectual freedom, filtering and blocking systems, rating programs, and child Internet safety are available from the following sources:

- Maxwell, N. K. (March/April 2001). Alternatives to filters. *Library Technology Reports,* Vol. 37, No. 2, 1–58.
- American Library Association's (ALA) "Filters and Filtering" available at http://www.ala.org/alaorg/oif/filtersandfiltering.html
- Family Friendly Libraries Home Page available at http://www.fflibraries.org/
- The Internet Content Rating Association available at http://www.rsac.org/
- Internet Education Foundation's "GetNetWise" available at http://www.getnetwise.org/tools/
- The National Coalition for the Protection of Children and Families' "FilterReview.com" available at http://www.filterreview.com/

XI. RESEARCH, INITIATIVES, AND FUNDING

There are several organizations involved in digital library and IS program research, development, production, and funding. Four organizations include The American Society for Information Science and Technology (http://www.asis.org/); The Berkeley Digital Library SunSITE, a cooperative effort by the Library of the University of California, Berkeley, and Sun Microsystems, Inc. (http://sunsite.berkeley.edu/); The Digital Libraries Initiative of the National Science Foundation in cooperation with other federal agencies (http://www.dli2.nsf.gov/); and The Text Retrieval Conference (TREC) which encourages research in information retrieval from large text collections (http://trec.nist.gov/).

XII. THE FUTURE

New developments are part of all information system landscapes, including library IS applications. The Library & Information Technology Association (LITA), a division of the American Library Association, maintains an excellent Web site titled "Top Tech Trends" (http://www.lita.org/committe/toptech/mainpage.htm). The following trends and future issues are gleaned from the LITA Web site, new and evolving IS applications, and areas requiring additional refinement. Foreseeable trends involve four major areas: telecommunications and hardware, information organization and products, access and standards, and text/information retrieval and extraction.

A. Telecommunications, Wireless Technologies, and Hardware

Use of *advanced telecommunication systems* will increase. This includes new generation wireless and radiofrequency technologies, Voice over Internet Protocol, and uses for hardware devices such as personal digital assistants (PDAs). Health and medical science libraries have already adapted and integrated PDA technologies with selected library services.

B. Information Organization and Products

Information syndication, new product packages, and development of vertical portals (called *vortals*) seem to be increasing. Syndicated e-content means that the same content is packaged and sold through multiple portals. Vortals generally combine e-commerce, software applications, trade news, and content. All are delivered to the user for a fee. Academic market vendors are modeling some products in this manner by

providing customized research portals. One example is *Ovid On Call* from Ovid Technologies, providing portal access to medical and health resources. The current use of customization and personalization of web/database home pages may also increase. Libraries and users may find selecting often-used resources, links, and databases to be an improved way of approaching research. Of course, as one-stop searching and automatic Internet site categorization improves, this may be less necessary.

Internationalization, multilanguage access, and translation functions of library information systems will be more viable through international alliances, programming, and standards that facilitate document translation.

E-books are still in an early development stage. For e-books to be a viable, mass-market, and ongoing information technology, e-book producers will have to develop and adopt better standardization and features that make reading online suitable beyond brief reference use.

C. Access and Standards

Interoperability and interconnectedness between information sources will increase and improve with use of OpenURL, the Open Archives Initiative (OAI), and XML. Several major libraries (National Library of Medicine and the NASA Astrophysics Data System) have selected the XML format (Miller). XMLMARC is under development and XML is being integrated into ILS and commercial information databases. Table III provides definitions and additional information regarding selected and evolving specifications.

The Open Archives Initiative (http://www.openarchives.org/) is developing and experimenting with OAI metadata, harvesting protocol, and interoperability standards to "open up" the Web. In practical terms, the goal of OAI is to make more of the "invisible or hidden Web" more accessible. The invisible or hidden Web includes academic/research information and other database resources which are often inaccessible from Internet search engines. "One-stop searching" or at least a categorized Web that offers full search capabilities seems to be an obvious direction.

D. Text and Information Retrieval and Extraction

A definite trend in information retrieval and library applications is one of *convergence*. This is already apparent and likely to continue. Metadata, source documents, and related materials such as reviews, cited sources, and other referring materials and Web sites

Table III Selected Specifications and Definitions[a]

SGML—Standard Generalized Markup Language (1986) serves as the basis of more widely used structuring and formatting document standards including HTML, DHTML, XHTML, and XML.
"XML [eXtensible Markup Language] is a metalanguage, or a set of rules governing the development of unique tags for encoding documents. Instead of using the pre-existing tagset available in HTML, users can design their own tags, which define the content, syntax, and semantics of their data." (Exner and Turner, pp. 51–52). XML is a subset of SGML which incorporates the core design features and the more useful aspects of SGML; allows tagging of documents or objects with any number of tags and specifying these tags within the document type definition (DTD); provides better linking functionality; and is increasingly used in the publishing and electronic content industry.
RDF (Resource Description Framework) is being developed as "an infrastructure that enables the encoding, exchange, and reuse of structured [web] metadata. . . . RDF is a W3C proposed standard for defining the architecture necessary for supporting web metadata. RDF is an application of XML that imposes needed structural constraints to provide unambiguous methods of expressing semantics for the consistent encoding, exchange, and machine processing of metadata." (Miller, E.)
OpenURL is a metadata transport standard for providing descriptive data about electronic resources to an information service using various vendors or proprietary programs. SFX is an example of a citation linking system based on OpenURL. SFX allows customizable searching and access by the library, "dynamically creat[ing context-sensitive] links that fully integrate their information resources regardless of who hosts them—the library itself or external information providers." (http://www.sfxit.com/).

[a]Sources: Author; Reviewer; Robert Alexander and Glenn Tarbox, 11 January 2002 [e-mail Correspondence Regarding Table], Personal e-mail (11 January 2002); Dick R. Miller, Summer 2000, "XML: Libraries' Strategic Opportunity," *Library Journal netConnect,* available at http://libraryjournal.com/xml.asp [December 2, 2001]; SFX, "SFX Context-Sensitive Reference Linking: SFX for Librarians," available at http://www.sfxit.com/, [25 January 2002]; Miller, Eric. May 1998. "An Introduction to the Resource Description Framework." *D-Lib Magazine,* available at http://www.dlib.org/dlib/may98/miller/05miller.html [January 23, 2002].

will be converging, consolidated, and linked. Information retrieval and information extraction systems will be combined to provide more powerful search systems in libraries. Database and ILS producers are beginning to enhance search retrieval systems by incorporating functions such as fuzzy search logic, relevancy searching, intelligent agents, and natural language processing. There are a number of reasons for these trends including:

- Increased digital full text/image sources, large text database collections, and less structured databases requiring additional means of retrieving textual information
- More hybrid e-content systems that include academic, research, news, and industry information (These databases and organizations combine different search systems and sources, resulting in synergistic search systems, improving or at least adding to information retrieval products.)
- The likelihood that there will be continued interest in and development of automatic indexing and classification, text summarization and machine generated abstracting for both structured and unstructured information sources. Such systems as *Clairvoyance Technology* (http://www.clarit.com/technology.htm) and *UniFind Information Retrieval, Management, and Delivery System* from KCSL Inc. (IRMDS at http://unifind.com/) have been utilized for these purposes. Natural language processing, clustering, fuzzy search techniques, and text mining will be areas to monitor.

Finally, the *Semantic Web* will most likely play a major part in the next generation of Web and information retrieval. The Semantic Web is the idea of Tim Berners-Lee (inventor of the World Wide Web). The basic vision is to make data on the Web understandable to computers through the use of intelligent software agents. For library and information retrieval systems, this will translate into more "elaborate, precise automated searches" according to Berners-Lee, Hendler and Lassila. The technical elements needed for the Semantic Web are XML, RDF, and a good system of subject ontology—categories, structures, and relationships of knowledge or systems (http://semanticweb.org/). While some of the basic building elements already exist, the Semantic Web is still under development. It will be a topic at many future professional and technical conferences.

SEE ALSO THE FOLLOWING ARTICLES

Copyright Laws • Ethical Issues • Law Firms • Privacy • Psychology • Research • Sociology

BIBLIOGRAPHY

American Library Association, Reference & User Services Association's (RUSA) Machine-Assisted Reference Section. (2001). Expert systems—Innovative Web-based reference services: A selected list. Available at http://www.ala.org/rusa/mars/expertsys.html.

Barber, D. (May/June 2001). Networking checklists for library managers. *Library Technology Reports,* Vol. 37, No. 3, 1–63.

Barry, J. (April 1, 2001). Automated system marketplace 2001: Closing in on content. *Library Journal.* Available at http://libraryjournal.reviewsnews.com/.

Berners-Lee, T., Hendler, J., and Lassila, O. (May 2001). The Semantic web. *Scientific American,* Vol. 284, No. 5, 35–43. Available at http://www.scientificamerican.com/2001/0501issue/0501berners-lee.html.

Bluh, P. M. (Ed.) (2001). Managing electronic serials, in *ALCTS Papers on Library Technical Services & Collections,* No. 9. Chicago: American Library Assoc.

Breeding, M. (September 2001). The ALA Annual Conference 2001: Here are the latest in library automation trends and announcements. *Information Today,* Vol. 18, No. 8. Available at http://infotoday.mondosearch.com.

Breeding, M. (June 1, 2001). The state of the art in systems. *Information Today,* Vol. 18, No. 6, 46–47. Available at Dow Jones News Retrieval.

Ensor, P. (Ed.) (2000). *The cybrarian's manual 2,* revised ed. Chicago: Amer. Library Assoc.

Exner, N., and Turner, L. F. (November/December 1998). Examining XML: New concepts and possibilities in Web authoring. *Computers in Libraries,* Vol. 18, No. 10, 50–53.

Furrie, B. (2000). *Understanding MARC bibliographic machine-readable cataloging,* 5th ed. Washington, DC: Library of Congress, Cataloging Distribution Service. Available at http://lcweb.loc.gov/marc/umb/.

Garlock, K. L., and Piontek, S. (1999). *Designing Web interfaces to library services and resources.* Chicago: Amer. Library Assoc.

Kenney, A. R., and Rieger, O. Y. (Eds.) (2000). *Moving theory into practice: Digital imaging for libraries and archives.* Mountain View, CA: Research Libraries Group. Tutorial available at http://www.library.cornell.edu/preservation/tutorial/index.html.

Lancaster, F. W., and Warner, A. (2001). Intelligent technologies in library and information service applications, in *American Society for Information Science and Technology ASIST Monograph Series.* Medford, NJ: Information Today, Inc.

Library & Information Technology Association. (2001). *LITA.* Chicago: Amer. Library Assoc. Available at http://www.lita.org/.

Lynch, C. A. (Winter 1997). Building the infrastructure of resource sharing: Union catalogs, distributed search, and cross-database linkage. *Library Trends,* Vol. 45, No. 3, 448–461.

Mates, B. T. (2000). *Adaptive technology for the Internet: Making electronic resources accessible to all.* Chicago: Amer. Library Assoc.

Mayo, D., and Nelson, S. (for the Public Library Association). (1999). *Wired for the future: Developing your library technology plan.* Chicago: Amer. Library Assoc.

Meadow, C. T., Boyce, B. R., and Kraft, D. H. (2000). *Text information retrieval systems.* San Diego: Academic Press.

Miller, D. R. (Summer 2000). XML: Libraries' strategic opportunity. *Library Journal netConnect.* Available at http://libraryjournal.com/xml.asp.

Needleman, M. (2000). Z39.50—A review, analysis and some thoughts on the future. *Library Hi Tech,* Vol. 18, No. 2, 158–165.

Publishers International Linking Association. (2000). CrossRef. Available at http://www.crossref.org/home.htm).

Rothenberg, J. (1999). *Avoiding technological quicksand: Finding a viable technical foundation for digital preservation.* Amsterdam: European Commission on Preservation and Access; Washington, DC: Council on Library and Information Resources. Available at http://www.clir.org/pubs/abstract/pub77.html.

Schmidt, K. A. (1999). *Understanding the business of library acquisitions.* 2nd ed. Chicago: Amer. Library Assoc.

Schultheiss, L. A., Culbertson, D. S., and Heiliger, E. M. (1962). *Advanced data processing in the university library.* New York: Scarecrow Press.

Sewell, R. G. (2001). *Library materials budget survey 2000/2001.* Washington, DC: Association of Research Libraries. Available at http://arl.org/scomm/lmbs/lmbs2001.html.

Tennant, R. (March 15, 2001). XML: The digital library hammer. *Library Journal Digital.* Available at http://libraryjournal.reviewsnews.com/.

Wilson, T. C. (1998). *The systems librarian: Designing roles, defining skills.* Chicago: Amer. Library Assoc.

Linux

Bryan Pfaffenberger
University of Virginia

GLOSSARY

BSD Acronym for Berkeley Software Distribution. A leading variant of UNIX that was created, initially, by faculty and students in the Department of Computer Science at the University of California, Berkeley. The influential BSD distribution was the first to pair UNIX and TCP/IP networking. Open source versions of BSD, including FreeBSD, NetBSD, and OpenBSD, are often preferred over Linux for high-load, high-demand servers.

daemon A program, often a server, that runs as a background process, ready to spring into action when needed. An example of a server daemon is the Apache Web server.

distribution A smoothly integrated software package in which a compiled Linux kernel is matched with file utilities, a windowing environment, desktop environments, networking services, server daemons, and application software.

embedded operating system A minimally sized version of an operating system that is designed to perform specialized functions within a specialized hardware device, such as a Web-capable cellular telephone.

Free Software Foundation A nonprofit organization based in Cambridge, Massachusetts, that funds the GNU Project and related activities.

General Public License (GPL) A software license that grants the user the right to access the program's source code, to create derivative works, and to redistribute the original or derivative versions of the program, as long as these same rights are not denied to recipients of the redistributed software.

GNU/Linux A term used by some to refer to a complete Linux-based operating system, in preference to "Linux." Proponents justify this term by pointing to the many GNU-derived utilities and the programming environment that is included with Linux distributions. Critics point out that much of the accompanying software was not created by the GNU Project.

GNU Project A software development project, initiated in 1984, that seeks to create a freely redistributable alternative (called GNU, an acronym for Gnu's Not UNIX) to commercial versions of UNIX.

kernel The core of the operating system that is responsible for managing processes, synchronizing events, interacting with the user, and communicating with the computer's various devices,

Linux Properly speaking, the term "Linux" refers to the UNIX-like kernel initially developed (and still managed) by Linus Torvalds. The term is also used to refer collectively to all the software needed to transform the kernel into a complete, working operating system; however, this usage is inaccurate.

loadable kernel module In an operating system that employs a monolithic kernel, a means of reducing kernel size by moving seldom-used functions to disk-based modules, which are automatically loaded into memory as needed.

microkernel A kernel that is kept as small as possible; additional components are loaded on the fly by means of synchronous management processes.

Minix An early UNIX-like operating system for Intel-based hardware, created in 1987, that was developed for teaching purposes.

monolithic kernel A kernel that embodies all the necessary operating system code into a single, large program. Another design option is the *microkernel* architecture.

open source software (OSS) Software that is distributed in such a way that users can access, modify, and redistribute the program's source code. This term encompasses software distributed with a variety of licenses, including the BSD and GPL licenses, which vary significantly in the expectations they place on software recipients.

UNIX A trademark currently owned by the Open Group, Inc., which is licensed to UNIX-like operating systems only after payment of a fee and tests that demonstrate conformance to the Open Group's specifications for UNIX compatibility. Linux would likely not pass some relatively obscure portions of the test; it is compatible, but not strictly compatible, with licensed versions of UNIX.

UNIX-like operating system An operating system that closely resembles, and is largely compatible with, officially licensed versions of UNIX.

window manager In the X Window System, a special type of client program that enables the user to control and customize the windowing environment.

windowing environment An operating system layer that provides graphical windowing support for applications.

XFree86 A freely redistributabler version of the X Window System that is typically distributed with Linux.

X Window System (X) A graphical windowing environment for UNIX and UNIX-like operating systems. The version of X distributed with Linux is called XFree86.

I. INTRODUCTION

Linux is the *kernel* (operational core) of an interactive, multiuser, and UNIX-like operating system, also (and somewhat confusingly) called Linux, that is rapidly gaining market share as an alternative to commercial server operating systems (such as Microsoft Windows NT/2000/XP). The Linux kernel stems from an international, Internet-mediated development project, launched in 1991 by a Finnish computer science graduate student named Linus Torvalds; the project bore fruit 3 years later in the form of Release 1.0 (March 14, 1994). The Linux kernel project was part of a broader push to develop a "clone" of the commercial UNIX operating system that would run on inexpensive Intel-based hardware and help users avoid the licensing and copyright issues associated with UNIX code.

Since the release of the first Linux kernel, Linux has assumed its place in history as one of the most successful experiments in *open source software* (OSS), which is collaboratively developed and released under licensing schemes that enable recipients to modify and redistribute the software freely. At this writing, Linux *distributions* (smoothly integrated packages that include the Linux kernel and needed utilities) have captured about one-quarter of the server operating market (chiefly in the low-end server market), posing stiff competition for Microsoft's NT-based server operating systems as well as older, commercial UNIX systems; thanks in part to another highly successful OSS project, the Apache Web server, Linux is expected to become the dominant Web server platform worldwide. Linux is also making inroads in the emerging market for *embedded operating systems,* which are special-purpose operating systems contained within a consumer device, such as a Web-enabled cellular telephone.

Linux is well on its way to becoming one of the most widely used operating systems in the world. Originally designed to run on Intel-based PC hardware, Linux has since been ported to most of the hardware environments that are in common use worldwide, including mainframes. Once developed almost exclusively by volunteers working on their own time, Linux is now benefitting from massive corporate investments from hardware vendors such as Sun Microsystems and IBM; as a result, hundreds of full-time, expert programmers are able to devote themselves to Linux development. In spite of its gains and rosy future, however, some question remains whether Linux will be able to penetrate the high-end, high-load server market (currently dominated by commercial versions of UNIX) or the desktop OS market (currently dominated by Microsoft Windows).

II. HISTORY OF LINUX

Linux is a UNIX-inspired operating system that brings together two strands of software development, including the kernel development effort spearheaded by Torvalds and the suite of cloned UNIX utilities and program development tools developed by the GNU Project, based in Cambridge, Massachusetts, which seeks to develop a freely redistributable version of UNIX. By 1993, Torvalds had developed a UNIX-like

kernel that was winning increasing respect, but a complete operating system requires additional utilities and tools. At the same time, the GNU Project had not yet developed a functioning kernel. The combination of the Linux kernel with GNU and other tools and utilities resulted in a complete, usable operating system. For this reason, some prefer to differentiate the kernel from the operating system by calling the latter GNU/Linux, but others disagree, pointing out that some commonly used Linux OS components derive from sources other than the GNU Project.

A. Origins, 1991–1994

Linux was not the first UNIX-like operating system developed for Intel-based hardware. Among earlier efforts was an Intel 386-based port of BSD UNIX, described in a series of articles in *Dr. Dobb's Journal* beginning in January 1991. Linux's immediate ancestor was Minix, an Intel-based operating system which was developed and initially released in 1987 by Professor Andrew S. Tanenbaum, a computer science professor at Vrije Universiteit, in Amsterdam, The Netherlands. Designed to run on Intel hardware, Minix was conceived as an instructional aid in operating system design courses; it was distributed with Tanenbaum's textbook, which made thousands of lines of operating system source code available for study. Tanenbaum did not want the code to grow to the point that students could not understand it. For this reason, he refused the many requests he received to add features to Minix.

Frustrated with Minix's lack of advanced operating systems features, Linux Torvalds, a Finnish university student studying for a master's degree in computer science, decided to create a new UNIX-like kernel by using the Minix code as a guide. In 1991, Torvalds announced the project on the Minix newsgroup. Torvalds was aided by dozens of top-notch programmers, who helped Torvalds add new features and debug the evolving code. Among the developers were a few trusted "lieutenants," who were (and are still) responsible for major chunks of the code (such as TCP/IP networking or device drivers). However, Torvalds reserved the final right to accept or reject contributions to the "official" version of the code. Version 1.0 of the Linux kernel was released in March 1994. Although the modest Torvalds initially called the kernel "Freax" (i.e., free UNIX), his friends prevailed on him to accept the name Linux instead.

A well-publicized spat between Torvalds and Tanenbaum served to highlight the differences between Minix and Linux. Tanenbaum was a staunch advocate of *microkernel* architectures for operating systems; in such an architecture, the operating system software is broken up into smaller, more easily managed units whose actions are tightly synchronized by timing and communication routines. But Torvalds' kernel was *monolithic,* meaning that all of the code is kept in one, large program. This design is efficient, in that all program routines are kept together in memory and can therefore access each other speedily, but adding too many features can cause a monolithic kernel to become sluggish and unwieldy. Tanenbaum branded Torvalds' kernel "obsolete" and admonished the young programmer that he would have received a very low grade in Tanenbaum's course. Torvalds retorted that Minix's many flaws are forgivable when its instructional, ivory-tower goals are kept in mind; Linux, Torvalds said, was designed for the real world.

Although operating systems experts are still divided on the monolithic kernel vs microkernel issue, experience has shown that the communications overhead required to implement a microkernel imposes a costly performance penalty. With respect to Linux, the debate has been rendered moot, for most, by the development of *loadable kernel modules* in version 1.2 of the Linux kernel. These modules provide functions such as peripheral device support. They are dynamically loaded only when they are needed. With loadable kernel modules, new features can be added without significant increases to the kernel's overall size.

B. The GNU Project

To understand the success of Linux fully, it is important to realize that a kernel, by itself, can accomplish little. Torvalds' achievement, impressive as it was, might have accounted for little had it not been for the GNU Project's efforts to create freely redistributable versions of the standard UNIX utilities and programming tools. GNU, a recursive acronym that stands for "GNU's Not UNIX," was conceived by the GNU Project's founder, Richard Stallman, as an alternative to commercial, closed-source software. Although the GNU Project had not yet developed a kernel by the early 1990s, its achivements were considerable, including the Bourne Again Shell (bash, a command-line interpreter), the Emacs text editor, gawk (a GNU implementation of the awk text processing language), GCC (a compiler for C, C++, Objective C, and Fortran), and a host of file and text utilities. Torvalds deliberately designed the Linux kernel so that it would work smoothly with the GNU utilities in general and with the GCC compiler in particular.

C. The General Public License (GPL)

The GNU Project contributed more than a suite of tools and utilities; it also contributed its unique free software philosophy and its licensing mechanism, called the *General Public License* (GPL). The GPL is one of a family of software licenses that enable users to access the source code and to distribute the software to others. The GPL grants the software recipient a number of rights, including the right to distribute the software freely, to obtain and read the source code, and to create and distribute derivative versions of the software. In contrast to most other free software licenses, however, the GPL is specifically designed to prevent anyone from incorporating GPL-licensed code into closed-source products. The GPL does so by insisting that any redistributions of the code, including redistribution of derivative works, must be done under the terms of the GPL; in other words, the license says, in effect, "You may distribute this software (including the source code) in any way you wish, as long as you refrain from denying to others the same rights you received."

The GPL has long inspired controversy. Pointing out that Stallman himself calls the GPL a form of "copyleft" (as opposed to copyright), some believe that the GPL encapsulates an anticorporate, anticapitalist mentality that is out of synch with the realities of the IT market. Still, one could plausibly argue that the GPL draws its inspiration from the American revolution, not the Bolshevik. The GNU Project's attempt to develop a growing body of freely accessible, open source code is in line with what leading legal scholars believe to be the philosophy regarding intellectual property that is expressed in the U.S. Constitution, which—even when the text is read in isolation—clearly specifies that the rights of intellectual property holders should be balanced against the legitimate right of the public to obtain and use information. In the 2 centuries since the U.S. Constitution was written, wealthy copyright holders have persuaded Congress to extend the length of copyright from the original 14 years (with the option of a 14-year extension) to the current maximum of 153 years; under current law, no more copyrighted material will enter the public domain for an entire generation. In short, the GPL is deliberately designed to provide a socially constructive alternative to the public domain. It is already responsible for a huge and growing body of publicly accessible computer code, which students of computer programming can inspect, modify, improve, and redistribute to others in a way that confers the same user rights.

Other GPL critics worry that corporations will be wary of adopting GPL-licensed code out of fear that their own, proprietary code will become so intermingled with GPL-licensed code that they will be forced to reveal their source code and distribute it freely, thus compromising trade secrets and making considerable programming investments available to competitors. But such concerns are misplaced. The GPL clearly states that a derivative program must be released under the GPL in its entirety if it contains GPL-licensed code; this stipulation does mean that a derivative program must be released under the GPL in its entirety if it incorporates GPL-licensed program libraries, which are portions of a program that are kept in separate files so that they can be loaded when needed. But the GPL does *not* require a program to be released under the GPL if it contains no GPL-licensed code. In addition, GPL licensing is not required for a *de novo* program that works with GPL-licensed software without actually incorporating GPL-licensed code in a substantial way; for example, a *de novo* program that is designed to work with GPL programs, but does not actually incorporate their code, does not require GPL licensing. For this reason, a program that uses standard Linux system calls and application programming interface (API) functions, and does so without incorporating GPL-licensed code into its own code, will not require GPL licensing. GPL licensing is not required for a *de novo* program that is written using GPL-licensed text editors and compiled using GPL-licensed programming tools. Nor is it a violation of the GPL to install and run proprietary, closed-source code on a GPL-licensed operating system. It is not a violation of the GPL to create derivative programs containing source code that is not divulged to the public, as long as such programs are used in-house and not redistributed to third parties. In short, the GPL is admirably clear. It states that you must release derivative software under the GPL if (1) the derivative software contains GPL-licensed code and (2) the derivative code is released to third parties.

At this writing, the GPL has not been tested in a court of law. Some legal experts believe that its licensing provisions are unenforceable. Others counter that GPL software is protected in the first instance by copyright, so that redistributions without a license constitute copyright infringement; if someone redistributes GPL-licensed code, they are violating copyright unless they do so in accordance with the GPL's provisions, which are included with every copy of GPL-licensed code. The GPL's legal critics respond that actions against copyright infringement are not likely to succeed unless the copyright holder can prove actual financial losses, which is not likely because virtually all GPL-licensed software is available free of charge; however, recent changes to copyright law, as well as key court decisions, have es-

tablished civil and even criminal penalties for cases of infringement in which copyright holders are unable to prove financial losses resulting from the infringement. These changes have been driven, in no small measure, by the successful rent-seeking of wealthy copyright holders (such as motion picture studios and recording industry giants); ironically, by strengthening copyright beyond all previous bounds, they may have inadvertently strengthened the GPL and, with it, a meaningful alternative to a shrinking public domain.

D. Server Daemons and XFree86

Another factor in Linux's success is the availability of several high-quality but freely redistributable server *daemons,* programs that run in the background until their services are needed. Chief among these is the Apache Web server. Apache is derived from the public domain Web server code created by Rob McCool at the U.S. National Center for Supercomputing Applications (NCSA); the first release occurred in 1995. Also crucial to Linux's success were wu-ftpd (a freely redistributable FTP server originally developed at Washington University in 1994), as well as Linux ports of freely redistributable or public domain UNIX daemons, such as sendmail (for electronic mail services).

Yet another key element in Linux's success was the availability of a graphical user interface, called XFree86, that enabled Linux to run thousands of UNIX progams developed for the X Window System. The XFree86 project began in 1992 with an attempt to create an Intel version of Release 5 of X Window System version 11. As Linux developed, the XFree86 developed in tandem, with the two projects helping to stimulate the other. Since then, the XFree86 project has unhinged itself from its Intel-specific origins (which are implied by the "86" in XFree86), and is now rightly regarded as having taken a leadership role in X Window System development.

E. The Rise of Linux Distributions

By 1993, all of the core elements of a GPL-licensed UNIX-like operating system were in place—the kernel, the GNU utilities, and the server software—but Linux found favor only among computer hackers, programmers, students, and computer science faculty. The reason lies in the inordinate complexity involved in modifying all of these components so that they function together smoothly.

Linux might have remained a curiosity had it not been for the rise of *Linux distributions,* which are complete, smoothly integrated packages that capture the experts' system integration achievements. Linux distributions are typically disseminated by means of a CD-ROM or via Internet FTP connections. A distribution typically includes a compiled and configured kernel, an installation utility, documentation, online and telephone support, a range of applications (including demos of commercial applications), and a packaging mechanism (software for installing binary versions of programs easily). The first such distribution issued from the computer center of Manchester University in 1992; following shortly was the Softlanding Linux System (SLS), the first distribution to incorporate the X Window System. One of the oldest Linux distributions that is still maintained, Slackware, is based on a 1992 variation on the SLS theme, as was SuSE Linux, founded in late 1992. Today's leading Linux distribution, a commercial venture called Red Hat, was founded in 1993. In part to counter the growing trend toward commercial Linux distributions, a group of volunteers founded Debian GNU/Linux in 1993; this respected distribution remains under active development and is still staffed entirely by volunteers.

Companies that distribute Linux commercially can charge money for their integration services and for any proprietary software that they include in the distribution; nothing in the GPL's language forbids programmers or companies from charging money for their services or products. However, these companies must pass along the GPL-mandated rights to their customers. How have companies been able to make money on a product that, once purchased, can be freely copied and given to others for free? The answer to this question is that the distribution's system integration services are worth something in themselves. Furthermore, a typical Linux distribution is enormous; the current Red Hat distribution requires two compact disks' worth of code. Even with a relatively fast Internet connection, downloading so much data would take many hours; under such circumstances, it is sometimes easier—and cheaper—to purchase a package that includes the CD-ROMs, technical support, and a well-written manual. Companies also hope to make money providing technical support and on-site system integration.

III. TECHNICAL ORGANIZATION OF LINUX

Linux was initially designed to function on standalone, desktop PCs; nevertheless, like other variants of the UNIX operating system, it is a full-fledged, interactive operating system that is capable of supporting many users with apparent simultaneity, so that each user has the impression of having sole control of

the system. Indeed, Linux more closely resembles a powerful server operating system than a consumer-level desktop operating system. This fact handily explains Linux's meteoric success in the low-end server market. As organizations increasingly connected desktop computer systems to local area networks (LANs), Linux users were able to set up low-cost Web, FTP, and e-mail servers by taking advantage of Linux's built-in features (such as TCP/IP networking) and the daemons commonly packaged with Linux distributions. Often, this was done without necessarily clearing the matter with IT administrators in large organizations; such administrators came to realize that, at the same time they were saying, "We do not use Linux and we don't intend to," there were several dozen Linux servers in operation right under their noses.

A. Characteristics of Interactive Multiuser Operating Systems

Like commercial versions of UNIX, Linux is able to perform all of the following advanced operating system functions:

- Run user processes efficiently so that users do not endure unacceptable delays, even when more then one person is using the system
- Manage memory efficiently so that frequently used blocks of code remain in the computer's main memory, where they are directly accessible to the processor, while less-frequently used blocks are swapped out to the computer's hard disk
- Provide an efficient, clear, and straightforward means for establishing user and group user identities and managing their files
- Manage input and output devices so that all available resources, including printers, network resources, disk drives, modems, and other peripherals, are easily available to users with the appropriate permissions
- Support a variety of interfaces, ranging from command-line interfaces to Microsoft Windows-like graphical user interfaces (GUIs), so that a variety of users can use the system comfortably
- Provide an efficient software development environment so that programmers will be encouraged to develop applications

Operating system design invariably involves a series of trade-offs. In particular, an operating system that excels in allocating CPU resources efficiently among hundreds or thousands of users may do a relatively poor job of providing a comfortable, easy-to-use environment for end users. Historically, UNIX has done a far better job in the server end of the client/server continuum, while users vastly prefer desktop-based systems running Microsoft Windows or Mac OS. At this writing, it is far from clear whether Linux, even equipped with an attractive graphical interface and Windows-like applications, can compete effectively in the desktop market; no matter how friendly the interface, there is a powerful, multiuser operating system beneath the surface, so that successful Linux use invariably requires more system and network administration expertise than most Windows or Macintosh users are likely to possess.

B. The Unix Philosophy

Because Linux is a Unix-like operating system, it conforms to the Unix design philosophy, as outlined by Mike Gancarz in *The Unix Philosophy*. In brief, this philosophy holds that an operating system should be composed of small, easily managed units, each of which should accomplish one task well. The interfaces between the various components of the operating system should be standardized, so that users can choose among alternative components at any level of the system. In addition, these interfaces should be hardware independent; in UNIX, this is done by conceptualizing and treating all system resources—including hardware devices—as if they were files. Finally, the design should adhere to the principle "worse is better"; that is, portability and utility should be chosen over efficiency or conformity to the latest ivory tower conceptions of operating system design.

C. Components of a Linux System

Like all operating systems derived from UNIX, Linux implements the Unix design philosophy by means of a layered organization, which can be conceptualized as a series of concentric circles; at the center of the system is the kernel. Additional components—specifically, the shell (command interpreter), X Window System, desktop environment, and user applications—surround the kernel, like the skins of an onion. Each succeeding layer requires the services of the previous one.

1. Kernel

The kernel is a single, monolithic executable that accomplishes two major functions:

- **Hardware and network abstraction.** By means of *abstraction,* the Linux kernel avoids the complexity involved in directly accessing hardware and network resources. There are two major components that provide abstraction services. The first, Virtual File System (VFS), provides a virtual interface to physical filesystems and hardware devices. Thanks to this interface, Linux can work with a wide variety of physical filesystems and devices. However, in order to do so, filesystem- and device-specific code must be present in the kernel while it is running. The second abstraction component, Abstract Network Services, provides a virtual interface to network services. Like other Unix-derived operating systems, the Linux kernel uses the BSD-derived system of abstract network interfaces, called sockets.
- **Resource management.** The kernel manages two types of system resources: memory and processes. The memory manager manages the virtual memory system, which enables the computer's hard disk to be used as an extension of physical memory. The *process manager* manages the time-slices and priorities of processes (executing programs).

Communication between the kernel and running applications is handled by *system calls,* which fall into three major classes: process control (starting, terminating, or dividing processes), file manipulation (creating, opening, reading, writing, closing, and deleting files), and device maintenance (requesting devices, reading and writing to devices, getting and setting attributes, and closing devices).

2. Shell

The second major component of Linux is the *shell,* which functions as a command interpreter. Although Linux enables users to select the shell of their choice, by far the most widely used shell is the GNU Project's Bourne Again Shell (bash). The shell runs in text mode; if X is running, it runs in a terminal window.

The bash shell is a fully programmable shell with many advanced features, including aliases, job control, filename completion, command history, shell functions, and much more. Although difficult to learn and use, it provides a powerful environment for executing text-mode programs and scripts.

3. Windowing Environment

A third major (but optional) component in a Linux system is the *windowing environment,* a software layer that provides windowing services for applications. When Linux is used primarily as a server operating system, users generally prefer to run Linux in its text-only mode, controlling the computer and applications by means of the shell; however, graphical applications require a windowing environment.

In Linux, windowing services are provided by *XFree86,* a freely redistributable (but not GPL-licensed) emulation of the *X Window System (X).* X was initially designed to provide cross-platform windowing support for graphics-based applications in a heterogeneous networked environment; it provides a consistent application programming interface (API) for graphical applications that relieves programmers of the need to concern themselves with the details of a particular system's display hardware. X's origins explain its design, which varies significantly from the display systems found in consumer operating systems. In consumer operating systems, the video hardware is directly accessed. In contrast, X is designed with a client/server architecture. Although this design imposes communication overhead between applications and the system's video hardware, it also introduces a degree of flexibility that is not easily achieved with consumer operating systems. For example, X enables network users to configure their interface to their liking, even if they are all simultaneously using a centrally administered application. This capability may prove advantageous as organizations modify their computer systems so that they meet mandated accessibility guidelines.

Confusing to many new X users is the system's usage of the terms *client* and *server.* In X nomenclature, the *X server* is housed on the local machine (rather than a remote system elsewhere on the network). A given X server is designed to take full advantage of the video hardware (including the video adapter and monitor) found on that system. *X* clients are X-compatible applications. By means of the X Window System's API, clients request windowing services from the X server. A special type of client, the *window manager,* gives the user tools for configuring the X environment. By means of the window manager, users can control the appearance, position, and size of application windows. There are several alternative window managers available; one of the attractions of X is the freedom it gives users to choose the window manager they prefer.

4. Desktop Environment

The X Window System was not designed to provide many of the features of consumer operating systems, such as drag-and-drop, antialiased onscreen fonts, and

an assortment of desktop utilities; instead, it leaves such features to applications. Unfortunately, X applications vary in the way they provide these services, forcing users in many cases to relearn basic skills as they move from one application to the next. The solution to this problem lies in a *desktop environment,* an additional operating system layer that provides graphical user interface consistency. In Linux systems, desktop environments run on top of the X Window System and take advantage of X's services. The leading desktop environments can run X applications (and text-mode applications), but they do not enable features such as drag-and-drop or fonts unless the user is running an application specifically designed for the current desktop environment.

Several desktop environments are available for Linux systems, including the two leading environments, the K Desktop Environment (KDE) and GNOME, both of which are licensed under the GPL. The two environments are somewhat compatible with one another, as long as both are installed on a given system; for example, KDE can run GNOME applications, and vice versa, but with some loss of functionality. Both are attractive, well-designed desktop environments with a growing suite of applications. Of the two, KDE is the older and, in consequence, the richer in applications, including KOffice, an office suite consisting of full-featured word processing, spreadsheet, and presentation graphics modules. GNOME offers a well-designed file manager (Nautilus), an Outlook Express-like e-mail and personal information management program (Evolution), and a growing pool of applications.

Although the use of Linux on desktop and notebook computers is steadily increasing, the pool of Linux-compatible applications lags far behind the comparable pools for Microsoft Windows and, to a lesser extent, Mac OS. Many would-be Linux enthusiasts find that, after attempting to use Linux as a replacement for Windows, they are unable to find an easy-to-use or feature-rich application for some specific task that they must perform. Such users typically abandon Linux for Windows, or configure their systems so that they can choose which operating system to run in a given session.

As the pool of Linux-compatible applications continues to grow, one may expect the number of Linux desktop users to rise; however, Linux system administration is still beyond the skill set of the typical desktop user. Efforts are underway to create Linux distributions specifically designed for end users; these distributions, analogous to the home user's version of Microsoft Windows XP, reduce Linux's complexity by removing difficult-to-configure services that most desktop users do not need.

IV. MAJOR FEATURES AND CAPABILITIES OF THE LINUX KERNEL

Because Linux derives its inspiration from UNIX, it resembles UNIX in its ability to provide powerful and useful tools for serving data in multiuser, networked environments, which accounts for much of its market penetration success to date. However, Linux is also the first viable UNIX variant available for inexpensive personal computers and, as such, has consistently faced user demand for desktop PC support, including support for USB and Firewire peripherals, high-end video cards, and high-speed hard disk interfaces. At the same time, developers are under pressure to improve Linux's ability to compete more effectively in the enterprise market, where the operating system's ability to withstand high loads is of paramount importance. As a result, Linux kernel development is being pushed in two contrasting directions, the one mandated by the need to support new PC hardware standards, and the other mandated by the need to deal with high-demand applications (and, in particular, to work with systems that employ multiple processors). The conflicting demand is partly responsible for lengthy delays in releasing new versions of the kernel. The following summarizes the kernel's features for both markets, interactive multiuser computing and end-user computing on the desktop.

A. Linux Features for Interactive Multiuser Computing

The version of the Linux kernel available at this writing (2.4) includes the following features commonly found in high-end, commercial UNIX operating systems that are intended for high-demand server applications in the enterprise:

- **Multitasking.** Linux takes full advantage of today's high-performance microprocessors by enabling two or more programs to execute simultaneously. With Linux version 2.4, Linux can work with as many programs as the system's memory can accommodate.
- **Multiuser.** Two or more users can use a Linux-equipped system simultaneously. With Linux version 2.4, system administrators can assign a theoretical maximum of 4.2 billion unique users and group IDs.
- **Symmetric multiprocessing.** Linux can work with systems equipped with two or more CPUs. Linux

2.4 is optimized to work with up to 8 CPUs, but can work with as many as 64.

- **Multithreading.** The Linux kernel supports multiple independent control threads within a single running program (called a *process*).
- **Dynamically loaded modules.** The Linux kernel is *monolithic*—that is, it is a single, large program in which all the components have access to the internal data structures and routines. The problem with monolithic kernels is that they grow much larger as additional device support and other functionality is added. One alternative is the *microkernel* architecture, in which the kernel is broken down into separate, functional programs linked by real-time communication mechanisms; but microkernel designs perform sluggishly. The Linux solution is to employ dynamically loadable program modules for functions that are needed only occasionally or not at all.
- **Protected memory.** Each process runs in its own, protected memory space, so that a crashed process does not bring down other applications (or the entire system).
- **Virtual memory.** The computer's hard disk can be used as an extension of random access memory (RAM) so that programs can be run without having to load the entire program into memory.
- **Support for massive amounts of RAM.** Enterprise users need copious amounts of RAM to run large databases. The Linux 2.4 kernel can work with up to 64GB of RAM.
- **Support for massive files.** Enterprise users also need to create very large database files. The Linux 2.4 kernel can work with files as large as 16 terabytes.
- **Logical Volume Management (LVM).** Like the leading commercial UNIX systems, Linux offers local volume management (LVM), in which a single, dynamically sizeable storage volume can be defined that spans more than one physical drive.
- **Journaling filesystem.** The Linux kernel includes a journaling filesystem. As opposed to the standard, static filesystem used with Linux (ext2), a journaling filesystem reduces the chance of data loss by recording all changes made to all files; this feature enables data to be reconstructed in the event of a system failure or power outage.
- **Shared program libraries.** As opposed to *static libraries,* which are included in a program when it is compiled, shared program libraries are loaded into memory only as needed, resulting in more efficient memory use. Like commercial UNIX systems, Linux uses the Executable and Linkable Format (ELF) system for shared program libraries. Thanks to Linux's virtual memory architecture, two or more programs can use a shared program library simultaneously.
- **X Window System.** Originally developed at Stanford University and MIT, the X Window System (or "X" as it is commonly called) is a cross-platform windowing environment for UNIX and UNIX-like operating systems. The implementation of X for Linux is called XFree86 (http://www.xfree86.org).
- **Program development support.** Linux provides a comprehensive platform for software development, including full-featured text editors (including emacs and vi), compilers for a variety of programming languages (including C, C++, Java, perl, and many more), debuggers (such as Electric Fence), project control utilities (such as make and automake), version control utilities (such as cvs), and a programmable shell. For its application programming interface (API), Linux incorporates much of the SVR4 UNIX and BSD standards.
- **Networking support.** Linux features advanced, built-in support for TCP/IP networking at the kernel level, and this support conforms to standard UNIX conventions. Support is also provided for many other networking protocols, including Sun's Network File System (NFS), X.25, AppleTalk, and IPX. When distributed with Samba, a package of utilities that implements Microsoft Windows networking services on Linux systems, Linux computers can join Windows peer-to-peer networks, log in to Windows NT/2000/XP domains, and in many cases emulate many of the services provided by Microsoft Windows NT/2000/XP servers.
- **Security.** Like commercial UNIX systems, Linux includes utilities and programs designed to ensure security during networking operations, such as firewalls, system logs, and software for intrusion detection and recovery.
- **Extensive, powerful server software.** Most Linux distributions come equipped with state-of-the-art Web, e-mail, newsgroup, and FTP servers. One of the most popular applications of Linux involves using the licensed Apache server to take over low-end Web server duties.
- **POSIX compatibility.** Although Linux does not claim strict POSIX compatibility due to licensing restrictions, Linux conforms (in general) to the Single UNIX Specification (a combination of the two prevailing POSIX standards), as defined by The Open Group, a UNIX standards organization.

At least two Linux distributions have independently sought and received POSIX.1 certification; however, the Linux kernel is not currently 100% compatible with certain obscure parts of the standard. Because Linux is largely POSIX-compatible, programmers can expect that their programs will be able to run on UNIX operating systems, although certain modifications may be necessary.

- **A 64-bit processor compatibility.** Versions of Linux are now available that support the 64-bit Intel Itanium and Alpha/AXP processors. Installed in an array of Itanium processors using symmetric multiprocessing, Linux is capable of creating a formidable computing platform that can execute more than 24 billion instructions per second.

Although Linux has made impressive strides and gained credibility for low-end server applications in the enterprise (such as running low- to medium-demand Web and e-mail servers), the operating system has not yet established credibility for high-end enterprise applications, such as ERP (enterprise resource planning), CRM (customer relationship management), and large-scale symmetric multiprocessing installations; however, this situation is expected to change as hardware and database vendors, including IBM, Oracle, and Sun Microsystems, invest heavily in Linux development so that they can provide an alternative to Microsoft's server operating systems.

B. Linux Features for End Users

To make sure that Linux continues to be usable on today's state-of-the-art desktop systems, Linux developers must provide the needed device support for a dizzying variety of hardware devices, such as sound cards, printers, mice, keyboards, network cards, modems, storage devices, video devices, CPUs, memory technologies, and motherboard chip sets; only rarely do hardware vendors provide the drivers themselves. Some vendors refuse to disclose the information required by driver authors; in such cases it is impossible to use such devices with Linux. In addition, Linux developers must keep up with rapidly changing technology; in the space of just 2 years (1998 to 2000), for example, CPU speed tripled and new bus technologies (USB and FireWire/IEEE 1394) gained widespread support. During this period, Linux failed to provide adequate support for existing Plug-and-Play (PnP) technologies, let alone the new ones.

In general, Linux lags behind commercial operating systems in hardware support. However, the 2.4 kernel made up for some of the lost ground; for example, support was finally provide for ISA Plug-and-Play devices, and support was provided for the following features of desktop computer systems:

- **Frame buffer video and hardware graphics acceleration.** These features are helping to transform Linux into an attractive gaming platform.
- **Emerging peripheral bus standards,** including Universal Serial Bus (USB), IrPORT IrDA (infrared), IEEE 1284 advanced parallel port transfer modes (EPP and ECP), and IEEE 1394/FireWire.
- **Advanced configuration and power interface (ACPI).** This Intel-developed power management standard is the standard power conservation architecture for new PCs.

The key to Linux's success in the desktop, end-user marketplace is software availability. As economists point out, end user acceptance of operating systems amounts to a self-fulfilling prophecy; if end users believe that more applications will become available for an operating system they like, they will be more inclined to choose that operating system over one that has a questionable or diminishing pool of applications. At this writing, the dearth of high-quality, full-featured Linux applictions constitutes the major barrier to the operating system's acceptance in the end-user, desktop market. Nevertheless, the pool of Linux-compatible applications continues to grow. A major step forward was Sun Microsystem's decision to release the code of its full-featured office suite StarOffice, in two versions, a commercial version (called StarOffice) and a GPL-licensed version available from Openoffice.org. Both provide Linux users with a full-featured office package that includes an outstanding word processing program as well as good spreadsheet and presentation modules. In addition, all of these modules are capable of reading from, and writing to, Microsoft Office file formats.

Key application software developments can quickly fuel Linux's penetration into important vertical markets. For example, Beowulf clustering software enables organizations to assemble what amounts to a bargain-basement supercomputer, constructed from dozens of Linux systems that are networked by means of ultra-high-speed links. The Beowulf software provides high-speed network drivers, and more importantly, the parallel computing support that enables the cluster to break a complex program down into parts so that it can be quickly solved. Linux-powered Beowulf clusters are leading to major advances in sci-

entific and engineering research laboratories worldwide; in China, for example, a Linux cluster helped researchers map the gene sequence of traditional, disease-resistant varieties of rice, while University of Illinois researchers are using a similar cluster to improve aeronautical understanding of turbulent air flow.

C. Toward an Implosion?

Some question remains whether Linux kernel development is headed in the right direction. Each new version of the kernel seeks to keep up with the frenetic pace of PC hardware innovation while, at the same time, beefing up the operating system's capabilities for the high-end, high-demand server market. Despite the considerable achievements and genius of the key kernel developers, a point may be reached in which the demands of pursuing both of these goals results in ever-mounting delays and unreliable code. Although earlier versions of the Linux kernel developed an enviable reputation for reliability, the same cannot be said of the first few ostensibly stable releases of the considerably more complex 2.4 kernel, which proved sufficiently unstable that some system integraters moved their customers back to the 2.2 kernel.

In contrast to Linux, the developers of three freely redistributable versions of the Berkeley Software Distribution (BSD), an early UNIX variant, do not attempt to make their systems compatible with the latest cool gadgets; instead, they focus on optimizing their products so that they function efficiently as servers. Also developed initially for Intel-based hardware, these three BSD variants—called FreeBSD, NetBSD, and OpenBSD—pose a growing challenge to Linux's prospects in the high-end, high-demand server market. Currently, FreeBSD—not Linux—is the operating system of choice if one is in the market for an open-source server operating system that is capable of dealing reliably with punishing loads. Still, the BSD distributions are themselves controversial, owing in large measure to a long-simmering dispute in the open source community regarding the merits of the BSD license. Like the GPL, the BSD license makes the source code available to recipients and allows free redistribution. Unlike the GPL, the BSD places no constraints on the manner in which subsequent distributions take place. Under the BSD license, one can take the code, lock it up in closed source binary programs, and charge money for the resulting product, without violating the license.

Advocates claim that the BSD license, unlike the GPL with its "copyleft" associations, is more amenable to the corporate mind-set; critics point out that the license is amenable to corporations for a reason, namely, that corporations can appropriate BSD code and incorporate it into commercial, closed-source products. For example, Apple's System X is a copyrighted, closed-source operating system based partly on BSD.

V. ADVANTAGES AND DISADVANTAGES OF LINUX

Despite its many attractions and advantages, summarized in the following section, Linux is clearly not for everyone; currently, considerable technical expertise is needed in order to use the operating system successfully. Other disadvantages, including poor technical documentation, the lack of traditional technical support, and the lack of vendor accountability, are delaying the business community's acceptance of Linux.

A. Advantages

Compared to competing operating systems, Linux offers the following advantages:

- **Zero price tag.** Thanks to its GPL licensing, Linux is available at no cost, and there is no additional cost for making duplicate copies for additional computers. Of course, Linux is not without costs; since it is considerably more difficult to learn and use than consumer operating systems, the zero price tag advantage will be offset to some extent by increased training and administrative costs. Nevertheless, the zero price tag extends to GPL-licensed applications, as well, making the Linux platform exceptionally attractive to Third World countries, nongovernmental organizations, charitable organizations, and others who have difficulty coming up with the funding for commercial operating systems and software.
- **Excellent server performance.** Linux has generally equaled or surpassed the performance of Microsoft Windows NT on single-CPU systems, but Microsoft's operating systems held the edge, until recently, in multiprocessor systems. Thanks to the Linux 2.4 kernel's enhanced SMP and 64-processor support, Linux-powered servers are able to equal or surpass not only the performance of Windows products, but also that of high-end commercial UNIX systems.
- **Outstanding reliability.** Linux is well known to be one of the most reliable operating systems

available. University of Wisconsin researchers found that the Linux kernel performed, on average, twice as well as the leading commercial UNIX kernels in reliability scores. It is common for Linux kernels to remain running for months, and even years, without crashing. The Linux kernel is also demonstrably more reliable than Microsoft Windows NT, but Linux may no longer have so pronounced an edge; Microsoft's more recent operating systems are considerably more stable.

- **Quick security fixes.** In the commercial software world, the security measures built into operating systems are regarded as trade secrets. In open source software such as Linux, the security measures are open for everyone to see—including attackers. For Linux critics, the fact that the source code is open to inspection poses a security risk. They say that intruders can study the code to determine how to break into Linux systems, and they prefer to purchase commercial software from vendors who regard the source code as a trade secret. Security experts discount such concerns. Linux is created and maintained by expert programmers who are working in real-world situations, in which their systems are constantly exposed to intrusive attacks. When they discover a security flaw, fixes are created quickly and distributed by means of the Internet, often within hours of a security breach.
- **Freedom from licensing shakedowns.** Commercial operating systems specify severe penalties for failure to remain in compliance with licensing terms; should a company breach these terms, the result could be an expensive, time-consuming lawsuit or even criminal charges. An organization can violate commercial software licenses inadvertently; for example, some commercial Internet transaction programs specify a maximum number of concurrent transactions, but this number may be exceeded should the service experience an unexpected surge in demand. If the transgression is detected by the vendor's monitoring software, the organization could find itself threatened with hefty fee increases or a lawsuit. Commercial software licenses expose organizations to unwanted scrutiny from vendors; Linux carries no such burden.
- **Customizability.** Linux is distributed under the terms of the GPL, which means that recipients have the right not only to view the source code, but to create derivative versions of the code if they wish. Derivative versions of Linux source code must be made available to the public under the GPL, but this is true only if the derivative code is publicly distributed

B. Disadvantages

Linux's advantages make the operating system particularly attractive to professional network and system administrators with UNIX experience. The disadvantages, summarized in this section, will prove daunting to end users and nontechnical IT purchasers.

- **Difficulty of installation and use.** Even Linux's most ardent backers concede that the operating system is challenging to learn, configure, and use. It is modelled on commercial UNIX server operating systems, which are configured and operated by professionally trained administrative personnel. A beginning Linux user can easily render the operating system unbootable by making a single, ill-advised change to a configuration file. Although desktop environments such as GNOME and KDE help to make Linux easier to use than ever before, users soon realize that they will need to learn UNIX system administration skills—for example, running text-mode commands in the shell and writing *shell scripts*—in order to keep their systems running. In addition, users must keep their systems up to date with the latest security patches so that they can avoid common exploits. However, upgrading to a new version of a program (or a new version of a distribution) often brings with it unanticipated consequences, which can take hours or days of work to resolve. Organizations that hope to save money on Linux's zero price tag may find that they are paying the difference in technical support. This situation will improve as versions of Linux emerge that are specifically designed for end users.
- **Forking.** Although the Internet-mediated open source development method employed by Torvalds is clearly capable of creating excellent software, it is also prone to internal bickering. At the extreme, such bickering may lead to a *fork,* in which the code is appropriated—or discarded in favor of a fresh start—by a split-off group with differing ideas about where the project should go. Forking can prove very destructive; at the minimum, it robs open source development projects of their momentum by reducing the number of programmers willing to commit to

long, largely uncompensated hours of coding. At the extreme, forking can divide the entire Linux community and raise serious questions about the viability of open source development. From one perspective, the KDE vs GNOME split is an example of a particularly disastrous fork. From another perspective, it is in line with a key Linux value, namely, that users should be able to configure their systems by drawing from a variety of alternative resources at each level of the operating system (beyond the kernel).

- **Lack of standardization.** Variations among Linux distributions continue to plague Linux developers, who are faced with a bewildering variety of file locations, device nomenclature systems, software installation techniques, and application programming interface (API) disparities. The Linux Standards Base (LSB; see http://www.linuxbase.org) is working to reduce standardization issues by specifying an application programming interface for a standards-conformant Linux distribution—that is, how the distribution works in relation to binary applications. The LSB project seeks to specify a standard that is sufficiently focused to allow any compiled program to be installed and to run without modification on any Linux distribution for a given platform, such as Intel-based PCs. Specified in the standard are locations and interfaces for object files, base libraries, utility libraries, graphics libraries, system initialization routines, system initialization file locations, and user/group nomenclature. LSB calls for the use of the Red Hat Package Manager (RPM), version 3.0, as the standard for installing binary packages on Linux systems; the bash shell is chosen as the standard for LSB-compliant systems. LSB is only one of several Linux standardization initiatives. The Filesystem Hierarchy Standard (FSB) specifies not only a standard filesystem hierarchy, but also the exact location of utility programs and libraries; the FSB standard is incorporated, by reference, into the LSB standard. Another Linux-specific standardization effort is the Linux Assigned Names and Numbers Authority (LANANA), which is attempting to regulate the chaotic naming of device drivers. These efforts will bear fruit only if Linux distributions voluntarily comply with the standard bodies' suggestions.
- **Inadequate documentation.** Few open source programmers take the time to document their software, and even fewer do it well. When they do provide documentation, it is often written unclearly, contains technical errors, is only sporadically updated (if at all), fails to cover significant portions of the software, and assumes considerable Linux and UNIX system and/or network administration expertise. Some of the documentation shortfall has been remedied by commercial publications (the market leader is O'Reilly) and the Linux Documentation Project (LDP).
- **Lack of traditional technical support.** Although commercial Linux distributions such as Red Hat and Caldera provide traditional forms of technical support, direct support (such as telephone support) is expensive—prohibitively so for many potential Linux adaptors, who are attracted by the zero price tag. However, an adept user of the Internet can often find an answer to a Linux technical problem by searching newsgroups, joining Linux channels on Internet Relay Chat (IRC), and browsing technically oriented Linux Web sites.
- **Lack of accountability.** Many companies prefer to purchase commercial software so that, if the software causes a problem, the vendor can be relied upon to solve the problem (or provide compensation if the problem cannot be solved). Although the GPL has never been tested in court, it disclaims all liability for losses, expenses, or damages incurred by anyone using GPL-licensed software. Commercial software licenses typically include such disclaimers, but in some cases the courts have set such licenses aside and held vendors liable. If a customer were to win such a claim against a vendor who provided a GPL-licensed program, the customer would have a tough time collecting; the program's author may be difficult to find or identify and may not have enough money to make legal pursuit worthwhile. Such is the corporate mind-set that, in some cases, these facts alone are sufficient to rule out the use of Linux. Still, the trend toward tort reform makes it increasingly unlikely that software customers can expect compensation from any software vendor, including those with deep pockets, so this disavantage may decline in importance.

VI. THE FUTURE OF LINUX

Linux is here to stay. To be sure, some may question the wisdom of Linux's ambitious, two-pronged goals—namely, to keep up with the latest, cool PC gadgets

while, at the same time, improving the operating system's performance in high-demand, high-load servers systems. But this view ignores the fact that Linux—like Microsoft's various versions of Windows—performs reasonably well in the vast middle ground of operating system applications, ranging from desktop computers to the middle ranges of the server market. When the pool of high-quality, GPL-licensed applications reaches critical mass, Linux—with its zero price tag—could pose a genuine threat to Microsoft's worldwide domination in the operating system and office software markets.

SEE ALSO THE FOLLOWING ARTICLES

Computer Hardware • Internet Homepages • Internet, Overview • Javascript • Operating Systems • Programming Languages Classification • Unix Operating System • XML (Extensible Markup Language).

BIBLIOGRAPHY

Bovet, D. P., and Cesati, M. (2000). *Understanding the LINUX kernel: From I/O ports to process management*. Petaluma, CA: O'Reilly.

Raymond, E. S. (2000). *The cathedral and the bazaar: Musings on Linux and open source by an accidental revolutionary*. Petaluma, CA: O'Reilly.

Siever, E., *et al.* (2000). *Linux in a nutshell: A desktop quick reference*. Petaluma, CA: O'Reilly.

Sterling, T. (2001). *Beowulf cluster computing with Linux*. Cambridge, MA: MIT Press.

Torvalds, L., and Diamond, D. (2001). *Just for fun: The story of an accidental revolutionary*. New York: HarperBusiness.

Welsh, M., *et al.* (1999). *Running Linux*. Petaluma, CA: O'Reilly.

Local Area Networks

Judy Wynekoop
Florida Gulf Coast University

GLOSSARY

backbone High-speed networks connecting LANs.
bandwidth Though formally defined as the range of frequencies in a transmission channel, generally used to mean the maximum throughput of a channel.
baseband A single signal is sent over a medium. Most LANs use baseband signaling.
broadband Multiplexed analog signals are transmitted at different frequencies, allowing multiple communications to be sent simultaneously over a single medium.
electromagnetic interference (EMI) Electrical disturbance from low-frequency waves from natural sources (e.g., the sun) or electrical devices (e.g., light bulbs). Radio-frequency interference (RFI) is a disruption from high-frequency waves emitted by electronic devices (e.g., chips).
frame A grouping of data created at the data link layer of the OSI model that is the basic data unit transmitted on LANs. LAN frame sizes range from small (around 64 bytes) to large (over 18,000 bytes).
open systems interconnection (OSI) model The International Standards Organization's (ISO) standard defining a model for understanding and implementing networks. Standards are defined in seven layers: physical, data link, network, transport, session, presentation, and application. Although not fully implemented at higher levels, LAN standards use the model for physical and data link layer protocols.
plenum-rated cable Cable with a fire-resistant coating required by many local building codes to be used in the airspace in a building above the suspended ceiling, the plenum, when it is used for ventilation. Plenum-rated coating does not emit toxic fumes when burned.
point-to-point A direct connection between devices. Contrast with multipoint or multi-access, where multiple devices share a single connection.

I. INTRODUCTION

A local area network (LAN) is a group of computers and related devices interconnected over a limited geographic area. Devices include any equipment capable of sending or receiving data, such as printers, modem banks, or heat sensors. Although most LANs are found within a single building, a LAN's size can range from a single room to multiple buildings, although no common carrier facilities (such as phone lines or T1 lines) are used in a LAN. LANs allow users to easily share files, applications, databases, and printers or other peripherals, as well as communicate via e-mail or videoconferencing. Without LANs, computers would be isolated. File and resource sharing would be done via "sneakernet"—diskettes containing files would be carried from one computer to another. LANs facilitate distributed processing, process monitoring, communication, and software distribution, and support core business processes in most organizations today.

II. LAN PROTOCOLS

LAN protocols, or standards, are defined on the first two layers of the Open Standards Interconnection (OSI) model—the physical and data link layers. A LAN's protocol stack consists of the specific layered standards that define its operational characteristics. Physical layer standards specify items such as voltages and connectors. At the physical layer, data exist as bits; that is, data are not logically grouped together, nor is there a concept of a "message."

The data link layer has two sublayers. The Media Access Control (MAC) layer defines frame formats, LAN addresses, and standards for sharing the media. Standards at this layer differ for different types of LANs, such as Ethernet or Token Ring. The logical link control layer (LLC) is the same for all LANs and defines interfaces to the next layer in the OSI protocol model (the network layer), as well as methods to check the data link layer header for errors.

At the data link layer, bits are handled in "frames." A frame includes the data link layer header and trailer consisting of the sender's data link address, the receiver's data link address, control information and error checking information, as well as the data to be transmitted, encapsulated inside the headers from all higher layers. A data link address is a unique, permanent 48-bit number assigned by the manufacturer to a network interface card (NIC) which connects a device to the network. The data link address is also known as the media access control address, the hard address, and the physical address.

III. LAN CONFIGURATIONS

A LAN's configuration determines the cost and capacity of the LAN, as well as the type of data it can carry, its transmission speed, and efficiency. A LAN's configuration includes its topology, media access control methods, and transmission medium.

A. Topology

A topology defines how devices are interconnected on the medium. A LAN has two topologies: logical and physical. The logical topology describes how data is sent from node to node on the LAN. The physical topology is the wiring layout, or how the cables are run.

1. Logical Topologies

The two logical topologies are broadcast (also known as bus) and sequential (also known as ring). In a broadcast topology, all devices on the network receive every message transmitted. Each device is responsible for recognizing messages meant for it. If a device is not the intended receiver, the message is ignored. An analogy would be a teacher calling on a student in class. All students in the classroom would hear the teacher, but only the one whose name was called would respond.

In a sequential topology, data moves by point-to-point transmission. Each device connected to the ring receives and regenerates the signal. If the message is not addressed to it, the computer passes it to the next device.

2. Physical Topologies

There are three basic physical topologies: bus, ring, and star. That is, the cables in a network approximate the shape of a bus, ring, or star. In a bus physical topology, all devices are connected in a line along a single channel. Signals propagate along the entire length of the bus. Each end of the bus must be terminated to prevent signal loss and echoes. A drawback to a bus network is that a cable break or loose connection in the network will cause the network to stop working.

In a ring each device physically removes a frame from the media and regenerates it. Therefore, a physical ring has the same shortcoming as a physical bus topology: a cable break or loose connection in the network could cause the whole network to stop working.

In a star, each node communicates directly with only the central device in a point-to-point connection. Therefore, any problems in a particular computer or the media connecting it to the central device do not affect the rest of the network. Various types of hardware, such as multiport repeaters or switches, can serve as the central device, depending on the network architecture. A star is easier to troubleshoot than is a physical ring or bus. If a computer on a star has trouble receiving or sending data, the problem must be between that computer and its connection to the device at the center of the star. Problems on a ring or bus cannot be isolated as easily. However, in the star, since all communication passes through the central device, if it is not working, there is no network.

Networks with both broadcast and sequential logical topologies (i.e., *logical* buses and rings) are usually implemented today as physical stars. The circuitry of the central device replicates the electrical activity of a logical bus or ring—signals are broadcast or passed sequentially within the central device.

There are also variations on the basic physical topologies. For example, a mesh topology is similar to

a star with point-to-point connections between devices that are not in the center of the star. In a full mesh network, each pair of computers is directly connected.

B. Media Access

Media access control (MAC) protocols enforce a methodology to allow multiple devices access to a shared media network. Before LANs, communication between computing devices had been point-to-point. That is, two devices were connected by a dedicated channel. LANs are shared media networks, in which all devices attached to the network receive each transmission and must recognize which frames they should accept. Media sharing reduced the cost of the network, but also meant that MAC protocols were needed to coordinate use of the medium. There are two approaches to media access control in LANs: contention and token-passing.

Contention is a first-come, first-serve approach. Carrier sense multiple access/collision detection (CSMA/CD) is the most used contention-based MAC protocol, used in Ethernet networks. When a device must transmit, the NIC monitors the network to determine whether or not another device is transmitting. The NIC cannot transmit if it senses electrical signals on the network that indicate another device is transmitting. Due to propagation delay (the time it takes a signal to reach a point from its sender), it is possible for two stations to transmit simultaneously or almost simultaneously, causing a collision and garbling the messages. When the sending NIC senses that the message propagating along the network is not identical to that which it is transmitting, transmission ceases. The sending NICs then wait a random amount of time before resending the data. The busier a network is, the more often collisions occur and the longer it takes to transmit data.

CSMA/CA (Collision Avoidance) is a contention method designed to prevent collisions. In CSMA/CA, the NIC monitors the line for a longer time and the line must be idle for a specified period before a station can transmit. A short handshaking packet is then sent before the message. If there is a collision due to simultaneous transmissions, only the handshaking packets collide and the sending stations wait and try again. CSMA/CA is used in wireless networks, since devices are farther apart in a wireless network than in a wired LAN.

Token-passing ensures a sending station has the network's entire bandwidth to itself by requiring that a sending station possesses a specific data frame (the token) before it can transmit. On shared-media LANS with high data traffic, greater throughput is achieved using token-passing than a contention-based access method, since there are no collisions with token-passing. Since token-passing is a deterministic access method (the timing of signals can be predicted), it is well suited to time-dependent traffic, such as telemetry. Contention is nondeterministic.

C. Media

Media can be guided or broadcast. Guided media include conductive (wire-based) cabling, such as shielded and unshielded twisted copper wire and coaxial cable, as well as fiber-optic cables (Table I). Wireless LANs transmit data by broadcasting them over frequencies in the electromagnetic spectrum.

Most LANs currently use guided media, which differ on many characteristics, including bandwidth, maximum segment length (the maximum distance a signal goes before attenuating beyond reconstruction), and susceptibility to interference from naturally occurring and electrical sources (electromagnetic interference, EMI, and radiofrequency interference, RFI) in the environment. Not only do different types of media differ in cost, but specific types of media have a wide cost range (see Table I). Cable quality and bandwidth affect the cost of guided

Table I Guided Media Characteristics

Media type	Also called	Price ($/ft)
Unshielded twisted pair (categories 3–7)	UTP	0.10–0.20 (up to 0.30 for plenum-rated)
Shielded twisted pair	STP	0.42
Coaxial cable—thick	Frozen yellow garden hose; Thicknet	1.10
Coaxial cable—thin	Thinnet; Cheapnet	0.32
Fiber optic	Fiber	1.00

media, as does the casing material. Nonrated cable is roughly half the cost of plenum-rated cable.

Data are transmitted as electrical voltages over conductive cabling. Unshielded twisted pair (UTP) is currently the most popular medium for connecting desktop computers to LANs. Twisting copper wire reduces crosstalk, interference from neighboring lines, and interference from other environmental sources. Cables normally have two or four pairs of wires. Each pair is used for two-way communication. The more twists per foot in copper wire, the higher the quality of data communications over the wire. UTP Categories 3 and higher are considered to be data grade, since they meet electrical standards that ensure transmission speed and quality, while categories 1 and 2 are considered voice grade. Since data propagate through the wire as electrical voltages, UTP is susceptible to EMI and RFI. Shielded twisted pair (STP) has more insulation from interference than does UTP. However, STP is harder to work with and costlier than UTP, and is used less frequently. Category 5 UTP is the most used LAN media.

Coaxial cable (coax) is also made from copper conductors, which share a common axis (hence the name). This was the first type of cable used in LANs and is still widely found in existing LANs. Coax comes in various grades. Coaxial cable has more shielding and insulation than does twisted pair, so data can be sent with fewer errors. There are few new installations of coax in LANs, since fiber is more cost-effective for long cable runs such as backbone networks, and UTP is more cost-effective to the desktop.

Fiber-optic cable, unlike conductive cabling, is made from solid glass or plastic filaments and carries data as pulses of light emitted by a laser or light-emitting diode (LED), not electrical voltages. Since the signal is not susceptible to EMI or RFI, signals can travel farther before being amplified on fiber than over coax or twisted pair. Fiber has the highest bandwidth of the guided media, and signals can travel farther without being regenerated. However, fiber is costly to install and maintain.

There are two kinds of fiber-optic cabling: single-mode (SMF) and multimode (MMF) fiber. SMF focuses the light so that only a single wavelength is transmitted at a time. In MMF, multiple wavelengths are transmitted, causing numerous reflections and distortion. Throughput and segment length are greater in SMF, and SMF is more expensive.

Wireless LANs can use infared, laser, or spread spectrum radio transmission. Currently, wireless LANs exist mainly at low speeds, that is, under 20 megabits per second (mbps), although faster speeds will become more prevalent.

IV. BASIC LAN STANDARDS

The Institute for Electrical and Electronics Engineers (IEEE) formed the 802 committee to define standards for LANs and metropolitan area networks. The 802 committee creates LAN standards, defining physical and data link layer specifications based on the OSI architecture. The main LAN standards in use today are Ethernet, token ring, fiber distributed data interface (FDDI), and asynchronous transfer mode (ATM). Architectures that may still be found in legacy installations include AppleTalk, ARCnet, and 100VGAnyLAN. Representative architectures discussed here are summarized in Table II.

A LAN architecture includes a physical and logical topology, frame format, and MAC protocols. A network technology that defines a data link frame format, MAC methods, and a logical topology can generally be wired in more than one way—it can have more than one possible physical topology or media.

A. Ethernet

Due to its low cost and ease of installation, Ethernet is today's most popular network architecture. Ethernet networks use a logical broadcast topology and MAC is done using CSMA/CD. Although the term Ethernet is generically used today to refer to all such LANs, there are different, incompatible Ethernet standards, each having a different frame format.

Early versions of Ethernet were developed by Xerox Corporation's Palo Alto research Center (PARC) in the 1970s. In 1982, Xerox, Digital, and Intel published the first 10 mbps Ethernet standard, using baseband signaling and a variable-length frame over thick coaxial cable. Ethernet was defined as a physical bus, with a logical broadcast topology, using contention as an access method. In 1990 IEEE published its 802.3 specification.

Table II includes the most used Ethernet standards. The names of Ethernet standards include information about the physical layer specifications, such as the maximum speed possible (e.g., 10 mbps; 100 mbps) and the type of signaling used (baseband or broadband). The third part of the name refers to media characteristics. Physical bus specifications (e.g., 10Base5) include a reference to the maximum segment length (e.g., in 10Base5, its 500 m). Early Ethernet specifications (10Base5 and 10Base2) are physical buses using coaxial cable. 10Base5 networks using thick coax may still be found in legacy network backbones and 10Base2 LANs using thin coax still exist, although new installations are limited.

Table II Characteristics of Representative LAN Technologies

Standard	Also called	Current primary use	Characteristics	Logical/ physical topology	MAC protocol	Maximum speed (mbps)	Media
Ethernet							
10Base5	Thicknet	Legacy backbones	Hard to install and troubleshoot; costly	Bus/Bus	CSMA/CD	10	Thick coax
10Base2	ThinNet	Legacy LANs	Easy to install; hard to troubleshoot	Bus/Bus	CSMA/CD	10	Thin coax
10BaseT	Star-wired bus	LANs	Easy to configure and troubleshoot	Bus/Star	CSMA/CD or switched	10	UTP Cat 3+
100BaseTX	Fast Ethernet	LANs or backbones	Easy to configure and troubleshoot; costly; short segments	Bus/Star	CSMA/CD or switched	100	UTP
100BaseFX	Fast Ethernet	LANs or backbones	Easy to troubleshoot; costly	Bus/Star	CSMA/CD or switched	100	MMF
1000Base LX/SX	Gigabit Ethernet	Backbones	Fast; costly	Switched/Star	Switched	1000	SMF; MMF
1000BaseT	Gigabit Ethernet	Backbones	Fast; costly; short segments	Switched/Star	Switched	1000	UTP Cat 5e +
Token Ring							
4/16 mbps	Star-wired ring	Legacy LANs	Reliable; costly; hard to troubleshoot	Sequential/Star	Token passing	4/16	STP; UTP
100 mbps	High speed token ring	4/16 mbps LAN upgrades	Reliable; costly; hard to troubleshoot	Sequential/Star	Dedicated token passing	100	UTP Cat 5 or fiber
FDDI	Fiber distributed data interface	Backbones	Fast; secure; reliable; long distances; costly	Sequential/Ring	Modified token passing	100	Fiber
ATM	Asynchronous transfer mode	Backbones	Fast; costly	Switched/Star	Switched	155	Fiber
Fiber channel	Fiber channel	SANs; peripherals	Fast; lack standards	Switched/Mesh	Switched	100 mbps –3.2 gbps	Fiber; STP; coax

Most current Ethernet installations are 10/100 BaseT, with a physical star topology, using a hub or switch as the central device, and a 10- or 100-mbps data rate. Ethernet specifications defining a physical star, such as 10BaseT or 100BaseTX, include a media reference instead of segment length. The "T" in 10BaseT stands for twisted pair; the "F" in 10BaseF stands for fiber. The central device, or wiring center, is typically a hub. A frame sent from an attached computer enters the hub at the PC's port, but leaves the hub through all ports. Therefore, media access (contention) occurs in the hub. Whether Ethernet is implemented as a physical star, as in 10BaseT, or a physical bus, as in 10Base2, contention, and therefore collisions, still occur.

Although 10/100BaseTX is popular, a cable connecting a PC to a hub cannot exceed 100 m (328 feet). If longer connections are needed, multiple hubs can be linked together, up to a specified point (e.g., 4 hubs may be used in 10BaseT and 2 in 100BaseTX). Backbone networks may be used to vastly increase the span of a LAN by interconnecting LANs using switches or routers.

B. Token Ring

Token ring networks are defined in the IEEE 802.5 specification. The 802.5 specification defines a logical ring, physical star topology, using token passing for media access. IBM, the leading proponent of token ring networks, in conjunction with the IEEE, has defined network architectures operating at 4, 16, and 100 mbps. A gigabit token ring specification is currently being developed.

The "token" is a specific data frame constantly moving around the network. When a station wants to transmit, it alters a field in the token to indicate it is in use, appends the data to be sent and address information to the token, and sends the frame. The intended receiver changes bits in the token, indicating the address was recognized and the frame copied, and sends the frame on. When the original sender receives the frame indicating it was successfully received, it releases the token by resetting the fields before transmitting it. Only one data frame can be transmitted before the token is released to the next station, ensuring equal network access and making token ring deterministic.

Gigabit and 100 megabit token ring use a different token-passing MAC protocol, called dedicated token ring (DTR), based on using a switch as the central device of the star. Each node is connected to the switch using a full duplex, point-to-point connection.

Although token ring LANs have greater throughput than do Ethernet shared media LANS, Ethernet has the greater market share, since it has continually been less expensive and complex, with adequate capacity. According to IBM, customers who use token ring need reliable deterministic networks for mission critical applications. New installations of token ring networks are rare and most installations of 100-mbps token ring are done to upgrade existing token ring sites.

V. HIGH-SPEED NETWORK ARCHITECTURES

Until the last few years of the twentieth century, LANs were mainly used for e-mail and data file transfer. However, as hardware and Internet technologies have evolved, traffic traveling over LANs has increased. Large data files, as well as multimedia files, including audio and video, are now being downloaded and transferred. LAN links to high volume servers or workgroups and backbones connecting individual LANs need more bandwidth than shared media LANs can supply.

A. Fast and Switched Ethernet

Until the 1990s, LANs were largely shared media networks—all devices transmit a baseband signal over the same channel, necessitating MAC methods. Switched LANs establish point-to-point connections between sending and receiving devices, eliminating the need for shared media. Since switched Ethernet is the dominant architecture today, this discussion is limited to Ethernet.

In shared media Ethernet, although using a hub changes the physical topology of the LAN, it does not change its logical topology or data link layer. That is, messages are still broadcast and CSMA/CD is still used, albeit inside the hub (Fig. 1). Due to collisions, on busy 10baseT LANs, throughput may actually be only several megabits per second, rather than 10 mbps. Although throughput is higher on 100BaseTX LANs, networks can only be up to 200 meters from end to end (versus 500 meters in 10BaseT).

To improve throughput without sacrificing network size, switched Ethernet can be used. A switched LAN architecture uses a LAN switch, or switching hub, as the central device in a star topology. A switch is a data link layer (OSI Layer 2) device that establishes a di-

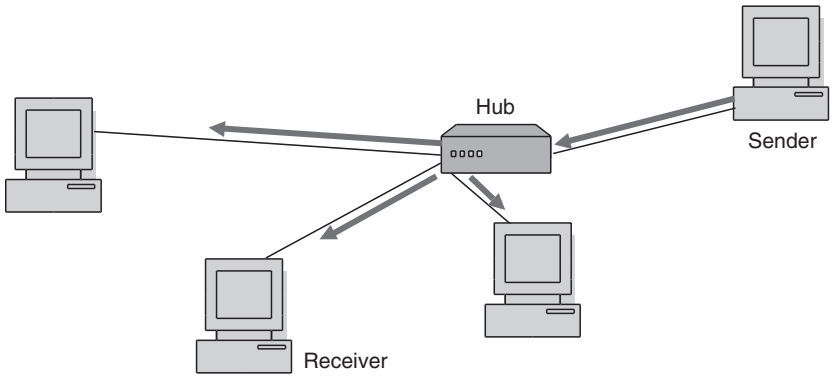


Figure 1 Shared Media 10BaseT LAN.

rect, dedicated connection between the transmitting station's port and the receiving station's port, thus eliminating shared media problems and most collisions. In a switch, messages are not broadcast out all ports, but only leave through the port to which the designated receiver is attached (Fig. 2). Although 100BaseTX can be implemented with a hub, greater throughput is achieved when a switch is used. Gigabit Ethernet (1000 mbps) must be switched.

Ethernet was originally designed to have a broadcast logical topology. Switching allows parallelism in a LAN and transmission becomes point-to-point. The issue of whether or not switched Ethernet is really Ethernet remains unresolved. Both switched Fast Ethernet and Gigabit Ethernet maintain the Ethernet frame format and use CSMA/CD, even though no collisions are detected. Thus, upgrading to switched Ethernet from hub-based Ethernet is simple—the hub is simply replaced by a switch. With a switch, each device is connected with the capacity equal to the entire original LAN. Increasingly, larger and busier Ethernet LANs are implemented with a switch instead of a hub.

When Ethernet is switched, transmission can be full duplex, essentially doubling throughput. Shared media Ethernet is half duplex; that is, a device can send or receive, but not both at the same time. Gigabit Ethernet (IEEE 802.3z) is full duplex and Fast Ethernet is most effective when it is full duplex. Full duplex Ethernet requires full duplex NICs, two channels connecting each device to the switch, and the use of full duplex switches.

B. 100VGAnyLAN

When Fast Ethernet standards were in development, the committee split into two groups, one supporting CSMA/CD as a MAC method, and the other favoring token-passing. The second group developed a standard known as 100VGAnyLAN (IEEE 802.12). The standard defines 100 mbps transmission over 4-pair UTP, fiber and 2-pair STP. "VG" indicates that voice grade twisted pair is included in the specification. "AnyLAN" refers to the fact that 100VGAnyLAN supports all design rules and topologies for 10BaseT and token ring, facilitating interconnectivity.

100VGAnyLAN uses Demand Priority, a centrally controlled token-passing method, for MAC. Demand Priority eliminates delays from both collisions and token-passing and allows traffic priorities to be set. This guarantees support for time-critical applications, such as multimedia and video. Although many considered 100VGAnyLAN superior to 100BaseTX, the popularity of Ethernet was unshakable. Few new installations of 100VGAnyLAN will occur.

C. FDDI

FDDI is an ANSI (American National Standards Institute) not IEEE, standard, although it does support IEEE logical link control protocols and can be used with IEEE-compliant upper layer protocols. FDDI has a sequential logical topology, physical ring topology, data rates of 100 mbps over fiber, and uses modified token-passing for MAC. Standards for FDDI over copper wire, Copper Distributed Data Interface (CDDI), have been developed.

Nodes on an FDDI network are typically connected to two counter-rotating rings, offering reliability. FDDI is self-healing—if there is a break or problem in the main ring, traffic will automatically be routed over the secondary ring. This overcomes the susceptibility of token ring networks' failure. The cost of an FDDI

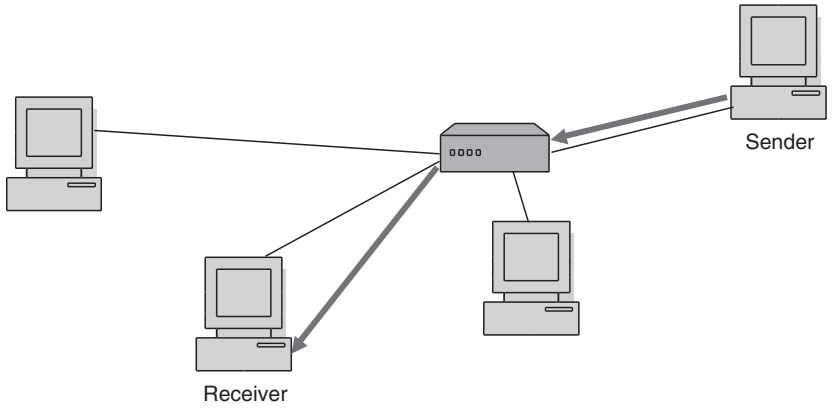


Figure 2 Switched Ethernet.

network can be decreased by attaching devices to only one ring, but reliability is diminished. Even with dual-attached devices, however, FDDI is fault-tolerant against a single failure only. Due to its reliability and throughput, FDDI is often used for backbone networks to interconnect LANs within an organization.

FDDI uses modified token-passing, since the speeds and distances involved do not require the "free" state of the token to be set to "busy" until the sending station receives delivery confirmation, as in token ring. In FDDI, a sending station removes the token from the ring, sends its frame, and releases the token immediately. Collisions are still avoided since a station cannot transmit without the token.

D. ATM

ATM is a versatile, high-speed, switching technology that transmits data in fixed-length cells. Although LANs may be ATM-based (ATM to the desktop), it is generally not needed or cost-effective in any but highly video-intensive environments. ATM is most often implemented using fiber as a backbone network connecting non-ATM LANs (e.g., Ethernet). A LAN, such as a 10/100BaseT LAN, connects to a router or switch which connects to the ATM backbone. Because of the fixed-length cells, ATM is isochronous—transmissions can be predictably timed. This allows reliable transmission of streaming traffic, such as voice and video.

VI. WIRELESS LANS

Wireless LANs today are generally used with wired networks, rather than alone. Wireless LANs are used to extend wired LANs beyond distance limitations, connect wired LANs across right-of-ways, connect nomadic users to wired LANs, give mobile users access to office LAN resources, or provide backup connectivity for a wired LAN. Wireless LANs can also be found in older or historical buildings where it would be prohibitively expensive or impossible to run in-wall cabling.

Most current wireless LANs operate at speeds under 11 mbps, although 20-mbps LANs are appearing. Wireless LANs have no set physical topology. Unlicensed frequencies of the electromagnetic spectrum are used for wireless data communication. The higher the frequency, the tighter the broadcast beam, requiring an increasingly clear line of sight between transmitter and receiver. Wireless LANs typically use infrared, laser, or spread spectrum radio in unlicensed frequency spectra. In a wireless LAN, the NIC attaches to an antenna, not a cable. An access point (i.e., transmitting/receiving device) must be attached to the cabled network to communicate with portable devices.

Although infrared is high bandwidth, infrared transmissions in LANs are limited by range or line of sight. Typically, infrared technology is used to dock laptops and other portable devices to wired networks. Since infrared is close to the frequency of visible light, it is subject to interference. Most current infrared technologies require an unobstructed view between the transmitter and receiver.

Laser-based LANs also require a clear line of sight between transmitter and receiver—solid objects block the beam. Laser-based LANs share the limitations of infrared LANs, although they are not subject to interference from visible light.

IEEE 802.11 specifications define three physical layer standards for wireless LANs, including specifications for diffuse infrared LANs at speeds of 1–2 mbps and for spread spectrum LANs at speeds of 1–2 mbps. The recent 802.11b specification defines speeds of 11 mbps, with a 20-mbps standard (802.11g) in process. Spread spectrum LAN technologies simultaneously transmit a signal over multiple frequencies. Although using multiple frequencies decreases the chance of interference, microwaves, cordless phones, scanners, and Bluetooth devices that use the same frequencies can interfere with 802.11 LAN signals. IEEE 802.11 LANs can easily be integrated with wired Ethernet LANs.

IEE 802.11b standards are being coordinated with the European wireless LAN standards defined in HIPERLAN/2, which defines speeds up to 54 mbps for short-range communication.

VII. LAN SOFTWARE

Computers on a LAN run telecommunications software, often called a network operating system (NOS). The two major classes of NOS are peer-to-peer and client/server.

In a peer-to-peer network, all computers use the same NOS and can be configured to be a service provider (server), a service requestor (client), or both. Individual users can share files, printers, and other resources on their computer with others and are responsible for the security of their own files and hardware. For example, I could share a file on my computer and allow everyone else on my LAN to use it. However, I might only want one or two people to be able to share a particular database, so I would set up accounts and passwords for those users, and they would log-in to my database when they wanted to use it.

In a client/server network, service providers and requestors each use a different operating system. Servers, such as file servers, application servers, mail servers, or database servers, use a server NOS, which is more complex and costly than a client operating system. Usually a specific person or group is designated as the network administrator to manage the servers and the network.

Most current client operating systems include the capabilities of an operating system, as well as client software to communicate with the server NOS. Most also include peer-to-peer networking capabilities. Popular client operating systems include Microsoft's Windows 9x, 2000, and NT Workstation and Apple's Macintosh Operating System. Client operating systems include data link layer (MAC) drivers; network and transport layer protocols; and network redirectors. The redirector is part of the operating system that intercepts all program or user input and output requests and diverts requests for network resources to the network. Many client operating systems contain remote access software (RAS) to allow them to connect remotely over phone lines to RAS servers.

Server network operating systems are more powerful and optimized for the tasks of a server. The server accepts requests for services from clients and returns responses to the client via the network. The most widely used server operating systems are Windows NT 4.0 (and its successor Windows 2000 Server); Netware 4.x and 5.x; Linux; and versions of UNIX. Traditionally, Netware has excelled at file and print services and Windows NT and UNIX at application services. However, current releases of both Microsoft's and Novell's products have leveled the playing field.

Basic network management software may be bundled with the server operating system; at other times it is purchased as a third party application. The increasing use of LAN servers for mission-critical applications is demanding more sophisticated network management and monitoring software to allow network managers to identify network problems and bottlenecks and take timely action. Many network devices, such as stackable or enterprise hubs, ship with local management software supplied by the manufacturer. Managed devices can also typically share management information with enterprise network management systems, such as HP OpenView or SunNet Manager. Network management information can be exchanged because it is formatted according to the simple network management protocol (SNMP) by specialized software called agents, gathering and formatting statistics in the network device being monitored.

Server operating systems also provide network security features. Authentication features identify who is trying to access the system, typically requiring a user ID and password. Authorization features control access to specific network resources. Traditionally, this has been done using access control lists, which list all network users and the rights they have to specific resources, such as files, databases, or printers. Each time a user tries to access a network resource, the server operating system checks the access control list before allowing access. As security becomes increasingly important in LANs, many expect increased use of Kerberos, a multiserver authentication method heretofore used only in mainframe and large client/server networks.

Various other types of utility software are used on LANs. Servers, and some clients, use backup software to back up critical data onto tape. Although network operating systems come with security features, such as requiring a user ID and password to log-on to the network, in many cases additional security or authentication software is used. Virus detection software is also typically used on servers and clients.

VIII. LAN HARDWARE

Computers require network interface hardware to handle the transmission and receipt of data frames. Specialized telecommunications hardware, such as hubs, routers, or switches, are also used for various network purposes.

A. Network Interface Cards

Devices are connected to the medium by a NIC, also called a network adapter card or network card. The network adapter may also be found on a computer's motherboard. The NIC contains electrical circuitry implementing data link and physical layer standards, including a port to connect to the LAN's medium. Each communicating device (node) on a LAN must have at least one NIC. If a data frame is addressed to the computer, the NIC stores a copy of the frame in a buffer and interrupts the CPU.

A NIC supports specific wiring standards and connectors. Some NICs, called "combo cards," contain ports for multiple connectors. The actual NIC/network connection depends on the network architecture. Sometimes the NIC contains all the necessary interface circuitry and attaches directly to the network (e.g., 10/100BaseT); and other architectures (e.g., 10Base5) require a drop cable from the NIC in the

computer to another electrical component, the transceiver, which attaches to the network.

Network operating systems communicate with the NIC through NIC driver software called Network Data-Link Interface Specification (NDIS, developed by Microsoft and 3COM) or Open Data-Link Interface (ODI, developed by Novell and Apple). Most NICs today support both NDIS and ODI.

B. Wiring Centers

Until recently, the central device, or wiring center, in a physical star LAN was most often a hub. Each device on the network has at least two connections to the hub—one to send and one to receive. Hubs are physical layer devices and act as multiport repeaters—the hub repeats the incoming signal on the outgoing line(s). The hub also repeats any noise or interference it receives. In an Ethernet hub, a signal comes in one port and goes out all the others. The central device in a token ring LAN is a multistation access unit (MAU), with more complex circuitry than a hub.

Self-contained hubs may be stand-alone or stackable. Stackable hubs include a stacking port that allows multiple hubs to be connected and act together as one large device. Enterprise hubs, also called intelligent concentrators or smart hubs, are modular and therefore more flexible and expandable. Enterprise hubs can be configured by adding different modules such as: Ethernet, ATM, or token ring modules; management modules; router, bridge, or wide area network (WAN) connectivity modules; or redundant power supplies. Managed hubs may be remotely monitored using network management software.

Although hubs change the physical topology of a network, they do not change the network architecture. Hence, although 10/100BaseT is a physical star, rather than a bus, the logical topology is still broadcast and only one device on the LAN can transmit at a time. Recently, layer 2 (data link) switches (also called switching hubs) have become popular for use as the central device in Ethernet LANs, since switches overcome broadcast limitations.

A switch is a data link layer device that establishes a direct, dedicated connection between the transmitting station's port and the receiving station's port, thus eliminating the shared media concept, as well as most collisions. That is, in a switch, messages are not broadcast out all ports, but only leave through the port to which the designated receiver is attached.

Like hubs, switches can be stand-alone or enterprise (modular). Switches are available for Ethernet (10/100 mbps or gigabit), as well as token ring, FDDI, and ATM. High-end switches support multiple LAN architectures and can interface to WAN services.

Since switches enable point-to-point communication between any two ports, not only can they be used as a wiring hub, but they may also be used to segment high-traffic servers or workstations to improve the performance of a network.

C. Interconnectivity Hardware

LANs have distance and device limitations. To ensure access times are reasonable and MAC protocols work correctly, LAN specifications define the maximum cable lengths allowed. The length of a LAN can be extended by using internetworking hardware.

Because a signal attenuates as it propagates over a medium, one way to extend the length of a LAN is by adding repeaters, devices to amplify the signal. Hubs can serve as repeaters and can therefore be used to extend the length of a LAN. However, the number of repeaters used to extend a LAN is limited: for example, in a 10-mbps Ethernet LAN there can be at most 4 repeaters.

In addition to extending distances, congestion and inter-LAN connectivity motivate internetworking, or connecting LANs together. To achieve interoperability between LANs, the protocols in each LAN's protocol stack must either correspond to, or be translated into, the corresponding layer on the connected LAN. A widespread way of internetworking is by segmenting the network or dividing the LAN into sections with fewer devices using an internetworking device such as a bridge or a router to connect the sections and translate protocols. Often, LANs are connected to a high-speed backbone network using routers. Servers may be directly attached to the backbone. Distinctions between interconnectivity devices, such as bridges, routers, and switches, are becoming blurred.

1. Bridges

Bridges interconnect LANs that use the same data link layers. More sophisticated bridges can convert between MAC formats. Unlike repeaters or Ethernet hubs, bridges do not propagate interference and do filter frames. A bridge knows which MAC addresses are on which of the networks attached to it, and either keeps a frame it receives on the same network, or forwards it to the other.

Since bridges and repeaters must be connected by cable, they cannot be used when LANs are separated

by public streets or land owned by others. Wireless bridges may be used. Remote bridges or routers can be used to connect two LANs over a leased line or wireless connection.

2. Routers

When a router processes a packet, it first removes the data link layer header to reveal the OSI network layer header, which contains the final receiver's address. Thus, while bridges are data link layer devices, routers are network layer devices. Whereas a bridge filters all frames it receives based on their MAC address, then either forwards packets or keeps them local, a router examines only those packets addressed to it, confirms the existence of the final destination address, and chooses the best path to get the packet to the destination. The router then adds a new data link layer header that includes the MAC address of the next router along the packet's path, and transmits it.

Unlike bridges or data link layer switches, routers can efficiently use networks with redundant paths, balancing network traffic on the different links. Routers can also be used to filter and stop unwanted packets and to create firewalls to protect specific LANs.

Routers are complex and costly. Additionally, since routing is done using software, routing can be slow. However, routers are well-suited for filtering packets and segmenting networks.

3. Switches

Congestion problems are better served via switches, as described above. Switching is essentially multinetwork bridging. However, whereas bridges filter frames using software, switches use hardware, or application-specific integrated circuit (ASIC) chips. Thus, switching is faster than routing. However, data link switches cannot filter or route packets, since they do not understand network layer protocols.

In the future routers and switches will merge functionality into a single device. This can be seen in the development of layer 3 (network layer) and layer 4 (transport layer) switches, which perform routing functionality and server-load balancing, respectively, as well as packet filtering, in ASIC.

Many switches can be configured using software to define a logical, or virtual, LAN, or VLAN. This allows workgroup members to be quickly and easily assigned to VLANs by using software to group the ports to which they are attached. This allows broadcast messages (those that should be processed by multiple computers) to be restricted to a single VLAN, thus better segmenting the network. However, VLANs are hampered today by a lack of standards.

IX. HOME NETWORKS

The growth in mobile work, as well as in home offices and home-based businesses, is driving the need for home computer users to share printers and other peripherals, files, or Internet access. Home and office LANs differ in applications and traffic patterns. Home users not only need to share resources, but also require support for multiplayer games, streaming media, and voice over the LAN. There are four options available for home networks today: wired, phone lines, AC power lines, and wireless.

Wired LAN solutions discussed above, such as Ethernet, can be used for home networks. These are mature technologies, with accepted standards. However, this solution is best used in new construction, when a home can be prewired using category 5 UTP or fiber. Installing cable in an existing home can be unsightly and costly. Rewiring the LAN when room layouts change can also be costly.

Standards and support for the use of phone lines and power lines for networking are emerging. The Home Phone Line Networking Alliance (HomePNA) describes standards based on Ethernet protocols for home networking using phone lines. Since phone calls do not use the entire bandwidth of the line, data can be transmitted over the unused bandwidth via frequency division modulation at speeds up to 10 mbps. Computers are connected to the network by connecting them to phone jacks. Although a simple solution, network size is limited by the number of phone jacks in the home.

The use of AC electrical wiring in the home for networking offers a more flexible solution, since wiring and outlets are more ubiquitous than are phone jacks. Although in the past data transmission over electrical wiring has been slow and unreliable, an industry consortium, HomePlug, is developing standards to improve the technology and achieve data rates of 10 mbps. Although in its early stages, the idea of power line networking is appealing, particularly to those developing technology for smart homes.

Wireless solutions for the home market have had transmission speeds under 2 Mbps, using CSMA/CA and spread spectrum technologies, with speeds up to 11 mbps now available. There are several competitors for the wireless home market, including IEEE 802.11b (also known as WI-FI), shared wireless access protocol (SWAP) developed by HomeRF, and Bluetooth.

Since IEEE 802.11b (discussed above) is the more mature technology, with greater industry support, more operating systems and laptop computers support it. However, it was developed for nomadic computing in offices and campuses, not for the home market. SWAP was specifically developed for the home market, and is better suited to dense environments since its lower powered signals have a short range. SWAP has better support for voice and streaming media than does WI-FI. SWAP incorporates digital enhanced cordless telephony (DECT, the European standard for home networking). Although many believe that SWAP is the technologically superior option for home networking, it is not yet clear whether it can capture the market.

Bluetooth is a competing short wave radio connection standard designed specifically for personal area networks (PANs) and can connect a few devices within a household. Bluetooth is designed to interconnect electronic devices (such as the TV, VCR, and computer), to connect notebook computers to printers, and to connect peripherals, such as keyboards or headphones, to computers.

X. CURRENT AND FUTURE DEVELOPMENTS

Ethernet, specifically 100BaseTX, is the most widely used LAN specification today. Although gigabit Ethernet is still in its commercial infancy, 10 gigabit Ethernet is in development. Thus, Ethernet backbones and connections to the desktop will become faster.

Currently, wireless LANs are much slower than their wired counterparts. For example, wired Ethernet is available in gigabit speeds, whereas wireless Ethernet LANs most often work around 11 mbps. As more bandwidth is opened up for wireless devices, speeds will likely increase. Before wireless LANs become widespread, however, technological standards need to be resolved.

Fiber channel has evolved from the hardware input/output (I/O) channel, scaled up for peripherals and LAN connections. Fiber channel is a mesh topology, with one or more switches establishing port-to-port connections between communicating entities. Fiber channel specifications exist for single and multimode fiber, as well as coax and STP, supporting speeds up to 3.2 Gbps for 2 kilometers over single mode fiber down to about 100 mbps for 80 meters over STP. Fiber channel currently is widely used to connect peripherals to computing devices; rarely for intercomputer connectivity. For general networking use it is competing against Ethernet and ATM.

Storage area networks (SANs) separate data storage from application servers by consolidating storage systems such as disk arrays, optical juke boxes, or tape servers and connecting them to the enterprise network using high-capacity connections. Fiber channel is typically used to interconnect the storage devices. Currently, SANs and fiber channel both lack standardization. However, as the technologies mature and standards develop, they should become widespread.

In short, future LANs will be faster than current LANs and increasingly incorporate wireless segments.

SEE ALSO THE FOLLOWING ARTICLES

Frame Relay • Integrated Services Digital Network • Internet, Overview • Mobile and Wireless Networks • Network Database Systems • Network Environments, Managing • Standards and Protocols in Data Communications • Transmission Control Protocol/Internet Protocol (TCP/IP) • Wide Area Networks

BIBLIOGRAPHY

ETSI. http://www.etsi.org (HIPERLAN2 & DECT)
Goldman, J. E., and Rawles, P. T. (2001). *Applied Data Communications,* Third Edition. New York: John Wiley & Sons.
Home RF. http://www.homeRF.org
Homeplug. http://www.homeplug.org (Power line)
Keogh, J. (2001). *The Essential Guide to Networking.* Upper Saddle River, NJ: Prentice Hall PTR.
Stallings, W. (2001). *Business Data Communications.* Fourth Edition. Upper Saddle River, NJ: Prentice Hall.
Tittel, E., and Johnson, D. (2001). *Guide to Networking Essentials,* Second Edition. Cambridge, MA: Course Technology.
UTexas. http://www.ots.utexas.edu/ethernet/
White, C. M. (2001). *Data Communications and Computer Networks.* Cambridge, MA: Course Technology.
Wireless Ethernet. http://www.wirelessethernet.org (WECA)

Machine Learning

K. Yoshida *Hitachi Ltd.*
A. Sakurai *Japan Advanced Institute of Science and Technology*

GLOSSARY

artificial neural network A pair of a directed graph and a set of functions that are assigned to each node of the graph. Learning systems try to find a best set of functions so that the resulting network approximates observed data.
association rule A rule in the form "if *A* then *B*" which expresses "if *A* appears in an event, *B* appears in it, too."
Bayesian network A directed graph showing interdependences of probabilities among events. Learning systems try to assign most plausible probability distribution to events to explain observed event frequencies.
case-based reasoning An instance-based learning method which simply stores examples and performs analogical reasoning only when it encounters new data. *k*-nearest neighbor is also an example of the instance-based learning method.
decision tree (i.e., classification tree) A tree-type data structure with leaves that specify classes for an example to belong to, internal nodes that specify conditions to be tested, and edges that specify the subtree to be traversed next.
explanation-based learning A learning framework which uses prior knowledge and deductive inference, and can learn even from a single example.
inductive logic programming A framework of machine learning which generates a Prolog program that entails given examples of target concepts.
meta-learning (e.g., boosting, bagging, stacking) Technique to construct a "strong" learning algorithm by combining "weak" learning algorithms. The idea comes from a theoretical framework called Probably Approximately Correct (PAC) learning.
reinforcement learning A framework for finding the best policy that specifies how you act next and that maximizes the total reward that you will get.
support vector machines A learning framework to map samples in a very high dimensional feature space and to find the hyperplane that separates samples into prespecified two classes with the maximum margin.

MACHINE LEARNING, according to Tom Mitchell, refers to "any computer program that improves its performance at some task through experience." Machine learning is vital in many computer applications such as adaptive user interfaces, computer games, speech recognitions, and web search engines/robots.

Roughly speaking, machine learning

1. Receives inputs that describe a new experience, for example, the opponent's move and the current state of a chess game, or today's Dow Jones indexes
2. Outputs the value that signifies the next move in the game or tomorrow's expected Dow Jones Indexes
3. Reorganizes its internal memory to model or hypothesize about the opponent's move or future economic situation.

A machine learning program may not start from scratch. It may start with a set of past experiences or training examples.

One aspect of machine learning is what information is given to the learning program. Suppose that you are to write a chess program that improves its performance when it plays with you or itself. When the program has a teacher who tells it the right move at each position, the learning is supervised learning. When the program has an adviser who sometimes gives it rewards for good moves, the learning is reinforcement learning. When the program has records of chess games and it learns what patterns appear more often than others, the learning is unsupervised learning.

Another aspect is what the program outputs in each experience. It may output the next move, tomorrow's Dow Jones indexes, clinical diagnoses, or the next position and speed of a robot arm. It may output rules or hypotheses expressing the relationships between the current situation and the required outputs, by which we can calculate appropriate outputs without using machine learning programs.

A machine learning program may construct an explicit description of a hypothesis when it receives training examples. Or it may simply keep all the experiences in its memory and obtain its outputs from similar past experiences in memory. In the former case, the program generalizes its past experiences before it encounters a new situation, but in the latter case, it postpones the generalization until it is required to make an output. The latter process is called instance-based learning.

Yet another aspect is how the program is written, or what algorithm and data structure it uses. It may use decision trees, graphs, artificial neural networks, Bayesian networks, logic programs, etc.

We focus mainly on the last aspect to show the readers some basic supervised learning techniques that may help them when writing learning programs. Decision tree learning (DTL) and inductive logic programming (ILP) for discrete values, feedforward artificial neural networks (ANN), and support vector machines (SVM) for continuous values, and Bayesian networks (BN) for probabilistic formalization are described in Section I. The field of machine learning is so broad that many important topics are not covered or are not even mentioned in Section I. In Section II, we briefly describe reinforcement learning (RL), meta-learning techniques, explanation-based learning, instance-based learning, clustering, association rule discovery, and genetic algorithms.

I. BASIC TECHNIQUES

A. Decision-Tree Learning

Given a set of training examples, decision-tree learning (DTL) systems construct a decision tree (sometimes called a classification tree). Here, each training example belongs to a certain class and is described by a set of attributes and their values. The constructed decision tree has leaves that specify classes and internal nodes that specify conditions to be tested. Table I shows the training examples from which DTL constructed the tree shown in Fig. 1.

A decision tree is used to find the class of the new data from their attribute values. Suppose that the examples in Table I are given as new data with their class unspecified. We can use the decision tree in Fig. 1 to find the class of each given data.

Taking, for example, Example 2 in Table I as new data, we test it for the condition specified in the root node of the decision tree shown in Fig. 1. The first test concerns the attribute "Number of legs." Since the "Number of legs" in Example 2 is 2, we then proceed to the second test "Has wings." Since the attribute value of the "Has wings" test in Example 2 is "Yes," the class that this new data should belong to is "Birds," which is specified in the leaf node.

As mentioned above, leaf nodes specify classes. A conjunction of tests specified in the nodes from the root to a leaf describes the conditions to satisfy for data to belong to a certain class. A decision tree is a disjunction of these conjunctions formed along the paths to all the leaves. Therefore Fig. 1 means:

```
If Number of legs = 0
     then Class = Fish,
else if Number of legs = 2
  ∧ Has wings=Yes
    then Class = Birds,
else if Number of legs = 2
  ∧ Has wings=No
    then Class = Humans,
else if Number of legs = 4
    then Class = Animals,
else if Number of legs = 6
    then Class = Insects.
```

Table I Training Examples

		Attributes	
Examples	**Class**	**A_1: Number of legs**	**A_2: Has wings?**
1.	Fish	0	No
2.	Birds	2	Yes
3.	Humans	2	No
4.	Animals	4	No
5.	Insects	6	Yes
6.	Insects	6	No

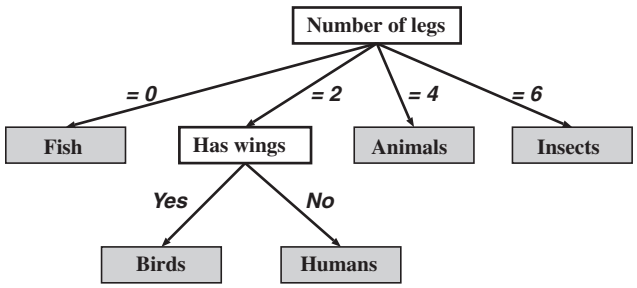


Figure 1 An example of a decision tree.

The divide-and-conquer algorithm shown in Fig. 2 is typically used to construct decision trees. Suppose the data in Table I are given as training examples, then the algorithm constructs a decision tree as follows:

- Starting with an empty tree, the algorithm first selects a number of attributes as test conditions, and then it divides the training examples into groups according to the results of the selected test. In other words, the training examples are divided into groups according to the value of the attributes selected. Let's suppose that the attribute "Number of legs" is selected as the first test condition so that the training examples could be divided into four groups.
- Then the algorithm is recursively applied to construct further subtrees, which classify each group. For example, the first group, whose members have no legs, i.e., "Number of legs = 0," has only one training example and all of the data in this group thus belong to the same class, "Fish." The subtree for this first group is then a single leaf node with the class "Fish."
- The third and fourth subtrees are similar. Each of these subtrees is a single leaf node. The second group, whose members have two legs, includes two training examples. Since these two training examples are of different classes, the algorithm selects the next test condition, "Has wings."

```
Algorithm Divide and Conquer (E)
  Variable  E: Training examples
            T: Decision tree
            c: Test condition
  Begin
    If all the examples in E belongs to class C
      T ← a decision tree having only
              a leaf node with class C
    Else
      T ← ∅
      Select test condition c
      Use c as the test condition of the node
        that classi es E into groups G = {G_i}
      For each G_i in G
        Add the result of "Divide and Conquer (G_i)"
          as a subtree of T
          for the data satisfying test condition c
    End If
    Return T
  End
```

Figure 2 Divide and conquer algorithm for DTL.

Figure 1 shows the decision tree constructed through this learning process.

A criterion called information gain is widely used to select a test condition at each step. It has been taken from information theory, and is defined as:

$$\text{Information Gain } (E,A) = \text{Entropy}(E) - \sum_{G_i \in G} \frac{|G_i|}{|E|} \text{Entropy}(G_i)$$

where

$$\text{Entropy}(E) = \sum_{i=1}^{n} - p_i \log_2 p_i$$

$|G_i|$ = The number of the data which belong to G_i.
$|E|$ = The number of all data.

Figure 3 shows the information gain of the above examples. The entropy of the whole set of training examples shown in Table I is $2.25 = -\frac{1}{6}\log_2\frac{1}{6} - \frac{2}{6}\log_2\frac{2}{6} - \frac{1}{6}\log_2\frac{1}{6} - \frac{1}{6}\log_2\frac{1}{6} - \frac{1}{6}\log_2\frac{1}{6}$. The information gain obtained by selecting attribute A_1, "Number of legs," for the first test is 1.92 [Fig. 3 (1)]. If attribute A_2 "Has wings" is selected, the obtained information gain is 0.58 [Fig. 3 (2)].

By comparing the information gain of each possible test, the algorithm shown in Fig. 2 can select the most appropriate attribute, i.e., the attribute "Number of legs" in this case, as the first test condition.

Practical issues in DTL include avoiding overfitting due to noisy data, handling a lack of attribute values, handling continuous-valued attributes, and handling large data sets, etc. Among these issues, particularly close attention must be paid to overfitting. Real data tend to include noise, i.e., errors in attribute values and class information. A typical problem due to such noise is overfitting of decision trees into training examples. Figure 4 shows the typical accuracy of trees generated during the learning process by the divide-and-conquer algorithm. Although the accuracy of the tree increases on the training examples, its accuracy on the new data starts to decrease after a certain point in the learning process.

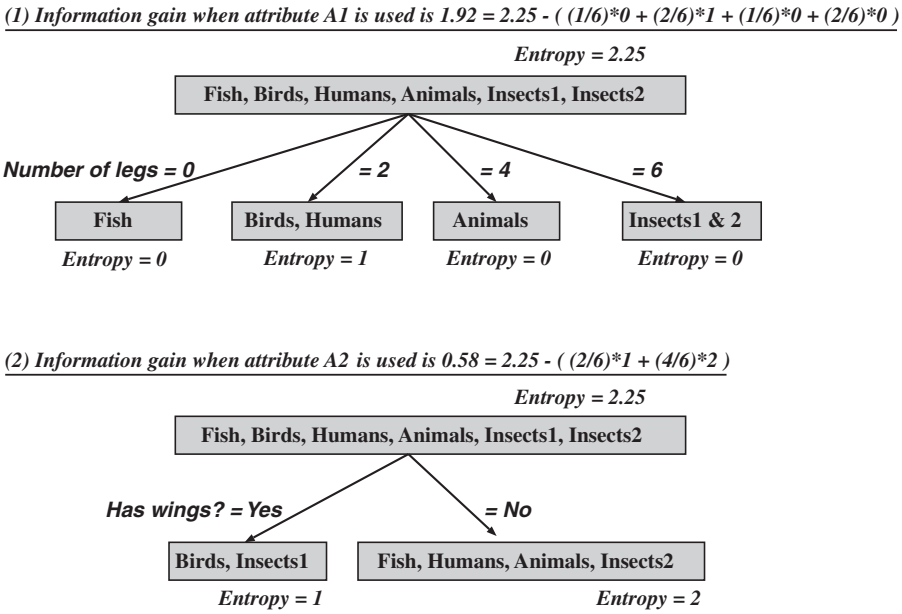


Figure 3 Information gain.

Various techniques to cope with this problem have been developed. One such technique is cross-validation, which is widely used. This technique uses a subset of training examples to construct a decision tree and then uses the remaining data to select the tree that gives the best accuracy for the remaining data.

B. Inductive Logic Programming

Inductive logic programming (ILP) learns a Prolog program that entails given examples of a target concept. The examples of the target concept and the background knowledge are given in the form of a Prolog program.

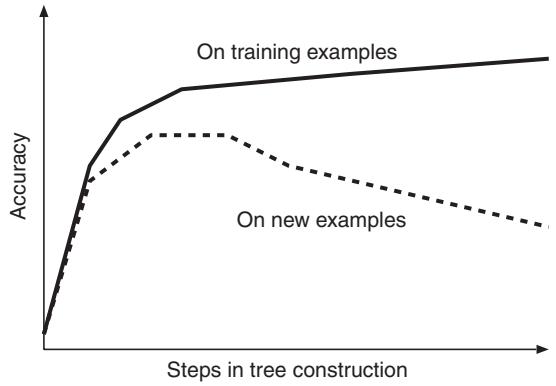


Figure 4 Overfitting into noisy data.

Figure 5 shows an initial Prolog program (examples of target concept C1 and C2 and background knowledge C3 to C6) and a learned Prolog program (C7).

A Prolog program consists of a set of clauses. Each clause indicates a logical fact or an inference rule. For example, clause C3 in Fig. 5 indicates the fact that "Joe is the father of Mike" and clause C4 shows an inference rule, that is, "If *x* is the father of *y*, then *x* is a parent of *y*." These clauses can be used as a program. For example, clauses C3 to C6 can be viewed as a program about one's parents.

In the ILP framework, examples of the target concept are given as a set of logical facts (e.g., clauses C1 and C2 in Fig. 5). Related background knowledge is also supplied in the form of logical facts (e.g., C3 and C5) and inference rules (e.g., C4 and C6).

ILP generates new clauses, which form, along with the clauses given as background knowledge, a new program about the target concept. For example, ILP generates clause C7 in Fig. 5, which forms a program about one's child with clauses C3 to C6.

The core algorithm of ILP systems is referred to as the covering algorithm (Fig. 6). This algorithm repeatedly adds new clauses, each of which entails some of the given examples of the target concept until all examples have been covered by the added clauses, thus creating a new Prolog program about the target concept.

There are two major approaches to generating new clauses by the covering algorithm. The first approach starts from a general clause that predicts uncovered

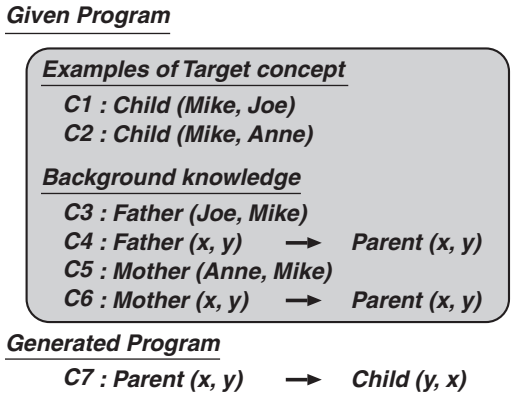


Figure 5 Inductive logic programming.

examples of the target concept (i.e., positive examples) with no preconditions. This first clause may erroneously classify a negative example as a positive one, and it adds a new literal as a test condition so that this new literal can prevent misclassification. Figure 7 shows the idea of this approach.

The following index is an example of criteria used to select new literals with which the new clause will cover a greater range of positive examples:

$$t\left(\log_2 \frac{P_{\mathrm{C+L}}}{P_{\mathrm{C+L}} + N_{\mathrm{C+L}}} - \log_2 \frac{P_{\mathrm{C}}}{P_{\mathrm{C}} + N_{\mathrm{C}}}\right)$$

Here, P_{C} and N_{C} are the number of positive and negative examples covered by the original clause; $P_{\mathrm{C+L}}$ and $N_{\mathrm{C+L}}$ are the number of positive and negative examples covered by the new clause created by adding new literal L to original clause C. The number of positive examples covered by both the original and the new clauses is given by t.

Algorithm *Covering* (E, B)
  Variable E : Examples of target concept
  B : Background knowledge
  C : Learned clause
  Begin
  $C \leftarrow \emptyset$
  $T \leftarrow E$
  While T **do**
  $C_{\mathrm{new}} \leftarrow$ "*Generate new Clause (T, E, B, C)*"
  $T \leftarrow T$ - $T_{Covered\ by\ C_{\mathrm{new}}}$
  $C \leftarrow C + C_{\mathrm{new}}$
  End While
  End

Figure 6 Covering algorithm for ILP.

Algorithm *Generate new Clause (T, E, B, C)*
  Begin
  $C_{\mathrm{new}} \leftarrow$ Clause predicting a non-covered example with no preconditions
  While Negative examples of the target concept satisfy C_{new} **do**
  Add new literal L as a precondition for C_{new}
  End While
  Return C_{new}
  End

Figure 7 Generating candidate clauses.

In the second approach, a so-called "inverse resolution" is used to generate new clauses. Here, resolution is the basic inference operation used in deduction (Fig. 8). Given initial clauses C_a and C_b, the resolution operation first finds literal P in clause C_a such that $\neg P$ occurs in C_b. It then generates new clause C_c as its conclusion by including all literals from C_a and C_b except for P and $\neg P$. Although resolution is applicable to more general cases, we can interpret it as an operation that generates new rule R_c from given rules R_a and R_b.

Inverse resolution generates clause C_a from clauses C_c and C_b. An ILP system uses this operation during the learning process in applying background knowledge, such as C3 and C4 in Fig. 9, to generate new clauses, e.g., C8 and C7.

An ILP system must decide whether a constant in an example should remain the same or should be generalized to a variable. For example, constants "Joe" and "Mike" of clause C1 are generalized to variables "x" and "y" in clause C8 (Fig. 9). These variables give strong data representation to first-order logic and make ILP a promising framework for learning classification rules. However, the use of ILP sometimes requires that care be taken to avoid explosive generation of clauses and variables in the search space.

One research issue in the field of classification rule learning is data representation. For example, ILP can be seen as an extension of DTL in which first-order logic is used to represent data and the covering algorithm is used as a learning algorithm. A learned Prolog program can be used as a classification rule of the

$$\begin{array}{llll} C_a : \neg \mathrm{F} & \vee & \mathrm{P} \\ C_b : \neg \mathrm{P} & \vee & \mathrm{C} \\ \hline C_c : \neg \mathrm{F} & \vee & \mathrm{C} \end{array} \qquad \begin{array}{llll} R_a : If & \mathrm{F} & Then & \mathrm{P} \\ R_b : If & \mathrm{P} & Then & \mathrm{C} \\ \hline R_c : If & \mathrm{F} & Then & \mathrm{C} \end{array}$$

Figure 8 Resolution process.

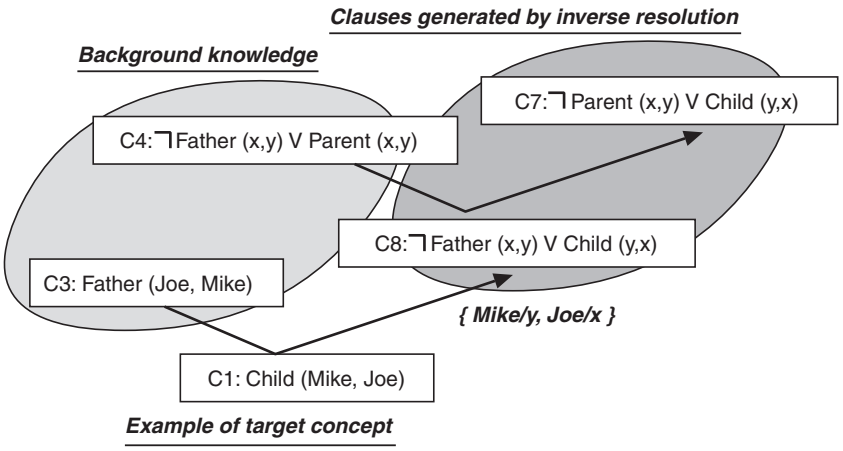


Figure 9 Inverse resolution process.

target concept. Graph-based induction (GBI) is another example of DTL extension. In GBI, each training example is represented as a colored-directed graph and is given to the learning algorithm. Then the algorithm extracts common subgraphs that frequently occur in the given graphs as classification rules. The color of the root nodes represents the class of the training examples. Other nodes represent attributes and their colors represent attribute values. The directed-graph structure represents the relationship between the attributes, and this graphical representation enables the use of nested attributes, which plain attribute-value tables used in DTL cannot represent.

Figure 10 compares DTL, ILP, and GBI. The hypothesis space in these three methods consists of varying structures with symbolic/discrete values. DTL uses an attribute-value table to represent data and hypothesis, whereas ILP uses first-order logic and GBI uses colored-directed graphs. The expressiveness of a graph is somewhere between that of an attribute-value table and that of first-order logic. The potential of GBI as a learning system is thus somewhere between that of DTL and that of ILP. The efficiency of learning is a major problem in ILP, and DTL tends to have the highest efficiency of the three methods.

When applying these methods to practical problems, selecting the data representation method that can hold the essential information of the problem area in the smallest hypothesis space is important. Note that some problems are difficult to represent with symbolic values alone. A set of continuous parameters with a fixed data structure is commonly used in such cases. In the next subsections, we will briefly explain three important methods, i.e., ANN, SVM, and BN, that are based on such representations and that use optimization as a form of learning.

Algorithm *Learning System with a Hypothesis Space that Consists of Variable Structures with Discrete Values*
  Variable C_{in} : Input Data
  T : Classi ers
  Begin
    $T \leftarrow \emptyset$
    Repeat
      Classify All Input Data C_{in} Using T
      Make Attribute Table by *Proc. 1, 2, 3*
      Select New Test Condition, and Add it to T
  End

Proc. 1 ***DTL:*** Always Use **Same** Attribute Set.
Proc. 2 ***ILP:*** Select New Attributes from **First-Order Logic.**
Proc. 3 ***GBI:*** Select New Attributes from **Graph.**

Figure 10 Comparison of DTL, ILP, and GBI.

C. Artificial Neural Networks

An ANN is a pair of a directed graph, G, and a set of functions that are assigned to each node of the graph. An outward-directed edge (out-edge) designates the output of the function from the node and an inward-directed edge (in-edge) designates the input to the function (Fig. 11).

If G does not include a loop, the ANN is called a feed-forward network, and its meaning is then straightforward, i.e., it carries out functional composition. If it includes a loop, we understand the ANN to be either (1) a continuous-time dynamical system or (2) a state machine (a discrete-time dynamical system) by introducing

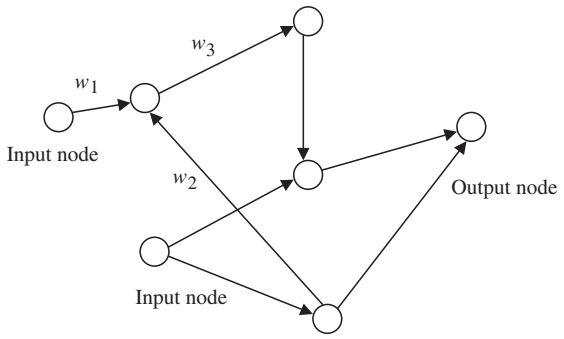


Figure 11 General topology of an ANN.

unit delays to the feedback signals. A Hopfield network and a Boltzman machine represent examples of the former type while a recurrent neural network (RNN) is an example of the latter type of network (Fig. 12).

We now look at a feedforward ANN, since this is the most common type of network. For the functions on nodes, it is common to use $\tau(\Sigma_i w_i x_i)$ where τ is sigmoidal function $\sigma(x) \equiv 1/(1+\exp(-x))$, or radial basis function $\phi(\|x - w\|)$ with, typically, $\phi(t) = \exp(-t^2)$. Here, w_i is a parameter defined at each node, which is often called a connection weight, and x_i is the input from another node or from outside the ANN. Sometimes a higher order polynomial replaces the above $\Sigma_i w_i x_i$, and this is referred to as a higher order network. We usually group nodes that are at the same distance from the input, where the distance is the minimum number of edges from the input to the node, in the same layer. In many cases, connections that skip layers are not used. Nodes that have only out-edges are called input nodes and nodes that have only in-edges are called output nodes. The other nodes are referred to as hidden nodes.

For brevity, let us consider a three-layered network with sigmoidal functions whose outputs are $\sigma(\Sigma_j\ w_{2ij}\ \sigma(\Sigma_k\ w_{3jk} x_k))$. There exists the universal approximation theorem, which states roughly that any smooth function can be approximated as closely as we want as long as we have a sufficiently large number of nodes.

Because ANNs are seldom handcrafted, they are constructed by learning. The objective of the learning process is to approximate unknown target function g. Target function g can be real valued, binary valued, integer valued, or vector valued. The goodness of the approximation is measured by the distance between target g and the ANN's output f. The measure of distance most commonly used is $\|f - g\| \equiv (\int \|f(x) - g(x)\|^2\ dx)^{1/2}$. As is usual in machine learning, we do not have complete knowledge of g. Instead, we are given a finite set of samples $\{(x_k, t_k) | k = 1,\ldots,N\}$ which are pairs of values input into the ANN and those output from the ANN.

To approximate target g, we begin by fixing the network architecture or the underlying directed graph and functions on the node and then find appropriate values for the w_i parameters. Finding a good architecture is difficult and all we have is guidelines to assist us in this task. Fortunately, many experiments have shown that from a few to a few dozen hidden nodes in a three-layered network are enough for relatively simple everyday problems.

To find a good set of weights, we approximate $\|f - g\|^2$ by $E(\mathbf{w}) \equiv (1/N)\Sigma_{k=1}^{N} e_k$ and $e_k \equiv \|f(x_k) - g(x_k)\|^2 = \|f(x_k) - t_k\|^2$ and minimize E with respect to **w,** which is a vector composed of all the connection weights in the ANN.

The simplest technique is the gradient-descent algorithm, which starts from random initial values for w_i and repeatedly uses $w_i \leftarrow w_i - \eta(\partial E/\partial w_i)$ until changes in w_i become small. Here, η is the learning rate. When w_i is a few edges away from the output of the ANN, $\partial E/\partial w_i$ is calculated by using the chain rule.

To be more precise, let us suppose that the output of a depth l unit (where a depth 0 unit is the input to the ANN and a unit is at the depth l if $l - 1$ is the maximum of the depth of the units that are directly fed into the unit) is u_{li} and $u_{0i} = x_i$. The forward

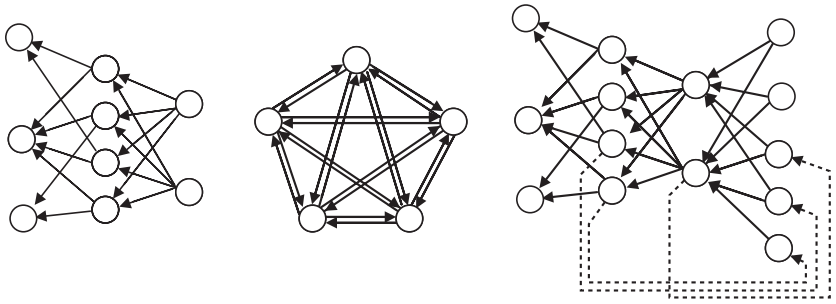


Figure 12 Feedforward, Hopfield, and RNN.

calculation for a depth d feedforward network proceeds as follows:

$$a_{1i} = \sum_j w_{1ij}\, x_i$$
$$a_{l+1\,i} = \sum_j w_{l+1\,ij}\, \sigma(a_{lj})$$
$$f_i(x) = \sigma(a_{di})$$

where the range of i and j varies layer by layer. Supposing that $a_{l+1\,i} \equiv \Sigma_j\, w_{l+1\,ij} u_{lj}$ are stored in the forward calculation, the backward calculation proceeds as follows (Fig. 13):

$$(\partial e_k/\partial a_{di}) = 2(f_i(x_k) \quad t_k)\sigma'(a_{di})$$
$$(\partial e_k/\partial a_{li}) = \sum_j w_{l+1\,ji}\, \sigma'(a_{li})\ (\partial e_k/\partial a_{l+1\,j})$$
$$(\partial e_k/\partial x_i) = \sum_j w_{1ji}\ (\partial e_k/\partial a_{1j})$$

Now $\partial e_k/\partial w_{lij} = \sigma a_{l\,1j}(\partial e_k/\partial a_{li})$ by which we get $\partial E/\partial w_{lij}$. Since, in the expression, the error information flows from output to input, this method is referred to as error backpropagation or, simply, backpropagation.

Backpropagation is not guaranteed to converge to an optimum value unless it has started close to an optimum value, although many experiments have shown that it will, in other cases, converge to a good suboptimum or else it may diverge.

The on-line mode of learning refers to the use of $e_k \equiv (f(x_k) - t_k)^2$ rather than $E = (1/N)\Sigma_{k=1}^{N}\, e_k$ as described in the above, which is called a batch mode. An ANN in the on-line mode less frequently becomes caught up in suboptimal regions than an ANN in the batch mode.

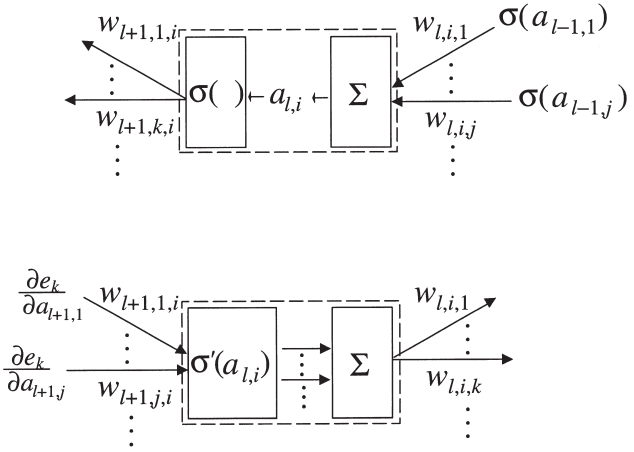


Figure 13 Forward (top) and backward (bottom) calculations.

Backpropagation is an effective algorithm in many applications, although it is slow. Many methods for speeding it up have been developed. One class of methods adds accelerating terms to the update formula, e.g., $w_i \leftarrow w_i - \eta\ (\partial E/\partial w_i) + \alpha\delta w_i$ where $\delta w_i \equiv$ current $w_i -$ previous w_i and $0 < \alpha < 1$ is an acceleration coefficient. Although this is a simple approach, it greatly accelerates convergence. In another class of methods, second-order derivatives are included in the update formula. This is computationally expensive, because we need to carry out matrix inversions or equivalent computation but since the number of times parameters are updated is small, the total computation time is usually shorter than in backpropagation.

Cross-validation is used to prevent overlearning. The samples are split into two groups, a training set and a validation set. The former is used for learning while the latter is used for testing or validation. We monitor validation errors during learning by calculating outputs and errors for the validation set and stop the updating of parameters when they have been confirmed to have reached their lowest point. Cross-validation is used to determine the number of hidden units, too. The greater the number of hidden units, the more vulnerable the algorithm is to overlearning. We can find the best number of hidden units by monitoring validation errors when the number of hidden units is being increased.

The original idea of ANN came from the study of the nervous systems of animals. Such systems are composed of around 10^8 to 10^{11} neurons and the systems learn or are trained after the animal's birth.

There are many fields of application for ANNs, because in real life there are many cases in which the functional form of the input/output relations is unknown, or does not exist, but we still want to approximate that function. Practical applications include the sensing and control of household appliances and toys, investment analysis, the detection of credit card fraud, signature analysis, process control, and others.

D. Support Vector Machine

The support-vector machine (SVM) classifies new $\mathbf{x}$ according to samples $\mathbf{x}_1, \ldots, \mathbf{x}_n$ with class labels $y_1, \ldots, y_n \in \{\pm 1\}$ by

$$y = \text{sgn}\left[\sum_{i \in S_n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) - h\right]$$

where sgn is the sign function, which outputs 1 for a positive input and -1 for other inputs, $S_n = \{i|\alpha_i \neq 0\}$, and K is the kernel function for which

$$K(\mathbf{x},\mathbf{y}) = \tanh(a\mathbf{x} \cdot \mathbf{y} - b) \quad \text{or}$$

$$K(\mathbf{x},\mathbf{y}) = \exp(-||x - y||^2/\sigma^2)$$

is often used. In these cases, the functional form of the SVM is the same as that of a three-layered feed-forward ANN, but the learning algorithm is different.

The α_i's are determined by solving a quadratic optimization problem: maximize

$$W(\alpha) \equiv \Sigma_{i=1}^{n}\alpha_i - (1/2)\Sigma_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

subject to:

$$\alpha_i \geq 0,\ i = 1, \ldots, n \quad \text{and} \quad \Sigma_{i=1}^{n}\alpha_i y_i = 0$$

The idea behind this is as follows. Using the property of kernel function $K(x,y) = \mathbf{z}(\mathbf{x}) \cdot \mathbf{z}(\mathbf{y})$ for some function z (called a kernel trick), the above problem is the dual of a quadratic optimization problem: minimize

$$L(\mathbf{w}) \equiv ||\mathbf{z}(\mathbf{w})||^2 \quad \text{subject to}$$
$$y_i(\mathbf{z}(\mathbf{w}) \cdot \mathbf{z}(\mathbf{x}_i) - h) \geq 1,\ i = 1, \ldots, n$$

A solution to this last problem is hyperplane $\mathbf{z}(\mathbf{w}) \cdot \mathbf{z}(\mathbf{x}) = h$ in $\mathbf{z}$ space that separates the two classes ($y_i = \pm 1$) by a maximum margin, which is the distance from the nearest $\mathbf{z}(\mathbf{x}_i)$ to the hyperplane. Point $\mathbf{z}(\mathbf{x}_i)$ with nonzero α_i is on hyperplane $\mathbf{z}(\mathbf{w}) \cdot \mathbf{z}(\mathbf{x}) = h \pm 1$ and is called a support vector.

In the SVM method, input vectors $\mathbf{x}_i$ are mapped to a higher dimensional (often infinite-dimensional) feature space $\mathbf{z}(\mathbf{x}_i)$, which is implicit in the sense that we do not need to calculate $\mathbf{z}(\mathbf{x}_i)$. The hyperplane that separates the two classes by the largest margin is then found. The generalization capability is guaranteed by the probably approximately correct (PAC) learning paradigm.

E. Bayesian Networks

A Bayesian network (BN) is a pair of a directed graph, G, and a (conditional) probability distribution of a random variable defined on each node of the graph. If node (or the random variable on a node) A has a direct inward-directed edge (in-edge) from nodes B_1 and B_2 but not from other nodes, it is assigned $P(A|B_1,B_2)$; if A has no in-edges, it is assigned $P(A)$, where $P(\cdot)$ is a probability distribution and $P(\cdot|\cdot)$ is a conditional probability distribution.

A lack of edges between nodes represents conditional independence between the corresponding random variables. Node A having a direct inward connection from nodes B_1,B_2 but not from nodes $B_3, \ldots, B_k$, means that $P(A, B_1, B_2, \ldots, B_k) = P(A|B_1,B_2)P(B_1,B_2, \ldots, B_k)$, although in general $P(A,B_1,B_2, \ldots, B_k) = P(A|B_1,B_2, \ldots, B_k)P(B_1,B_2, \ldots, B_k)$. One may say that a directed connection from node A to B means that A may affect B. Each random variable may be continuous or discrete, but the actual random variables in applications are usually discrete and designate some event or attribute.

Figure 14 is an illustrative example. The W in the figure refers to the state of grass and it depends on whether the state of S is on or off and on whether the state of R is true or false; and both of these variables depend, in turn, on whether the state of C is true or false.

When the events are discrete, the probability distribution functions assign probabilities to the events. In this case, a table containing a probability distribution is called a conditional probability table (CPT). For example, in Fig. 14, the tables are CPTs; and "cloudy is true" has probability 0.5, "cloudy is false" has probability 0.5, and "sprinkler is on" has conditional probability 0.1 if "cloudy is true."

The goal of a BN is probabilistic inference. Given some evidence (or occurrences of events on some nodes), we can calculate the probability of unseen events. Many algorithms such as message-passing and the junction-tree have been invented to calculate it even faster.

If "causes" are presented as evidence, a form of prediction is provided. Upon receiving the evidence, we calculate the margins of the unseen events. A naive way to do so, for example in Fig. 14, is to prepare tables of all the probabilities, $P(W,S,R,C)$ and $P(W|C) = \Sigma_{s,r}\ P(W,s,r,|C) = \Sigma_{s,r}\ P(W,s,r,C)/\Sigma_{w,s,r}\ P(w,s,r,C)$. This is unrealistic since we have to keep a table with

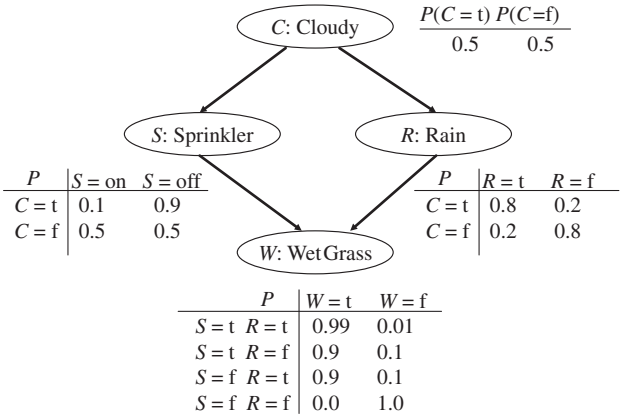


Figure 14 WetGrass: An example of a BN.

2^n entries for n binary variables. The BN tells us that calculating $P(W|C) = \Sigma_{s,r} P(W|s,r)P(s|C)P(r|C)$ on the fly is sufficient and fast. The reduction in the example is small but in general it is substantial.

If "effects" are presented as the evidence, a form of diagnosis is provided. Since the causal relations are probabilistic the diagnosis is in probabilistic terms such as "cloudy is true with probability 0.7" or "cloudy is true is most likely." Calculation is done by repeatedly applying the Bayes rule. Preparing all the probabilities $P(W,S,R,C)$ and calculating in a way such as $P(C|W) = P(C,W)/P(W) = \Sigma_{r,s} P(W,r,s,C)/\Sigma_{r,s,c} P(W,r,s,c)$ is unrealistic. By utilizing the independence relations in the BN, BN methods calculate $\Sigma_{r,s} P(C|r,s)P(r|W)P(s|W)P(C)/P(W)$ on the fly with a reduction of computational cost, especially when the number of nodes and that of independent pairs of nodes are large. The most probable configuration of events can also be calculated when the evidence is not restricted to ultimate causes (root nodes of the BN) or ultimate effects (leaves of the BN).

Let us now turn to BN learning problems. There are several types of such frameworks. The network structure may be known or unknown, and variables may be observable or nonobservable. As would be expected, a fixed structure with only observable variables is the most simple case. The most interesting case is, however, a fixed structure with unobservable variables, since we, in many cases, have intuitive knowledge of the core of relations of influence or of causal relations among events that are not necessarily observable. Examples of such cases are the fault trees of engines and medical diagnoses.

For a known structure and full observability, we find the maximum likelihood estimates (MLEs) of the entries in the CPT by maximizing, for example,

$$(1/N) \log \Pi_{j=1}^{N} P(d_j|G) = (1/N)\Sigma_{i=1}^{n} \Sigma_{j=1}^{N} \log P(X_i|\pi(X_i),d_j)$$

where G is the underlying directed graph, $\{d_1, \ldots, d_N\}$ are examples of what has happened, $X_1, \ldots, X_n$ are random variables, and $\pi(X)$ designates the parents of node X. For example, in Fig. 14, the MLE estimate of conditional probability $P_{\mathrm{MLE}}(w|s,r)$, $\hat{P}_{\mathrm{MLE}}(w|s,r) = \#(W = w, S = s, R = r)/\#(S = s, R = r)$ where $\#(S = s, R = r) = \#(W = 0, S = s, R = r) + \#(W = 1, S = s, R = r)$ in the case of a multinomial distribution, where, for example, $\#(S = s, R = r)$ is the number of observed events such that $S = s$, $R = r$.

For a known structure and partial observability, we use the EM (expectation maximization) algorithm to find a possible MLE. The idea is to start from random initialization of the unobserved value, then repeat the following: (1) Obtain new $\hat{P}_{\mathrm{MLE}}$ that is based on the samples supplemented with current expectations of occurrences of the unobserved nodes (M step), and (2) obtain new expectations of occurrences of the unobserved nodes on the basis of estimated $\hat{P}_{\mathrm{MLE}}$ (E step).

For an unknown structure, we have to have some prior knowledge of the directed graphs to avoid overfitting, since a complete graph will achieve the highest likelihood among the graphs with the same number of nodes and will be easily overfitting.

II. OTHER TOPICS

A. Reinforcement Learning

Reinforcement learning is a framework for finding the best policy that specifies how you act next and that maximizes the total reward you will receive. You do not have a teacher who tells you what the correct action would be but you do have an environment from which you get (possibly delayed) rewards.

Suppose that you are a two-legged robot learning how to walk. The goal you must achieve is to be able to walk around, without falling over, for as long as possible; that is, to maximize the period during which you can walk.

The RL framework for the above problem is as follows. The robot is in an environment. It decides its next action on the basis of sensed information about the environment and its own current state and state history. When the robot is to walk, the environment is the position of the robot's center of mass, the configuration of its legs and joints, the slope of the ground, etc. When the robot performs an action (say, moves its center of mass 1 cm forward), it gets a result; that is, a new state for its environment, including its own state, and a reward of some kind. The reward might be something that the robot gets when it achieves the goal or something the robot must pay as a penalty (for example, energy consumption) to move the center of mass. The reward can be delayed until the final achievement or it can be received little by little, in a way similar to the joy and appreciation received by an infant when it walks one step.

The transition from state s_t at time t to s_{t+1} is probabilistic (a deterministic transition is a special case of a probabilistic transition) and is specified by probabilities $P^a_{ss'} \equiv P(s_{t+1} = s'|s_t = s, a_t = a)$ where a_t is the action taken at t. The expected immediate reward is $R^a_{ss'} \equiv E\{r_{t+1}|s_t = s, a_t = a\}$.

Policy $\pi(s,a)$ specifies an action to be taken in state s as probability distribution $P(a_t = a|s_t = s)$. The return is the sum of the rewards or the cumulative fu-

ture discounted reward: $R_t \equiv r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{T-t-1} r_T$ where T is the final time step, which could be infinite, and $0 \leq \gamma \leq 1$ is a parameter called the *discount rate*. Here, γ is used for a continual case when $T = \infty$.

Value function $V^\pi(s)$ is defined for state s and policy π and specifies the expected return under the policy:

$$V^\pi(s) \equiv E_\pi\{r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s\}$$
$$= \sum_a \pi(s,a) \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V^\pi(s')]$$

If we know a value function that is optimal among all the policies,

$$V^*(s) \equiv \max_a E\{r_t+1 + \gamma V^*(s_{t+1}) | s_t = s, a_t = a\}$$
$$= \max_a \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V^*(s')]$$

then the action we are to take at s is the a that is the argument on the right-hand side of the equivalence. This means that the optimal policy is implicit in $V^*(s)$.

Value iteration is a method for obtaining V^* by repeating $V_{k+1}(s) \leftarrow \max_a \Sigma_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V_k(s')]$. This works well but has a drawback in that we have to have complete knowledge of the environment to do the calculation even for one step. The following method, Q learning, does not require this.

A Q function is defined for s and action a and specifies the expected return after action a has been taken:

$$Q^\pi(s,a) \equiv E\{r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s, a_t = a\}$$
$$= \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V^\pi(s')]$$

As for V^*, if we know an optimal Q function

$$Q^*(s,a) \equiv E\{r_{t+1} + \gamma \max_{a'} Q^*(s_{t+1},a') | s_t = s, a_t = a\}$$
$$= \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma \max_{a'} Q^*(s',a')]$$

then we are to take a such that $Q^*(s,a)$ is at its maximum at s. This means that the optimal policy is implicit in $Q^*(s,a)$.

Q learning is a method for obtaining $Q^*(s,a)$ by repeating $Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha(r_{t+1} + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t))$, where α is the learning rate. The above update is done for each step (i.e., for each action taken under a policy derived from current Q) for each episode (one course of action that leads to a final state). Any policy can be selected as long as it guarantees the convergence of the above repetitive updates to the optimal Q function, if all of the pairs of states and actions are updated infinitely many times (and $\Sigma_{t=0}^\infty \alpha(t) \rightarrow \infty$ and $\Sigma_{t=0}^\infty \alpha(t)^2 < \infty$). For efficiency, however, ϵ-greedy policy can be used, where the action is chosen randomly with probability ϵ or it is chosen as the action with the largest Q.

Q learning is a TD (temporal difference) method. A TD method estimates and uses $V(s_t) \leftarrow V(s_t) + \alpha(r_{t+1} + \gamma V(s_{t+1}) - V(s_t))$ instead of more straightforward estimate $V(s_t) \leftarrow V(s_t) + \alpha(R_t - V(s_t))$ and the name was thus coined. With the latter method we have to wait until a final state is reached to calculate updates of V but with the former method we can calculate the update after each step. A drawback of Q learning is that it takes a long time for the effects of the reward we get at the final state to propagate through the states and settle down and as a result, the convergence of Q values (and in the same manner V values) is slow. TD(λ) is a solution to this problem. For lack of space, TD(λ) is not explained here.

The above-mentioned and other techniques find many applications in the real world, ranging from robot control and planning to games such as backgammon, along with theoretical applications, e.g., simulating cerebellar functions.

B. Other Methods

We briefly describe meta-learning techniques (e.g., boosting, bagging, and stacking), explanation-based learning, instance-based learning (e.g., k-nearest neighbor and case-based reasoning), clustering, association rule discovery, and genetic algorithms in this section.

Boosting, bagging, and stacking methods were developed to implement the idea that we can construct an arbitrarily accurate "strong" learning algorithm by combining "weak" learning algorithms that perform just slightly better than random guessing. The idea comes from probably approximately correct (PAC) learning, a theoretical framework of learning.

Figure 15 shows the boosting algorithm. Boosting uses given "weak" learning algorithms to repeatedly construct classifier c_t. At each step t, the given learning algorithm assumes distribution D_i over training examples E_i to construct classifier c_t, and the boosting algorithm increases weight D_i of misclassified examples E_i so that the following weak learning algorithms could concentrate on these examples to increase the accuracy on the whole.

Explanation-based learning (EBL) starts with just a single training example. It generates an explanation of why the training example exists by using deductive inference based on the background knowledge it has. It then generalizes the explanation to be used in the future. It is very efficient compared to other methods

```
Algorithm Boosting
  Variable  c_t : Classifier generated by weak learner
            E_i : N Examples
            D_i : Weight of E_i
  Begin
    Equally initialize D_i = 1/N
    For t = 1,,,T do
      Make c_t by weak learner using distribution D
      Update distribution D with c_t:
          Increase weight D_i of
                incorrectly-classified example E_i
          Decrease weight D_i of
                correctly-classified example E_i
    Merge c_1,,,c_T and form final classifier
  End
```

Figure 15 Boosting algorithm.

in terms of the number of training examples needed. It requires, though, complete background knowledge.

In contrast to the model-based methods so far explained, instance-based methods such as *k*-nearest neighbor (*k*-NN) and case-based reasoning simply store examples without constructing generalized hypothesis and use the stored data directly when they encounter new data. The *k*-NN method classifies a newly encountered example based on the *k* past experiences closest to the new example. Here the similarity is measured by a distance function on the data attributes. The majority class of selected *k* past data is used as the class of newly encountered data.

A priori, a representative algorithm for association rule discovery, finds all the rules "if A is in S, then C is in S, too" with confidence and support above the predefined minimum, where each example is a subset of item set S, $A \subset S$, $C \subset S$, and $A \cap C = \phi$. Here confidence is the number of examples that include A in the given data set. Support is the ratio of the number of examples that include both A and C to the number of examples that include A. A priori is very fast and effective in finding valuable rules in large databases.

Clustering is a task to structure a set of training examples based on the similarities found in the examples. Because the target category of examples is usually not given, unsupervised learning methods are used. Any clustering algorithm takes into account the trade-offs between similarities and dissimilarities among members of clusters. If the number of categories is smaller than adequate, then the similarities in the same category become small. If it is larger than adequate, the dissimilarities between different categories become small.

Genetic algorithms are optimization methods characterized by the use of many genotypes (compact representations of a target hypothesis), the survival-of-the-fittest genotypes (the hypotheses that give better values), and by random generation of offsprings by mutation (partial change of genotypes) and recombination (merging and splitting of genotype pairs). They are used as search methods in machine learning when the hypothesis space is large and complex.

C. Applications

The importance of machine learning techniques in practical situations has increased as a result of the recent improvement of accessibility to huge data sets and the rapid development of high-speed computing and global networks. The use of various machine learning techniques for a variety of new applications has become an actively studied research field. Associative rule discovery for mining from databases is one example of this new research field. Bioinformatics also needs new and revolutionary methods. Search engines for the WWW were first developed as an engineering solution but soon became a new field, and research in this field has been rapidly growing ever since. Text mining will also be an important field for the WWW and knowledge systems in the 21st century.

SEE ALSO THE FOLLOWING ARTICLES

Automata Theory • Cybernetics • Engineering, Artificial Intelligence in • Expert Systems Construction • Goal Programming • Intelligent Agents • Knowledge Management • Neural Networks • Robotics • Search Engines • Search Techniques

BIBLIOGRAPHY

Cristianini, N., and Shawe-Taylor, J. (2000). *An introduction to support vector machines.* Cambridge, UK: Cambridge University Press.

Freund, Y., and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences,* Vol. 55, No. 1, 119–139.

Mitchell, T. M. (1997). *Machine learning.* New York: McGraw-Hill.

Muggleton, S., and Feng, C. (1992). Efficient induction of logic programs. *Inductive logic programming,* S. Muggleton (Ed.). San Diego: Academic Press.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems.* San Francisco: Morgan Kaufmann.

Reed, R. D., and Marks, R. J., II (1999). *Neural smithing: Supervised learning in feedforward artificial neural networks.* Cambridge, MA: The MIT Press.

Sutton, R. S., and Barto, A. G. (1998). *Reinforcement learning.* Cambridge, MA: The MIT Press.

Management Information Systems

William R. King
University of Pittsburgh

GLOSSARY

business application A specific business task, decision, or problem that has been modeled and computerized.
business process reengineering The redesign of a cross-functional set of business activities that are intended to serve customers; almost always involves the inclusion of higher levels of computer technology.
chief information officer The executive who has primary responsibility for the operation of the MIS organizational function and for planning for its future.
data Symbols, either numerical or textual, that have not yet been evaluated to be useful for some purpose.
effectiveness The degree to which the level of performance conforms to a preestablished goal.
efficiency The unit output per unit of input from an activity or process.
explicit knowledge Knowledge that is "written down" in some form, such as in patents and research reports.
information Data that have been evaluated to be useful and which are therefore acquired, stored, transformed, and disseminated through management information systems.
stakeholder An individual or group that has a "stake" in the performance of the enterprise (e.g., stockholders, employees, customers, etc.).
tacit knowledge A justified personal belief that is based on training and experience.

MANAGEMENT INFORMATION SYSTEMS (MIS) is a field of practice, study, and research that focuses on how computer and communications technology can be more efficiently and effectively employed by organizations and individuals to achieve their organizational, group, and personal goals.

In the case of a business firm, these goals may involve increasing profits by conducting analyses of the relative profitability of various products and markets. In the case of a government or public service organization, the goals may involve the design of policies and programs by obtaining and analyzing demographic data on groups of people who may be differentially impacted by a proposed program. Group or team goal pursuit might employ collaborative work systems or electronic workspaces to enable effective and efficient computer-supported cooperative work. Personal goals such as career planning may be pursued by individuals who are able to use their firm's intranet or the Internet to obtain relevant data and to analyze the consequences of choices among job opportunities or education/training programs.

I. THE EMERGENCE OF MIS

The MIS area emerged in the early days of the electronic digital computer era. Prior to the emergence of MIS, organizations primarily focused on using computers to perform tasks, activities, and processes that had previously been done manually, e.g., invoicing, the creation of financial reports, and order processing. The business department that operated the computers and performed these automation projects was then generally referred to as electronic data processing (EDP).

When it was recognized that the data that were being processed could also be useful to managers in making their business decisions, new varieties of computer applications began to be developed. These early applications and systems focused on the summarizing and analyzing of the data that had been collected and processed. Such summaries initially took the form of month-to-month and year-to-year comparative analyses, cross-classification analyses such as sales categorized by product and by region, and trend analyses which could provide the bases for forecasts.

These analyses rapidly became more sophisticated, and the term "management information systems" was created to describe both the new focus of computer systems and the broader scope of the organization's information processing function. This shift, from a focus on data to a concentration on information, and from a focus on processing to a concentration on management support applications and systems, had a profound impact on both theory and practice.

The singular form of the term, "management information system," thereby refers to a system with a particular objective—that of supporting managerial analyses, decisions, and actions. The plural form, "management information systems," obviously relates to a collection of such systems, but more importantly describes an organizational function that is responsible for developing, operating, maintaining, and improving an organization's computer systems.

The field has subsequently broadened further to incorporate supporting a wide variety of activities such as the user development of computer applications, the maintenance and improvement of the organization's expertise in employing computer and communication technologies, the management of telecommunications, the execution and management of business process re-engineering, and the development and implementation of a wide variety of new systems such as executive information systems, group support systems, and systems for electronic commerce.

Since the individuals and activities being supported are diverse and can no longer be described solely as "management," in some settings the MIS terminology has been replaced by the more generic term information systems (IS). In some organizations, the department is called Information Technology or Information Services or some other similar term. However, the term MIS remains in widespread use because it best describes the combination of management skills and technology skills which are essential to providing computer and communications support in an organizational context.

II. COMPARING MIS WITH COMPUTER AND INFORMATION SCIENCE

MIS encompasses ideas and viewpoints that importantly reflect technical, behavioral, and organizational considerations. This explains why the MIS field is usually taught and studied in a business school environment. However, since the term, information systems, which is sometimes viewed to be synonymous with MIS, is broad in scope, it also may be validly claimed by others who may be based in computer science or in information science areas.

Computer science and information science approaches to IS are generally different from MIS as it is taught and practiced in business schools and in business.

Computer and information science programs primarily focus on the operational effectiveness and efficiency of computer-based systems. On the other hand, MIS focuses primarily on systems and processes for the solution of organizational problems and the support of decision making, with the fulfillment of organizational objectives as the ultimate goal. Thus, in a business context, MIS is concerned with business applications systems and with the design, development, and implementation of computer-based systems that are an integral element of the organization's capabilities to pursue its business objectives.

Perhaps the best way to explain this distinction is to describe the sorts of tasks that are usually performed by graduates of an MIS program in a business school and contrast those with the sorts of tasks that are usually performed by graduates of other programs.

Tasks such as business applications development, business systems analysis, business process reengineering, project management, strategic IS planning, and the management of MIS departments are generally performed by business MIS graduates, while graduates of more technically focused programs in computer science or information science might develop computer operating systems, develop database products for vendors, develop systems for the retrieval of information from libraries, and perform technical or network support, database administration, or programming.

In practice, the demarcation line is not completely clear and distinct, since persons trained in nonbusiness programs often move up the career ladder to take on project leadership and assume other managerial roles; but, doing so may often require them to attend a business school to learn more about the management, behavioral, and organizational perspectives that are essential to be effective in the MIS field.

MIS competency clearly requires an understanding of organizational and human behavior in addition to technical knowledge, since MIS involves developing the human and organizational interfaces with computer systems to ensure that individuals, groups, and organizations are able to use such systems effectively and efficiently. For instance, the MIS graduate might study a business process such as order fulfillment to learn why each step is performed, what data are required to perform each step, and what information each step produces. Then, he might redesign the process so that costs or cycle time can be reduced or customer satisfaction can be enhanced. This would require assessments of the role of the process in fulfilling the overall goals of the business as well as its relationship to other business processes. He might also develop the specifications for a computer-based system to support the improved process design. In doing so, he would generally involve the business process participants in design and development activities to obtain their ideas as well as their support for the new process and he would be concerned with creating training programs that would enhance the likelihood of successful implementation and use of the new process and system.

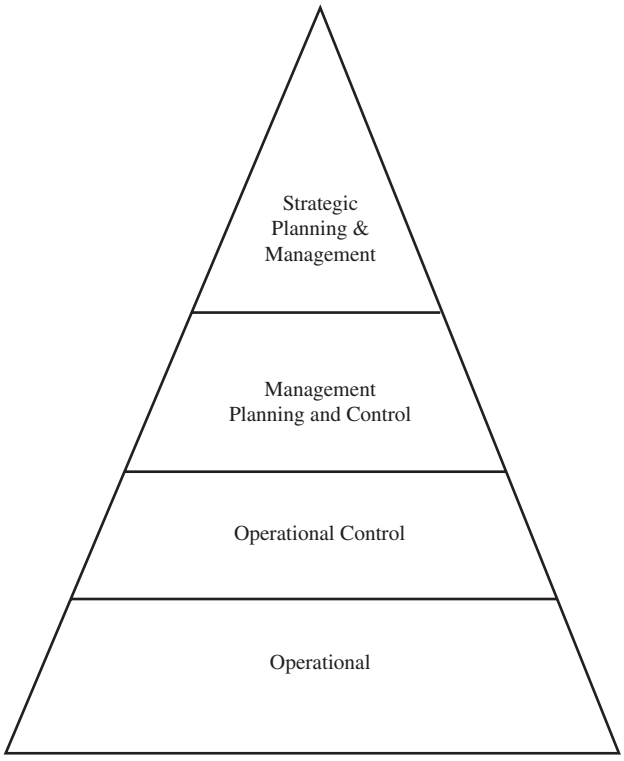


Figure 1 Hierarchy of organizational activities that are supported by MIS.

III. THE HIERARCHY OF MIS SUPPORT IN THE ORGANIZATION

MIS activities occur throughout all functions of the organization like administrative, marketing, production, financial, human resources, etc., as well as between the organization and its external stakeholders like suppliers, customers, distributors, etc. These activities take place at several different levels as described by Fig. 1. The triangle in that figure has the operational level of the enterprise as the foundation and operational control, management planning and control, and strategic planning and management, respectively, at successively higher levels. The triangular shape suggests that each level guides and controls the lower ones.

A. The Operational Level

This is the level at which computer systems are used to perform the organization's basic activities and transactions. Illustrative of this level are the processing of orders, arranging for the manufacture of products, the shipping of finished goods, the delivery of business services, and the processing of financial transactions. Computer systems—such as transaction processing systems, electronic commerce systems, and accounting systems—that function at this operational level are performing the basic work and recordkeeping of the organization. Increasingly, such systems involve linkages to other organizations, such as to suppliers and customers, since it is important that deliveries be carefully scheduled and that information on the status of orders be available to both the buying and selling organization. Systems that involve the Internet are also prevalent at this level. Many firms use such systems to purchase through Internet auctions or through suppliers' web sites. Many firms sell and/or promote their products on the Internet as well.

B. The Operational Control Level

At this level, information systems ensure that the operations are performing within the guidelines and parameters that have been established for them. These parameters might reflect elements that are primarily

technical, such as error rates or conformance to dimensional tolerances, or they might reflect managerial issues such as those related to controlling warranty costs or the organization's response times for responding to customer inquiries.

Often, computerized operational control systems operate on an "exception reporting" basis, i.e., they are programmed to highlight parameters whose values lie outside some prescribed range. For example, such systems might indicate when parts are being produced that do not fit within dimensional tolerances or when the level of accounts receivable has grown to some level that it should be of concern to management.

At this operational control level, routine evaluations of operational activities are also performed, for instance, reports are prepared daily, weekly, or monthly on the conformance of various activities to their budgets and on the quality levels being achieved. These evaluations usually involve comparisons over time, such as month-to-month, or comparisons with comparable indices from past years. They may also involve the use of standards that are developed from external sources such as the use of "benchmarking" data from other companies that are known to be high performers in some business process or area of operations and/or the use of standards developed from statistical data supplied by trade associations or information vendors.

C. The Management Planning and Control Level

This is the level at which information and systems are provided to managers to help them in judging and improving the business value of organizational activities. For instance, at this level issues are addressed such as: Is a particular product line profitable? Is a new production facility likely to be necessary? Might this product be saleable in other markets? With such information, managers can plan for changes in activities and practices to add value to the organization.

Information systems are developed to support the making of these judgements in various ways ranging from analytical cost-benefit models to group decision support systems in which the judgments of individuals can be shared within a group in order to judgmentally evaluate alternative strategies. Executive Information Systems (EIS) often operate primarily at this level since a major focus of many EIS is the providing of detailed information on the status of ongoing activities to top management when they make inquiries.

D. The Strategic Planning and Management Level

This is the level at which new organizational activities are conceptualized and evaluated as well as the level at which the organization's overall portfolio of activities is evaluated and periodically altered. For instance, at this level the issues addressed are: Should we adopt a strategy of developing a continuing stream of new products? Should we acquire or merge with other firms? What new business capabilities should we develop?

At this level, computerized planning systems support strategic planning. Most of the systems that support decision making at the management planning and control level may be used at the strategic level as well. Since most decisions at the strategic level are primarily the purview of the judgment of top business managers, group support systems that allow groups of managers and planners to jointly address complex issues are often especially useful. However, this is the level at which computerized support is the least well developed and implemented.

IV. THE WEB OF MIS RELATIONSHIPS

The role of MIS is in providing *information, systems,* and *services* at each of the hierarchical levels of Fig. 1 as well as to stakeholders that are external to the legally defined enterprise, e.g., customers, shareholders, suppliers, and government agencies. *Information* is provided to MIS clients in the form of routine reports or responses to inquiries, and by facilitating communications through electronic mail, group support systems, corporate intranets, and other systems. *Systems* are provided to MIS clients to address new applications in all areas of the enterprise, including those that service external stakeholders. *Services* are provided to MIS clients who are individual system users, or to groups of users, to enable them to use relevant systems to perform their own computing or to play a role in systems project teams that are developing systems.

These MIS outputs—information, systems, and services—support individuals, groups, organizational subunits, the overall enterprise, and some external stakeholders in *making decisions, integrating,* and *coordinating* their work with the work of others and providing services to their own clients within and outside of the enterprise. For instance, customer service personnel use information and systems to make decisions concerning responses to customer requests. They coordinate with warranty specialists in doing so and they

provide services to customers such as giving approval for warranty-covered product repairs. Similarly, human resource specialists are clients of MIS who use information and systems to serve their own internal clients to better plan for their training and career progression.

These MIS outputs are provided by conceiving, designing, developing, implementing, and operating computer-based systems and in providing support to non-IS people and departments to ensure that the organization's systems perform effectively (that they serve their intended organizational purposes) and efficiently (that they are productive and as low in cost as is practicable). MIS personnel operate as systems analysts and designers as well as in such leadership positions as project managers in the development of such systems. They also operate and administer the MIS Department, which typically performs a wide variety of user support, training, and other functions throughout the enterprise, while maintaining and improving the enterprises' MIS expertise.

Figure 2 shows this web of relationships in terms of the "products" provided by MIS, the clients of MIS, the client activities that are enabled and supported, and the overall impact of enabling clients to perform their mission and achieve their goals.

V. THE EVOLUTION OF MIS

To understand MIS, one must have an understanding of the significant changes that have characterized the MIS era and which have therefore determined the current nature and scope of the MIS function.

A. From Data to Information to Knowledge

One fundamental change that has occurred has been in terms of the content focus of MIS—from a focus on data, to a focus on information and then, to a focus on knowledge.

Data are symbols that represent attributes of something that is of interest. For instance, sales data are numbers, usually in terms of quantities or revenues, that describe an important aspect of a business' performance. Information is data that have been evaluated for some purpose. In other words, information is of higher value than data. The process of converting data to information occurs at the most basic level when an organization decides that some data are relevant to them and therefore that it should be acquired and stored. The process of enhancing the value of information continues, for instance, when sales data are cross tabulated by product and region, thereby transforming data into information that can serve as a basis for sales, production, and distribution planning. Knowledge is less tangible, but potentially more important, than information. It is reflected in the understanding of cause and effect relationships or in the specification of a "best practice" to use under a given circumstance.

In the EDP era, the focus of organizational computing was on capturing and processing data. The MIS era changed the focus to information that could enable managers to better perform their jobs. The most recent content shift has been to focus on knowledge in various forms ranging from explicit knowledge such as that contained in research reports and patents to the tacit knowledge that exists in the minds of experienced professionals or which is embedded in the business firm's products, relationships with customers, and suppliers and business processes. This is the purview of the area that is referred to as "knowledge management," which may be thought of either as an element of MIS or as a closely allied function. Knowledge management systems range from expert networks which can be used to identify individuals in the organization who possess a specified professional expertise to best practices repositories that enable one unit of an organization to share its knowledge with other units.

B. From Transaction Processing to Strategic Management

A second major change in MIS has been that from focusing on operational business objectives, to focusing on management planning and control objectives, to focusing on strategic business objectives. Thus, MIS has "worked its way up" the triangle in Fig. 1.

The earliest focus of computers in organizations was the *automation* of activities that were previously performed manually. As the MIS era began, the focus shifted to *enabling* managers to improve their decisions by performing analyses that had previously been difficult, if not impossible, to do. As well, in this phase, computer systems were developed to *enhance* the business' processes in order to create more value for the firm.

More recently, the focus has been on business *redefinition* through the use of electronic commerce to complement or replace traditional business models and the use of computer systems to create new sources of revenue and new possibilities for growth, such as through the sale of the customer purchase data that

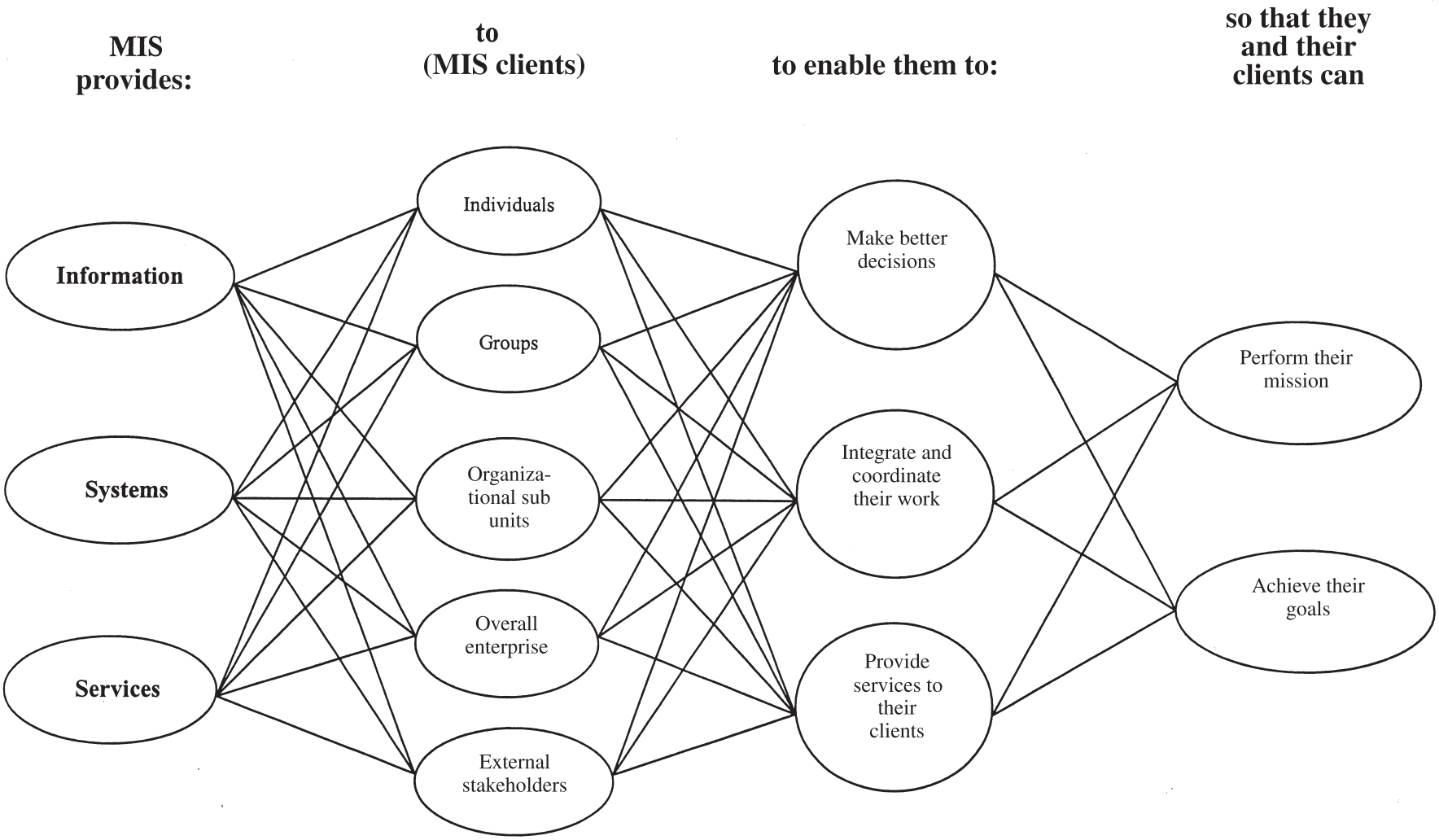


Figure 2 The web of MIS relationships.

are collected by the scanners in the supermarket check-out process.

C. From a Service Culture to Competitive Advantage

In the days of EDP and the early MIS era, the department provided back-office services to the other departments of the organization. Computers were located in "glass houses;" computer people were thought of as "geeks" who spoke in acronym-laced jargon and understood little about the real business of the enterprise.

Soon, it became apparent that computer systems were not like the electrical power system or the heating and air conditioning system. They could indeed provide a basic generic service as do these systems, but they had the potential to do much more. So, MIS came out of the glass houses and began to train their people about business rather than just about computer systems.

The result has been an almost complete integration of MIS with the business. Now, MIS people and business people work together to create systems that will provide sustainable competitive advantage to the business firm by creating new relationships with customers, new sources of revenue, and new markets.

D. From Mainframes to PCs to Networks

Technology trends have driven changes in MIS as well. In the mainframe computer era, MIS was a centralized function that utilized MIS professionals to develop computer applications for "users" in the organization.

With the advent of PCs, the overall MIS function became much more decentralized, although the centralized mainframe systems usually remained in place. This created the era of end-user computing (EUC) in which the development of some new computer applications could be performed by individuals who were outside the MIS organization.

The more recent networking of computers within and outside organizations permitted the development and use of computer applications at diverse levels: individual, group, and department levels, the overall organizational level, and the interorganizational level. Illustrative of organizational-level systems and applications are global communications networks and enterprise resource planning (ERP) systems. Interorganizational systems that facilitate the placing, tracking, and filling of orders with suppliers are in common use.

The evolution of the Internet and the World Wide Web has further expanded the range and scope of applications. Computer linkages with individual consumers (as contrasted with earlier linkages with wholesalers and business customers) have permitted such innovations as mass customization (the creation of products according to the specification of an individual customer while retaining the economics of mass production), and customer relationship management (CRM) applications such as the creation of individual customer profiles that facilitate systems that "suggest" new products/services that fit with a customer's previous buying patterns. Table I shows some of the varieties of applications that have been developed at, or for, these various aspects and levels of the organization.

E. From Information Poverty to Information Richness

Information "poverty" characterized the early stages of MIS. Information was difficult to obtain, untimely, and expensive. Sales data were obtained, laboriously compiled, and made available to managers on a monthly basis. Financial performance data typically were summarized quarterly. Indeed, it was the desire to have information available in a more timely fashion and at lesser cost that led to the need for, and creation of, MIS.

The current era is characterized by information richness—the availability of information in real time and at low cost. Indeed, some might characterize the situation as one of "information overload" if not for the development of inquiry-based systems which permit more information to be available on an "as needed" basis rather than solely in the form of regularly issued reports.

These changes have revolutionized management, making it possible to track transactions instantaneously (as in financial markets) and to acquire information at a micro level of detail, such as the "stream" of supermarket purchases of specific brands, sizes, and varieties of products by individual consumers. The systems that have been enabled by the availability of such data range from automated programmed trading systems for the purchase and sale of financial instruments to "frequent purchaser" programs for supermarkets that offer discounts, special incentives, and other benefits to their best customers, to customer relationship management which is facilitated by accumulating a wealth of information about a customer, his purchasing habits, and his purchasing search behavior.

Table I Illustrative Applications Serving Various Aspects of the Enterprise

Unit serviced	Applications (systems)
Individual	Personal productivity software (word processing, spreadsheets, scheduling, etc.)
Group	Virtual workspaces Group decision support systems Computer-supported-cooperative-work systems
Department	Sale force management systems Production planning and control systems Financial management systems Human resource systems
Executive management	EIS Strategic planning support systems Negotiations support systems Environmental scanning systems Competitive intelligence systems
Overall organization	Corporate intranets ERP systems Globally integrated communications systems
Interorganizational	EDI and internet links with suppliers and customers Business alliance support systems
Business to customer	CRM systems

F. Integration of the "Islands" of Technology

As the MIS era has progressed, various technologies that are computer-related have been integrated and are now managed in an integrated fashion. For example, "office automation" was once a free-standing technology centered around word processing. Now, a variety of administrative functions utilize networks to link various people and departments so that they may simultaneously work on the preparation of reports or business proposals. Banks once had separate systems for checking account customers, loan customers, and other "product" categories. Now, these systems are integrated, thus enabling the creation of a multifaceted relationship with customers that can be based on an assessment of the potential profitability associated with each. Telecommunications was once a telephony-centered function that was managed separately from MIS. Now, these and a variety of other technologies have been integrated into comprehensive systems that are managed through the MIS function.

G. Globalization

In the early MIS era, even the largest organizations were national or regional in scope. Organizational units shared a common language, a relatively common operating timeframe (e.g., a typical United States company's operations on the east and west coasts were only a few hours apart), and a common culture. Now, with many organizations operating on a global scale, a single company may have branches and subsidiaries in which many different languages are spoken, may face continuous operating time frames, and must accommodate to many diverse cultures.

Globalization has produced significant demands for enhanced communications, standardized systems that can be integrated into a global network, and for coordinating and integrating the efforts of employees operating all over the world.

While globalization has significantly increased the demands placed on MIS, it has also created new opportunities. For example, systems development projects may readily be conducted outside of the organization's home country. Indeed, some organizations now conduct projects on a 24-hour basis by "passing the work" around the globe; as evening comes in Europe, the project can be passed to North America, which in turn passes the work to Asia and Australia and then back to Europe to continue the cycle.

H. Increased Pace of Change

In the early days of MIS, the adoption of computer systems was itself the most radical innovation that an orga-

nization faced. Today, most organizations are in the constant state of change. Some changes are in the form of adaptations to fast-changing market and environmental conditions. Others are planned changes, or transformations, of the enterprise to enable it to become more competitive. Illustrative of such planned changes are "total quality" programs, continuous improvement, creating a learning organization, business process reengineering (BPR), and organizational development.

Many of these planned change programs are data driven; for example, one cannot pursue "total quality" without the creation of databases that enable the tracking of quality in its many forms from the absence of variation in product dimensions to customer satisfaction.

Changes in the "standard" metrics that are used to measure progress in organizations have also been widespread. The reliance on financial performance measures such as "profit" and "return on investment" (ROI), which characterized the earlier business world, has given way to a "balanced scorecard" measurement approach in which measures such as market share, quality, cycle time, and customer satisfaction are routinely monitored and used to evaluate organizational and managerial performance.

The rapid increase in the pace of change and the consequent need to gather, process, analyze, and distribute new and ever-changing varieties of data has placed new demands on the MIS function to develop systems to enable change, to facilitate it, and to monitor and measure progress.

I. From Real to Virtual Organizations

There has been a trend for firms to focus on their core competencies and to create relationships with other firms to provide them with services and functions that many large firms once provided internally. This trend is reflected in the outsourcing of service functions such as legal services and the utilization of manufacturing processes that are partially or entirely performed outside the firm. For instance, in the most modern auto factories, major subassemblies are produced by an outside supplier and delivered to the production line at the precise location and time at which they are needed. The final assembly operation may involve the assembly of only a small number of major modules.

Some firms have chosen to focus only on the design and marketing of products, leaving the entire manufacturing and distribution process to others. Indeed, in some industries, an order to a company may result in the delivery of a finished product to a customer without any employee of the company actually having seen or touched the product.

The most extreme example of a "virtual organization" is an Internet retailer that accepts orders that are placed electronically, passes these orders to a supplier who produces the product and arranges for delivery to the consumer. The entire process is handled electronically and the electronic retailer exists primarily as a virtual organization since its only real manifestations are a small group of executives, MIS specialists, and computer systems.

J. From "Go It Alone" to Partnering

Organizations that once operated on a "go it alone" basis now routinely form partnerships, joint ventures, or other strategic alliances in order to pursue new business opportunities. The cost and time required for developing the skills and capabilities that are required to operate in new countries or to create new products and services for sale can be substantially reduced through such alliances. For instance, a technologically strong firm based in the United States may partner with a foreign firm to produce and distribute products in the partner's home country. In such a case, the foreign partner might provide local knowledge and financial investment while the United States firm might provide product designs, brand names, and technical knowledge to the partnership.

The prevalence of such alliances has placed new demands on MIS, since systems must be developed to serve extra-organizational units that may previously have used systems that are incompatible with those with which the MIS function has been familiar.

VI. THE MIS FUNCTION IN THE ORGANIZATION

The MIS function may be performed within a centralized MIS department, by MIS operatives who are assigned throughout the enterprise, or through the providing of support services non-MIS personnel who perform some or all of their own computing. Commonly, some combination of these three approaches is used in an organization so that MIS can effectively function in support of the pursuit of the broad range of goals of the enterprise.

The MIS function in the organization involves a diverse set of activities. Among them are:

- Monitoring the evolution of information technology
- Managing the development and acquisition of systems

- Managing the organization's internal systems portfolio
- Managing the organization's strategic and enterprise-wide systems
- Managing the organization's extra-organizational systems
- Managing the MIS function
- Providing training and support to systems users
- Managing MIS's organizational relationships

A. Monitoring the Evolution of Information Technology

Information systems are technology-intensive and therefore, the organization must maintain its awareness of the latest developments in the fast-changing arena of information technology. The MIS function must constantly assess the potential utility of new technological developments to the various levels of the organization, to the functions that are being performed, and to the existing systems portfolio. As well, MIS must determine whether new technological developments will facilitate the application of technology to new areas such as the development of new products, the addressing of new markets, the use of new distribution channels, or the acquisition of new varieties of data. For instance, neural networks reflect an evolving technology that may be applied through systems to creatively design new products. The possibility of such systems and applications must be known to the MIS department so that they can work with product designers to determine the potential applicability of such systems to their organization.

MIS must be prepared to make recommendations on the size and timing of investments in new technologies by balancing the benefits that may be anticipated against the costs of doing so. Since the "costs" are not only the direct investment costs and operating costs, but also the cost of training and of the business disruption that is often associated with the implementation of new technologies, such assessments are challenging and usually involve the integration of data, assessments, and estimates from diverse sources.

B. Managing the Development and Acquisition of Systems

The development of new computer applications and systems has long been the responsibility of MIS professionals who operated from a centralized MIS department. This is still a major MIS responsibility; however, today, these professionals are complemented by users who develop their own applications under the guidance of MIS and by information systems specialists who operate within user departments, such as marketing and finance, to facilitate the development of functional applications. Joint project teams made up of user representatives and MIS staff are commonly employed in the development of new systems.

There has been a trend toward the acquisition of software and systems from vendors rather than the in-house development that once characterized the MIS function. The current mix of vendor-supplied and internally developed systems and software places greater demands on MIS than previously, since when a need is identified, a greater range of alternatives must be reviewed and evaluated. Moreover, systems that are acquired from outside vendors normally require a good deal of customization in order to fit well with a particular organization's existing systems, business processes, culture, and other unique features of the enterprise. For example, an enterprise-wide system that is acquired from a vendor might require that a joint team of users, MIS analysts, and vendor consultants work many months to implement the system.

There are a variety of systems development and acquisition practices, such as prototyping or the use of the systems development life cycle, techniques for assessing information requirements, computerized development tools (e.g., computer-assisted software engineering, or CASE) and standards that may be applied to the development and/or acquisition of systems. The development and administration of these processes and methodologies for the creation of new systems and for the maintenance and updating of existing systems is a primary activity of the MIS function. As more organizations operate on a global basis, these activities take on greater importance, since it is critical that common standards, applications, and systems be used throughout the organization's far-flung branches and subsidiaries.

C. Managing the Organization's Internal Systems Portfolio

Internal information systems serve a variety of generic organizational functions such as transaction processing, the recording and tracking of accounting/financial data and information, the support of problem solving and decision making, communications within the organization, and specialized applications throughout the enterprise. Managing this portfolio of systems and applications includes diverse activities

such as selecting the applications to be developed, ensuring that existing applications are updated and improved, and deciding when there is a need for the replacement of existing systems and applications. There is also a constant need to integrate systems to permit managers to be better able to use systems to consider the impact of decisions and actions in one area on other organizational areas.

Among the important varieties of systems and applications that require management in most organizations are

- Transaction processing systems (TPS), which capture, process, and maintain the data from day-to-day operations
- Operational control, and management planning and management control systems, which provide routine reports and/or information from databases in response to inquiries from organizational participants
- Office systems, which are intended to facilitate the productivity of office workers
- Decision support systems (DSS), which provide access to data and decision models and which can be used to forecast and to predict the possible impact of contemplated actions
- Group support systems (GSS), which facilitate computer-supported cooperative work (CSCW) by teams and other groups
- EIS, which provide executive management with information, usually on an inquiry basis, concerning the status of orders, the level of capacity utilization, and other indicators of the existing business situation
- Systems that are created to facilitate or enable the reengineering of business processes, which has become a continuing activity in many organizations
- Product systems, which are integral to, or complementary to, the products or services that the enterprise offers, such as those which incorporate self-diagnosis capabilities in products to enable them to signal the need for maintenance before breakdowns occur, or by offering products or services in combination with information that makes the services more valuable to the customer (such as providing research and analysis information to customers of brokerage firms)
- Knowledge-based systems, which focus on identifying and collecting knowledge into databases so that it can be disseminated to others in the organization who may find it to be useful; illustrative of this are databases for "best practices" or "lessons learned"

The variety of specific applications required by an enterprise is constantly changing as technology advances and the trend toward the integration of "free-standing" systems continues. This integration is occurring so rapidly that anyone who makes a list such as that above risks that it will be obsolete before anyone reads it. Nonetheless, each of these varieties of systems and applications have been important to MIS, and each remains important, even though they may no longer be distinct from each other.

Taken together, the internal systems and applications that an organization has in place represent its internal systems portfolio. The management of this portfolio is one of the basic tasks of the IS function.

D. Managing the Organization's Strategic and Enterprise-Wide Systems

Strategic systems—those that create a competitive advantage for the firm—and enterprise-wide systems constitute a unique challenge for MIS.

Illustrative of strategic IS are the computerized reservation systems of some airlines which have directly created profits for the parent corporations through the fees charged to other airlines for reservation listings and services, and which have indirectly provided advantages through the creation of increased ticket sales for the airlines that own and operate the reservation systems. Although some of the tactics employed in using these systems to generate increased ticket sales, like listing all of the flights of the parent company's airline before any flights of other airlines, have been discontinued after protests from competing airlines, there are many other ways in which systems can provide advantages to the owning airline, such as through the sale of capacity utilization information to the other airlines.

Today, business routinely uses IS to attempt to gain competitive advantage. For instance, one long-distance telephone company introduced a discounted calling plan in which customers could select a single area code for which they could receive a lower rate; another plan provided a discount to customers when calling friends who they identified, but only if those call recipients are also customers of the same long-distance company. In this way, the company obtained new customers and, in effect, motivated existing customers to act as their sales representatives. All of this was enabled by their computer switching and billing

systems and since their major competitors did not have IS that could perform these billing functions at that time, the calling plans were undoubtedly designed so that they would be difficult for a competitor to duplicate.

Enterprise-wide systems are those that integrate diverse functions that were previously performed independently across the enterprise. For instance, global networks allow for the communication of information in various forms—voice, data, e-mail messages, etc.—worldwide and virtually instantaneously. Such networks may be used to create "virtual factories" that are made up of multiple real factories that are located around the globe. The virtual factory software permits production to be planned in a manner that optimally integrates the real factories through consideration of available capacities, delivery demands, relative costs, and relative skills.

ERP systems have been implemented by many large firms to integrate the various steps in the value chain from supply chain activities through production to distribution. ERP systems involve the sharing of data across previously separate business functions to permit decisions to be made in a manner that considers "trade-offs" among the various functions to ensure the best overall result for the enterprise. Enterprise-wide systems are different in scale and complexity from most other internal systems. Strategic systems are often different in scale as well, but they also usually focus on activities and data that are generated outside the boundaries of the organization. It is the scale and complexity of these systems and the external focus which sets the challenge for MIS to manage them.

E. Managing the Organization's Extra-Organizations Systems

With the increasing emphasis on outsourcing, partnerships and strategic alliances, most business firms must be in real-time contact with a variety of other organizations. For instance, most firms utilize electronic data interchange (EDI) or Internet linkages with suppliers. This permits the instantaneous placement of orders, the monitoring of progress in the filling of orders, the opportunity for rapid changes in orders as well as the "just-in-time" (JIT) delivery of parts, assemblies, and supplies as they are needed by a manufacturing firm.

Firms that have become more global and/or more virtual require interorganizational systems that link them closely with suppliers and with other strategic partners who may be involved in joint activities around the globe that are integral to the firm's success.

Systems for electronic commerce enable customers and potential customers to obtain product and other relevant information and/or to enter purchase orders using the Internet. The enhancement of the capabilities of a firm's e-commerce systems is a continuing major task for MIS, as is the integration of extra-organizational systems with enterprise-wide and internal applications systems. Most firms that have effective ERP systems are concerned with fully integrating them with e-commerce systems, for instance, and firms that extensively employ e-commerce are concerned with effective CRM systems that tie into marketing application systems.

F. Managing the MIS Function

The MIS function, or department, is made up, in part, of the centrally operated hardware and software and the people who design, develop, operate, and maintain the organization's centralized IS. The MIS department also arranges for and administers the acquisition of all, or most, of the hardware and software in the organization, since with the passing of the era of the free-standing PC, standards must be developed and applied to ensure that the compatibility and integrity of computer systems, databases, and networks is maintained. Since the MIS function also is usually responsible for the maintenance and updating of this hardware and software, it is very much like a "business within a business" in that its resources must be husbanded and applied productively. This means that IS projects and activities must be identified, staffed, provided with resources, scheduled, and evaluated.

At the strategic level of the enterprise, MIS management is responsible for ensuring that the organization has the computer and communications resources and infrastructure that it needs to fulfill its mission and achieve its objectives. Decisions must be made and actions must be taken to ensure that the organization has an adequate infrastructure of computer systems, networks, and telecommunications capacity. Providing these hardware and software resources requires that comparisons and evaluations be made of various options. For instance, a client/server architecture might best serve one firm, while traditional mainframe-based architecture might be best for another company, while another might be best served by systems based on Internet protocols. One organization might require an intranet (an internal network that operates much like the Internet

and the World Wide Web) and another might not have such requirements.

Less obvious resources that must be provided include the information architecture—the structure in terms of which organizational data are formatted and within which the relationship among data elements are specified—and the design-development methodologies that the organization uses to create its systems. Unless these choices are made at the enterprise level for application throughout the organization, its ISs are likely to be disjointed and unintegrated and will contribute less to the creation of business value than they otherwise might.

At the strategic level of MIS management, another major decision is the selection of an appropriate role for MIS to play in the enterprise. For instance, MIS can be designated to operate as a "computer factory" emphasizing productivity and efficiency in data processing. In such a situation, a decision has to be made that the best possible way in which MIS can contribute to the organization is for it to be focused on the operational and operational control levels.

Alternatively, MIS can focus on decision support with either analytical and/or judgmental emphasis. If this is selected as the focus, MIS is being designated to address the top three levels in the triangle of Fig. 1 by providing such capabilities as those for computerized DSS or group decision support systems (GDSS). Within this role, MIS may also focus on the communications and collaborative aspect of an organization by providing intranets to facilitate knowledge sharing, virtual workspaces for electronic group planning activities, and global networks to link the organization's diverse locations.

Another possible role that may be chosen for MIS within the enterprise emphasizes the strategic level in Fig. 1. Such a role is addressed through development of systems that are directed toward the creation of sustainable competitive advantage for the enterprise.

G. Providing Training and Support to System Users

A major activity of MIS is to provide the training and support that systems users need when new technologies are being incorporated into the organization's processes, new systems are being implemented, and new applications are being developed. Such enablement activities run the gamut from formal training programs to the support of user "help" desks, phone lines, or "information centers" that can respond to telephone or e-mail requests to the providing of "hands on" help to users who may be developing their own applications or having difficulty with new technologies.

If such internal clients were not provided with the necessary training, advice, and help, they would not be as productive as they might be and the organization's performance would suffer.

H. Managing MIS's Organizational Relationships

The day when the MIS department could exist in a service role and remain relatively apart from the business functions of the enterprise are long past. Today, MIS is an integral element of the organization, but in some contexts it is no less susceptible to downsizing or to having some of its functions outsourced than is any other organizational activity.

To ensure that MIS is given appropriate consideration in an organization in which many of the senior executives may not be knowledgeable about computers and IS requires that MIS manage its relationships with other elements of the organization.

These "relationship management" activities begin at the level of the Chief Information Officer (CIO) who is the representative of the MIS function in the highest executive ranks in many organizations. The CIO must understand the current and potential roles of MIS, the vision for the future of MIS, and the business values that it currently creates and that it can create in the future. He must be aware of the role played by MIS in enabling and supporting various business processes and organizational capabilities and he must communicate these to top management so that the role and importance of MIS are visible to the top managers on whom MIS's future will depend.

However, the most important relationships that the MIS function must maintain and nurture are those that exist with MIS's clients: the internal and external individuals, groups, departments and organizations that are served by MIS and on whom it depends for future opportunities to serve. In maintaining these relationships, MIS management is solidifying and expanding the web of relationships that are depicted in Fig. 2, thereby ensuring that the organization makes the best possible use of its MIS resources and ensuring that MIS is a focus of organizational strategy.

VII. CONCLUSION

The field of MIS has evolved from the initial recognition that data from the automation of operational

activities could be useful to the organization's management to a point where the information systems of virtually every organization are critical to its survival. As systems and information become ever more integral to organizational processes, products and relationships, the MIS field is likely to continue to increase in its importance in enterprises.

SEE ALSO THE FOLLOWING ARTICLES

Corporate Planning • Data, Information, and Knowledge • Decision Support System • Globalization and Information Management Strategy • Knowledge Management • Operations Management • Strategic Planning for/or Information Systems

BIBLIOGRAPHY

Ackoff, R. L. (1967). Management Misinformation Systems. *Management Science* 14(4), 18-20.

Argyris, C. (1971). Management Information Systems: The Challenge of Rationality and Emotionality. *Management Science* 17(6), B-275-B292.

Dickson, G. W., & Simmons, J. K. (1970). Behavioral Side of MIS—Some Aspects of People Problem. *Business Horizons* 13(4), 59–71.

Gorry, G. A., and Morton, M. S. S. (1971). Framework for Management Information Systems. *Sloan Management Review* 13(1), 55-70.

Head, R. V. (1970). Elusive MIS. *Datamation* 16(10), 22.

King, W. R. & Cleland, D. I. (1971). Manager–Analyst Teamwork in Management Information Systems. *Business Horizons* 14(2), 59-68.

King, W. R. (1967). The Systems Concept in Management. *Journal of Industrial Engineering* 18(5), 320-323.

Lindgren, L. H. (1969). Auditing Management Information Systems. *Journal of Systems Management* 20(6), 22-27.

Mason, R. O., & Mitroff, I. (1973). Program for Research on Management Information Systems. *Management Science Series A-Theory* 19(5), 475-487.

McKeever, J. M. (1969). Building a Computer-Based MIS. *Journal of Systems Management* 20(9), 12-17.

Shipman, F. W. (1969). Designing MIS for Managers. *Journal of Systems Management* 20(7), 14-19.

Zani, W. M. (1970). Blueprint for MIS. *Harvard Business Review* 48(6), 95.

Manual Data Processing

Kenny K. F. Lee
Nanyang Technological University, Singapore

GLOSSARY

data Data represents the raw facts and figures about a business's activities and its processes. Often, it describes the attributes of a business (or financial) event or transaction. Data by itself is generally not of use and lacks value until it has been transformed or processed into usable output or information.

data processing The systematic performance of operations, whether manual or automated, upon data, for example, classification, coding, batching, validation, sorting, updating, and reporting.

files An organized collection of related records. In the course of transforming raw data into usable information, data processing makes extensive use of transaction and master files. Files of source documents are called *transaction files,* while *master files* are central files that contain the complete set of records connected with the processing of the system. An example of a master file is the subsidiary ledger of an accounts receivable system, which contains the entire collection of a business's customer accounts records.

inventory Refers to the stock of raw materials, work in progress, or finished goods held by a business.

transaction A business event that could impact either the financial or non-financial status of the business. For instance, a business transaction such as a sale would have an accounting impact because it affects the financial status of the business; however, an event such as raising a purchase order has no accounting impact. In manual systems a transaction is captured on a document, for example, a sale to a customer is recorded on a sales invoice.

I. INTRODUCTION

The manual method of processing data is one in which the various steps of a procedure for handling data are performed by some human effort, direction, and control.

The articles opens with a couple of illustrations to introduce this method of processing data. Building on these, the next sections develop the unique features of the manual concept of processing data and the basic processing operations involved. An extended inventory processing example follows to illustrate these concepts. A graphic depiction of the processing steps accompanies the example in the form of a simple flowchart. The emphasis throughout this article is on developing a sound grasp of the manual data processing concept, which is fundamental to the understanding of computer-based information systems. The chapter closes with the reminder that the manual processing model is still very relevant today despite the rapid proliferation of computerized systems.

> The proprietor of a small Expresso bar works late into a Sunday night to bring up to date her business accounts for the previous week. Assisted by an accounts clerk she pores over journal files of cash register paper tape records, shuttling back and forth between open filing cabinets and a photocopier machine. The

personal secretary to the CEO of a consumer electronics multinational subsidiary retrieves a stack of transportation and entertainment bills that her boss has accumulated over the course of the month and settles down with a general-purpose handheld calculator to fill out the company's expense claims form for him. A freshman student leafs through pages of industry statistics and compiles trends and comparative information in an attempt to assess employment prospects across industries and to decide the major he wants to pursue in his sophomore year.

The scenarios above are all illustrations of manual data processing instances in familiar business and nonbusiness organizational settings. Manual data processing is one of several models of processing data that is still relevant in today's business organizations. Despite the rapid proliferation of computer-based information systems over the latter half of the 20th century, this method of data processing remains the dominant form of processing data vis-à-vis computerized processing in most organizations today.

II. DISTINGUISHING FEATURES

The manual method of processing data is one in which the various steps of a procedure for handling data are performed by some human effort, direction, and control. Data is primarily processed by hand, and paper is used as the primary means, or medium, to capture, store, and duplicate business data. Often, there is the existence of significant amounts of raw data to be processed either on a repetitive, routine basis or on a one-off basis, as the opening scenarios illustrate. Office machines may also augment the process, e.g., cash registers capture transactions at source onto paper tape rolls, handheld calculators assist with repetitive computations, and photocopiers help create additional copies of source documents and distribute reports, while filing cabinets securely store away processed source documents and reports. However, these machines remain under the direct control of the operator and they do not change the basic character of the work or the process involved.

The paper forms on which input data is captured for processing are referred to as source documents, e.g., invoices, goods receiving notes, etc. Paper is also used as the medium of storage of file information, in which case they are called records, and as output media to communicate and to distribute processed data to information customers internal and external to the organization and across organizations.

The entire collection, or system, of data processing components (the data inputs, the human processor, information outputs, data storage devices, the processing medium, and the supporting office equipment) that processes raw data into useful information is called a *data processing system* or, in more contemporary language, an *information system* (IS). This arrangement is depicted graphically using a simple generic system [or input-process-output (IPO)] model in Fig. 1.

III. DATA PROCESSING OPERATIONS

The processing of transactions through an IS is cyclic. Initially, raw transaction data are classified, coded, and entered into the data processing system as input. The processing of the data then takes place using a successive sequence of data manipulation operations that may involve recording, classifying, sorting, calculating, summarizing, updating, and storing. Processed data emerges from the system as output contained on forms, reports, and useful information reports. The cycle of operations is then repeated for the next transaction and so forth until all transactions are processed. The basic data processing stages of input, processing, storage, and output operations make up a generalized data processing framework that is applicable to any data processing method or system.

Each processing stage may be made up of a number of more detailed, finer data processing steps. For instance, the data input stage may comprise detail operations such as coding, batching, checking, and the transmission of transactions. The processing stage frequently includes numerous data manipulation operations, the actual sequence of implementation of which depends upon the specifics of the requirements of the processing scenario. One common sequence is that of the inventory transaction processing illustration discussed in the next section.

Data, in and of itself, is generally not useful and lacks value until it has been transformed into usable output or information. In the course of transforming raw data into usable information, data processing makes extensive use of transaction and master files. A file is an organized collection of related records. A folder of invoices is a transaction file, while an inventory ledger is an example of a master file.

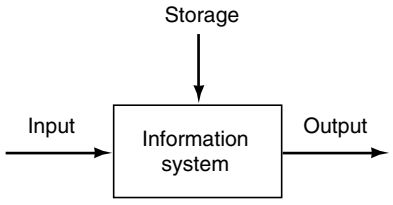


Figure 1 Systems/IPO model.

IV. MANUAL DATA PROCESSING EXAMPLE

To illustrate the mechanics of manual data processing operations, a simple inventory processing scenario involving the performance of six basic processing functions by a group of office staff (Fig. 2) will be discussed. The office comprises a messenger and four employees that process the data. Together they are responsible for maintaining the organization's perpetual inventory records for the range of products the company sells.

A. Input

In manual data processing there is first the need to collect or gather the data from the various sources and then to bring the data to a central point (the office) where it can be input into the processing system. In the inventory processing situation, a source of data is the individual sales and sales returns transactions that are recorded daily in the sales department. These sales and sales returns transactions are contained on hardcopy multipart forms or source documents. In this example, the assumption is made that sales and sales returns transactions are captured onto a common source document, the customer invoice. It is further assumed that each invoice is associated with only one product item. These invoices are collected periodically and delivered to clerk A (Fig. 2) of the office staff by a messenger who effectively performs the first function in the data processing system—input.

Since the invoices record individual business transactions, they are commonly known as *transaction records*. Each transaction record contains data on a sale or a sales return and is used to update inventory master data contained on an inventory master record.

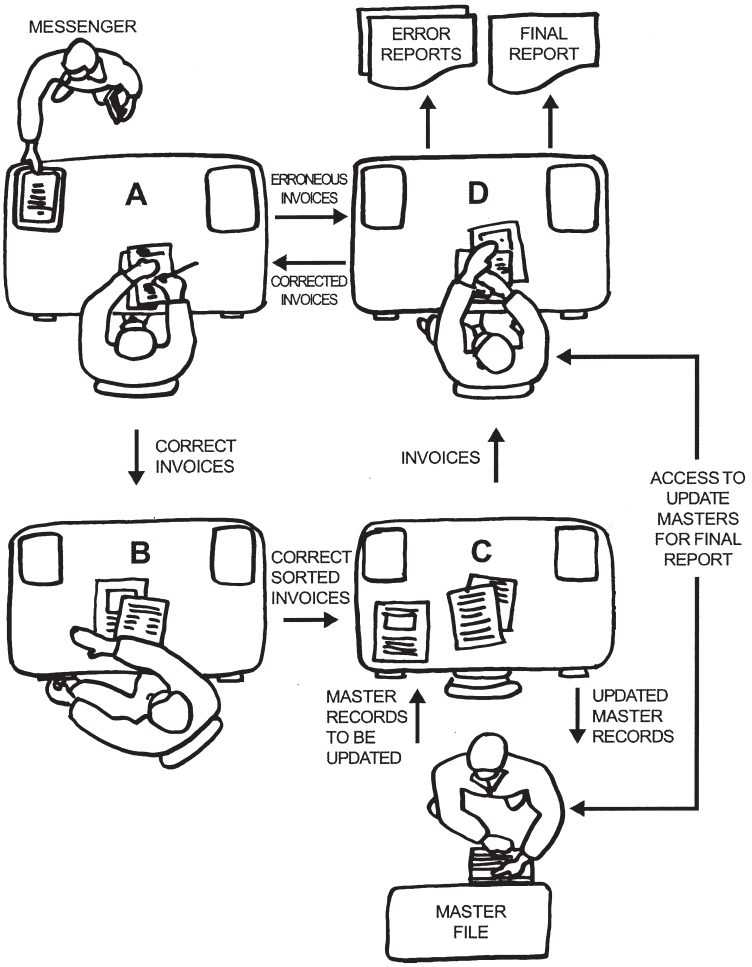


Figure 2 Inventory processing example [adapted from NCR Corporation. (1974). *Data processing concepts course.* location: Howard W. Sams & Co., Inc.].

B. Validation

The invoices need to be vetted or checked for errors first and then sorted into the sequence of the master file records to make processing easier and more efficient. This involves clerk A screening the invoices to see that product data are complete and accurate (Fig. 2). This screening might include such checks as making sure that the product number is within the range of product numbers produced by the company, that the product code is captured correctly, and so forth. Another check involves the validity of the transaction code: each invoice contains a numeric transaction code, for instance, the number 1 or 2 to differentiate between sales and customer returns transactions. The number 2 indicates a sale to a customer, and the number 1 indicates a sales return, i.e., goods returned by a customer.

Clean or error-free invoices are passed to clerk B for further processing. If clerk A discovers any data omissions or errors, the erroneous invoice is handed to clerk D, who investigates the error, corrects it, and passes the invoice back to clerk A. Clerk A rechecks the invoice and then forwards the corrected invoice to clerk B. This process of checking for errors and of error correction is called validation. The purpose is to ensure that invoices to be processed through the system are accurate and valid.

The process of validation must be done according to strict business rules. For example, one rule in a given processing application might be that the account number cannot contain any alphabetic characters. If an invoice is found that has an account number containing, for example, the letter "I" accidentally inserted in place of the number "1," then the rule has been violated (and clerk A must then pass the erroneous invoice to clerk D to rectify). Validation and the establishing of business rules are important elements of business data processing.

When all the invoices have been checked and corrected, clerk D prepares a report for management that gives the number and types of errors made by the sales office that day.

C. Sorting

A master file in a data processing system is arranged in some kind of logical order, usually alphabetically or numerically within a predetermined category. Say, for example, that the inventory master file is in numerical sequence by product number. To facilitate the master file updating process, the invoice transactions need to be sorted by clerk B into the same order as the inventory master file sequence. Following this, clerk B passes the sorted invoices on to clerk C to process against the inventory master file (see Fig. 2).

This depicts a highly simplified sorting scenario. In real-life processing, more complex sorting operations may be executed on the inventory movement transactions. For high-volume transaction environments, invoices may initially be input to a number of edit clerks. In such a case, checked or edited documents may need to be organized further for processing by first merging or combining the collective output of these edit clerks. Following this, the sorting clerk will sort the merged invoices into the various transaction types, or inventory movement categories, e.g., into inward movement transactions like goods received, returned, etc. (transaction code 1), ahead of issues transactions (transaction type 2). Following this, the transactions within each category are sorted or resequenced into ascending inventory number sequence. Again, this has the effect of (1) arranging the transactions into the same overall sequence in which the inventory master file is maintained and (2) grouping transaction types by movement category. In this way the inventory master file will be processed with the receipts items first, followed by the issues. This arrangement assures that customer experience of inventory stock outs will be minimized.

D. Processing and Filing

Updating is an important part of the processing function. In Fig. 2, clerk C is responsible for updating the information on the transaction records (the invoices) into the master records of the inventory master file. This master file contains a complete set of the business's inventory records, one for each product in the company's product line, and contains necessary data about the products such as product number, product name, and number of units on hand.

Beginning with the first invoice, clerk C finds the corresponding master record in the master file, removes it to his or her desk, and updates the information on the master record according to the data from the invoice. If the transaction code on the invoice is a 2, indicating a customer (product) return, clerk C adds the quantity on the invoice to the quantity on the master record. (All returns can be added back to the inventory.) If the invoice transaction code is 1, indicating a sale, clerk C subtracts the (product) quantity on the invoice from the quantity on the master record. The master file is stored in an appropriate storage location (Kardex tray, filing cabinet, etc.) in the office.

Following this transaction, all other invoices or source documents referring to the same master record are updated against the master record in the same manner (it is quite common to encounter more than one transaction for a master record). After one master record has been completely updated, clerk C returns it to the master file storage location and repeats the procedure for the next invoice from the stack of invoices he or she received from clerk B. This continues cyclically until all of the day's invoices (or the entire batch) are processed onto the master records and all records are filed. Clerk C then performs two of the data processing functions: processing (updating of the master records) and filing (placing the updated master records back into the master file) or data storage.

In practice, clerk C would probably remove a group of records from the file at one time so that it would not be necessary to go back and forth to the file continually. For example, if the filing key were alphabetical then all the master records beginning with "A" might be moved to the desk at one time. Individual master records would then be updated with the corresponding transaction records one at a time. The A records would be returned to the master file and the "B's" would be taken back to the desk. It would be faster and more convenient to process a group of records at a time; each record would be updated individually, but the retrieval from the file would be as groups of records at one time.

In practice, transactions are also processed in batches to facilitate efficient utilization of resources. This means that the invoices or source documents are initially accumulated into batches in the sales office. From there the document batches are transported to clerk A in the sales office where the documents are validated batch by batch, following which they progress to clerk B for sorting, and so on.

E. Output

The final step is to extract the processed results (or a part of these) from the inventory master file and to distribute these to the information customer(s). After the updating task is completed, clerk D prepares various reports by extracting the relevant data from the updated inventory master file. These reports may be a summary of all sales for the day, week, and month; a listing of all customer returns for a specified period of time; current inventory levels; or any other specific report that management might request. Often, the information contained in such reports will require additional data processing and manipulation operations. The reports that clerk D generates comprise the output of the illustrative data processing system in Fig. 2.

V. FLOWCHARTING MANUAL DATA PROCESSING PROCEDURES

A flowchart is the graphic representation of the sequence of operations required to carry out a data pro-

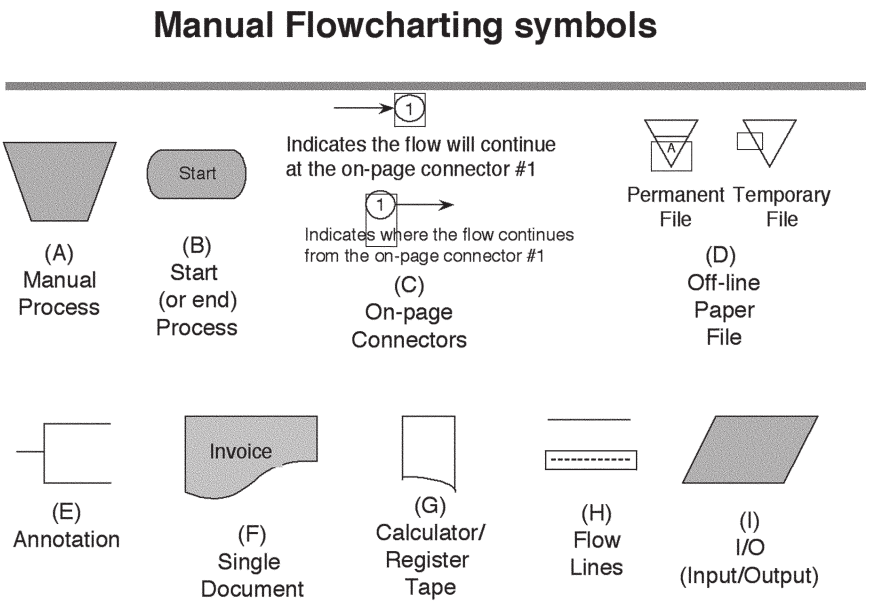


Figure 3 Manual flowcharting symbols. (Adapted from Hollander *et al.,* 2000.)

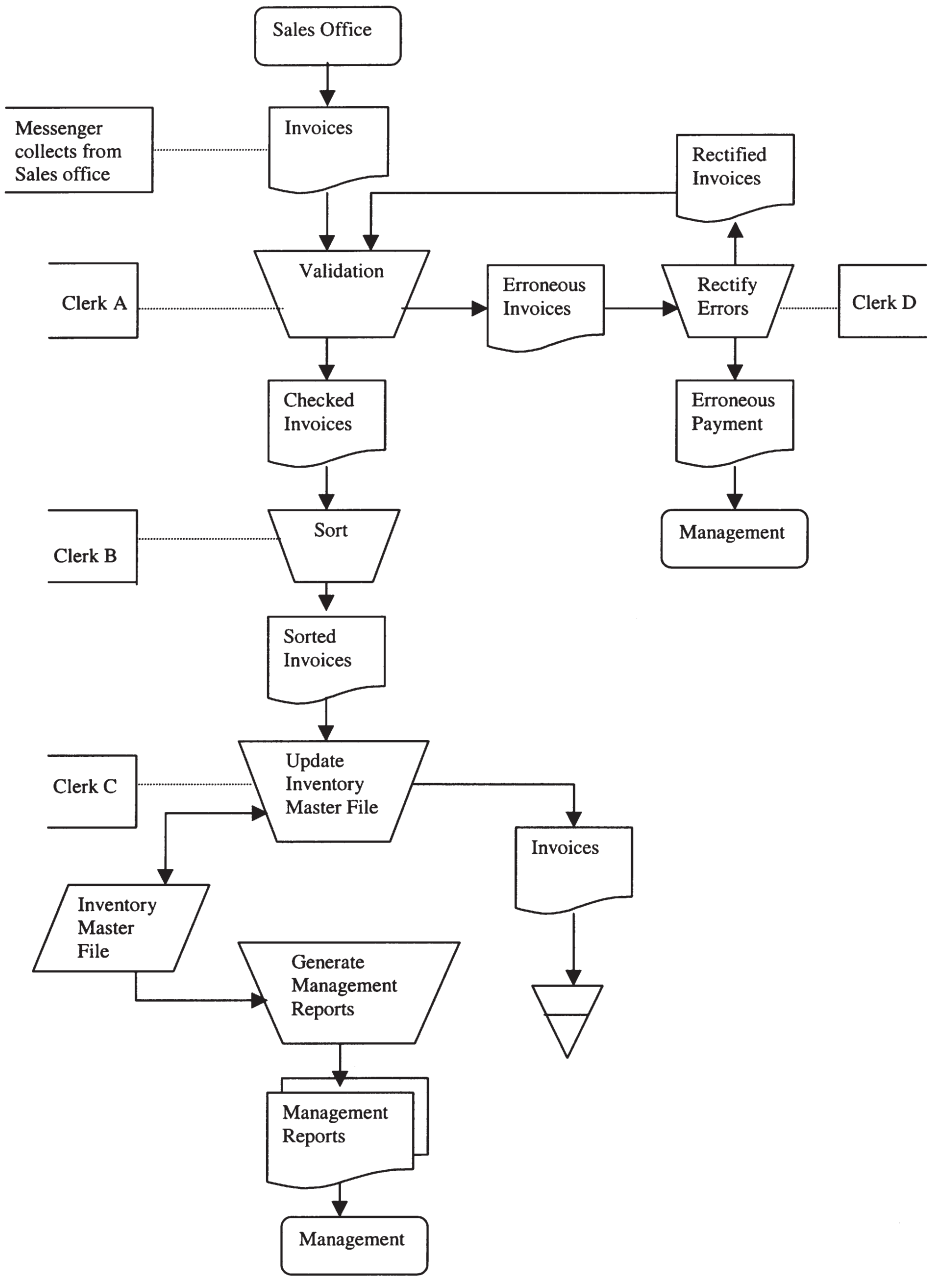


Figure 4 Manual flowchart of an inventory processing system.

cessing procedure or routine. Manual flowcharts clearly document a given business procedure and are often used to institutionalize standard manual data processing procedures into a business organization. A standard symbol set forms the basis for modeling data processing operations and the flow of data through a system; some of these symbols are contained in Fig. 3.

Based on the inventory processing example (Fig. 2), the essential data processing operations in processing inventory movement transactions may be identified as:

Step 1: Each day the inventory movement transactions (invoice documents) are collected from the sales office by the messenger and passed to clerk A in the central office.

Step 2: Clerk A validates the transactions for accuracy against the inventory master file.

Erroneous transactions are passed to clerk D; clean transactions are passed to clerk B.

Step 3: Clerk D corrects the erroneous transactions and passes them back to clerk A. A report of errors is also generated for management.

Step 4: Clerk A rechecks the corrected transactions then passes them to clerk B.

Step 5: Clerk B sorts the clean transactions into master file sequence and passes the sorted transactions to clerk C.

Step 6: Clerk C updates each transaction into the inventory master file, processing each transaction according to its transaction code.

Step 7: Management reports including summaries of inventory movements are prepared by clerk D and distributed to management.

A manual flowchart that documents this inventory processing process can now be drawn based upon these seven data processing steps (Fig. 4).

VI. CONCLUSIONS

Today, organizations typically utilize both manually processed and computer-based information systems. Despite the rapid deployment of information technology in business organizations today, manual ISs continue to play an important role alongside computerized systems and to outnumber them in terms of the quantity of systems and the volume of information processed. This chapter discusses the manual data processing model and attempts to illustrate its role within the context of transaction processing in the traditional, functionally structured business organization.

A sound grasp of manual data processing concepts lies at the heart of a good understanding of contemporary information processing systems. Given such an understanding readers will become more effectively positioned to assist their organizations in deploying the most appropriate information mechanism, whether manual or computerized, that is consistent with their organizational information perspective, their information processing capability, and the information richness desired.

SEE ALSO THE FOLLOWING ARTICLES

Database Administration • Database Development Process • Data Flow Diagrams • Transaction Processing Systems • User/System Interface Design

BIBLIOGRAPHY

Awad, E. M. (1973). *Automatic data processing: Principles and procedures,* 3rd ed. New York: Prentice Hall.

Daft, R. L. (1998). *Organization theory and design,* 6th ed. Cincinnati: South-Western Publishing.

Davis, G. B. (1978). *Computers and information processing.* New York: McGraw-Hill.

Drucker, P. F. (January-February 1988). The coming of the new organization. *Harvard Business Review,* Vol. 45.

Hollander, A. S., Denna, E. L., and Cherrington, J. O. (2000). *Accounting, information technology, and business solutions.* 2nd ed. New York: McGraw-Hill.

Mukerji, D. (2000). *Managing information: New challenges & perspectives.* New York: Prentice-Hall.

The National Computing Centre Limited. (1980). *Introducing data processing.* Manchester, England: NCC Publications.

NCR Corporation. (1974). *NCR data processing concepts course.* Indianapolis, IN: Howard W. Sams & Co., Inc.

NCR Corporation. (1979). *EDP concepts course.* Indianapolis, IN: Howard W. Sams & Co., Inc.

Wanous, S. J., Wanous, E. E., and Wagner, G. E. (1971). *Fundamentals of data processing.* Cincinnati: South-Western Publishing.

Marketing Information Systems

Robert R. Harmon
Portland State University

GLOSSARY

customer relationship management (CRM) Software applications manage the interaction of customers with an organization. They are used to increase the return on marketing efforts by enabling the understanding of the complete history of a firm's interactions with its customers. CRM systems are able to target promotions to likely buyers, facilitate sales efforts, and deliver customer service.

cybermarketing The convergence of the Internet, computers, information systems, telecommunications, and the customer with the marketing process.

data mart A scaled down version of a data warehouse that usually holds a subset of the entire data set in order to provide more focused and faster access to specialized data.

data mining Computer-based exploration and analysis of large quantities of data in order to discover meaningful patterns and rules for the purpose of improving marketing, sales, and customer service operations.

data warehouse Electronic storage that is a repository where data from internal and external sources are collected, organized, and stored for future analysis.

enterprise resource planning (ERP) Software applications that integrate back office systems for order processing, manufacturing, finance, accounting, and human resources. ERP functions, in turn, integrate with marketing front office activities and supply chain management activities.

geographical information systems (GIS) Software that enables the geographic mapping of information such as the locations of customers, competitors, suppliers, sales prospects, suppliers, and partners. GIS can be used for site selection, trade area analysis, environmental analysis, sales territory design, and the targeting of marketing communications.

marketing automation An emerging discipline that encompasses the automation of front office marketing activities such as e-commerce, decision support, sales, customer relationship management, and customer service.

marketing decision support system (MDSS) A set of core applications in the marketing information system that provides computer-based tools, models, and techniques to support the marketing decision making process.

marketing information system (MkIS) A computerized system that provides an organized flow of information to enable and support the marketing activities of an organization.

on-line analytical processing (OLAP) A family of analysis and report-generating tools used to access large databases. OLAP enables partially aggregated data or full reports to be stored for fast, convenient access.

supply chain management (SCM) Software applications that integrate electronic procurement, inventory management, quality management, and logistics systems that link an enterprise to its suppliers.

I. INTRODUCTION

The Internet is rapidly changing the way business views marketing information systems. New business

models present challenges and opportunities as organizations seek to adopt "e-business" methodologies in the search for competitive advantage. Organizations of all sizes are feeling the "ripple effect" of Internet-enabled customers, supply chains, and competitors. This pressure is particularly acute in the marketing function where information technology touches the customer and is increasingly becoming the key to creating superior customer value.

The importance of marketing information is particularly apparent as the economy continues to emphasize services as a primary source of value. Services are heavily information dependent. Information is rapidly becoming a service in its own right. Even in industries that are primarily manufacturing in nature, the information content of the final product is rapidly increasing. Mass customization, often described as "one-to-one" marketing or the customizing of products and services for individual customers, is heavily dependent on comprehensive and timely customer information.

Modern marketing organizations, with their focus on the Internet, exhibit different characteristics than their "old economy" brethren. They create and manage the customer interface where interactions are more virtual than face to face. They leverage information technology to integrate and coordinate with customers and business partners to rapidly achieve measurable business results. The emphasis is on the rapid conversion of knowledge into customer value which depends on the ability to develop, deploy, and manage powerful new marketing information systems (MkIS). The key to competitive advantage depends on the firm's ability to convert knowledge into customer relationships, reduce time to market, and lower costs.

To survive in highly competitive markets, companies need to be able to develop the marketing function and scale it up on "Internet time" with best-of-class decision support solutions for customer relationship management, sales force automation, market research, marketing communications, logistics, and product development. The purpose of this article is to provide an overview of the MkIS as it is evolving into an Internet-based system.

A. Definition—Marketing Information System (MkIS)

Simply put, a MkIS is a computerized system that is designed to provide an organized flow of information to enable and support the marketing activities of an organization. The MkIS serves collaborative, analytical, and operational needs. In the collaborative mode, the MkIS enables managers to share information and work together virtually. In addition, the MkIS can enable marketers to collaborate with customers on product designs and customer requirements. The analytical function is addressed by decision support applications that enable marketers to analyze market data on customers, competitors, technology, and general market conditions. These insights are becoming the foundation for the development of marketing strategies and plans. The MkIS addresses operational needs through customer management systems that focus on the day-to-day processing of customer transactions from the initial sale through customer service.

MkIS systems are designed to be comprehensive and flexible in nature and to integrate with each other functionally. They are formal, forward looking, and essential to the organization's ability to create competitive advantage. The MkIS is the firm's "window on the world," and, increasingly, it is the primary customer interface. Figure 1 presents the basic architecture of the MkIS.

B. The Strategic Role of the Marketing Information System

Historically, the role of the marketing function has been to support "make and sell" business strategies that emphasized increases in market share over the creation of long-term customer value. This view started to change after World War II with the recognition that satisfying the customer's needs and wants should be the focus of a firm's business activities.

The emphasis on the customer elevated the importance of marketing as a core business function on par with research and development and production. The marketing function has become the firm's window to the world in the sense that it must monitor the marketing environment for changes in buyer behavior, competition, technology, economic conditions, and government policies. Marketing is a "strategic" function in that marketing activities enable organizations to identify and adapt to changes in the market environment.

The strategic function of marketing is further emphasized as Internet-based technologies have enabled radically new approaches to selling where information technology for the first time touches customers and provides new means for collecting marketing information. In a knowledge-intensive economy, the ability to collect, analyze, and act upon marketing in-

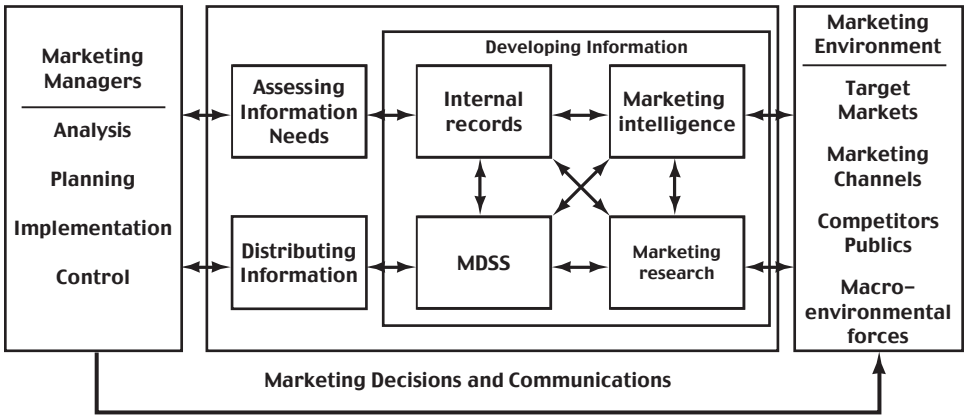


Figure 1 The marketing information system. From *Marketing Management: Analysis, Planning, Implementation, and Control*, 9th edition by Kotler, Philip © 1997. Reprinted by permission of Pearson Education, Inc., Upper Saddle River, NJ.

formation more rapidly than the competition is the core competency from which competitive advantage flows. MkISs provide the information technology backbone for the marketing organization's strategic operations. In a broader sense, the MkIS creates an organized and timely flow of information required by marketing decision makers. It involves the equipment, software, databases, and also the procedures, methodologies, and people necessary for the system to meet its organizational objectives. The MkIS encompasses a broad spectrum of activities from simple transaction processing to complex marketing strategy decision making.

C. The Role of the Internet

Information technology has transformed how firms conduct business. For example, financial service providers such as banks, stockbrokers, and insurance companies could not do business today without their client-server-based information technology. This technology has long supported marketing activities. However, it is the recent advent of the Internet, and especially the browser-based World Wide Web, which has ignited a revolution in MkISs.

The term "cybermarketing" is often used to describe the Internet's convergence of computers, information systems, telecommunications, and the customer with the marketing process. Internet marketing is characterized by interactivity, graphical user interfaces, multimedia content, and one-to-one connectivity. Internet technologies are not only providing new ways to reach the customer, but also to enable the reengineering of the entire marketing process and, indeed, the entire enterprise. It is no longer acceptable to view marketing as a stand-alone activity with lengthy time lapses between product concept, marketing strategy, and commercialization. Marketing has become interactive and real time.

The rapidly growing field of marketing automation encompasses customer management functions to support e-commerce. As depicted in Fig. 2, customer management applications include marketing decision support systems, customer relationship management, sales force automation, customer service, and e-commerce activities. These activities are often described as "front office" customer-oriented activities. "Back office" enterprise resource planning (ERP) activities include manufacturing, finance, and human resources. "Supply chain management" (SCM) activities encompass electronic procurement, inventory management, quality management, and logistics systems to link an organization with its suppliers. These three elements comprise the enterprise information system.

Large, integrated enterprise software companies such as Oracle, SAP, PeopleSoft, and IBM address all three major applications, and they are beginning to use Web-based technologies to redesign business processes throughout the organization. The result is leaner organizations, faster response times, and lower costs. Decision makers are able to integrate information from customers, suppliers, and the internal organization to obtain an enterprise-wide view of their ability to develop and execute marketing strategy. Until truly integrated browser-based systems are widely in use, the principal challenge for marketers is to tie e-commerce generated data with legacy information systems in order to create a unified view of each customer. Marketers will need to understand all the

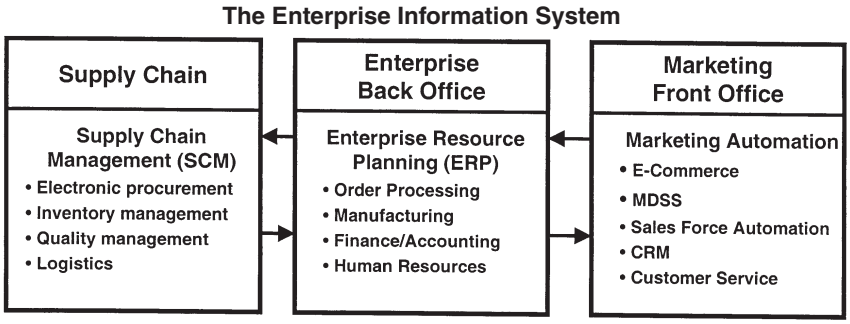


Figure 2 The interfaces of the MkIS.

various ways that customers are touching the business through existing interfaces and e-commerce. Many of the older interfaces, such as telemarketing centers, point of sale (POS) systems, and the sales force, are likely to be supported by legacy client-server technology. Marketers need to consolidate data from those systems with that from the Web-based e-commerce sources into a holistic view of the customer and make it available to decision makers.

D. Benefits of the Marketing Information System

The MkIS increases the number of options available to decision makers and supports every element of marketing strategy. MkIS affects marketing's interfaces with customers, suppliers, and other partners. The primary benefits of the MkIS impact the areas of functional integration, market monitoring, strategy development, and strategy implementation.

1. *Market monitoring.* Through the use of market research and marketing intelligence activities the MkIS enables the identification of emerging market segments and the monitoring of the market environment for changes in consumer behavior, competitor activities, new technologies, economic conditions, and governmental policies. Market research is situational in nature and focuses on specific strategic or tactical marketing initiatives. Marketing intelligence is continuous in nature and involves monitoring and analyzing a broad range of market-based activities and information sources. There are three major sources of market information. The first is syndicated data published by market research companies and industry associations. Company-sponsored primary research is another option. It is much more focused since specific questions are asked of respondents within certain markets. But, it is considerably more expensive and time consuming. Perhaps the best data available are the customer's behavior captured from Web site viewing, POS transactions, and systematic feedback from the sales force.
2. *Strategy development.* The MkIS provides the information necessary to develop marketing strategy. It supports strategy development for new products, product positioning, marketing communications (advertising, public relations, and sales promotion), pricing, personal selling, distribution, customer service, and partnerships and alliances. The MkIS provides the foundation for the development of information system-dependent e-commerce strategies.
3. *Strategy implementation.* The MkIS provides support for product launches; enables the coordination of marketing strategies; and is an integral part of sales force automation (SFA), customer relationship management (CRM), and customer service systems implementations. The MkIS enables decision makers to more effectively manage the sales force as well as customer relationships. Some customer management software companies are extending their CRM applications to include partner relationship management (PRM) capabilities. This has become increasingly important as many marketers are choosing to outsource important marketing functions and form strategic alliances to address new markets.
4. *Functional integration.* The MkIS enables the coordination of activities within the marketing department and between marketing and other organizational functions such as engineering, production, product management, finance, manufacturing, logistics, and customer service.

E. Marketing Information System Functional Components

As shown in Fig. 3, an MkIS consists of four major components: (1) user interfaces, (2) applications software, (3) databases, and (4) systems support.

1. *User interfaces.* The essential element of the MkIS is the managers who use the system and the interfaces they need to effectively analyze and use marketing information. The design of the system will depend on what type of decisions managers need to make. The interface includes the type of hardware that will be used; the way information is analyzed, formatted, and displayed; and how reports are to be compiled and distributed. Issues to resolve are ease of use, security, cost, and access.
2. *Applications software.* These are the programs that marketing decision makers use to collect, analyze, and manage data for the purpose of developing the information necessary for marketing decisions. Examples include the marketing decision support system (MDSS) software and customer management software for on-line sales and customer service.
3. *Marketing databases.* A marketing database is a system in which marketing data files are organized and stored. Data may be collected from internal and external sources. Internal sources largely result from transactions. They provide data from e-commerce sites, sales results, shipping data, inventories, and product profitability. External sources include market research, competitor intelligence, credit bureaus, and financial institutions. Data can be organized in a *flat file* (text file with one data record per line) or a *relational database* (data is stored in tabular form where each row represents one entity and each column represents one characteristic of that entity). For instance, each row could represent a customer with the columns providing the name, identification number, and purchase information.
4. *System support.* This component consists of system managers who manage and maintain the systems assets, including software and hardware networks; monitor its activities, and ensure compliance with organizational policies. This function may also include a help desk for system users.

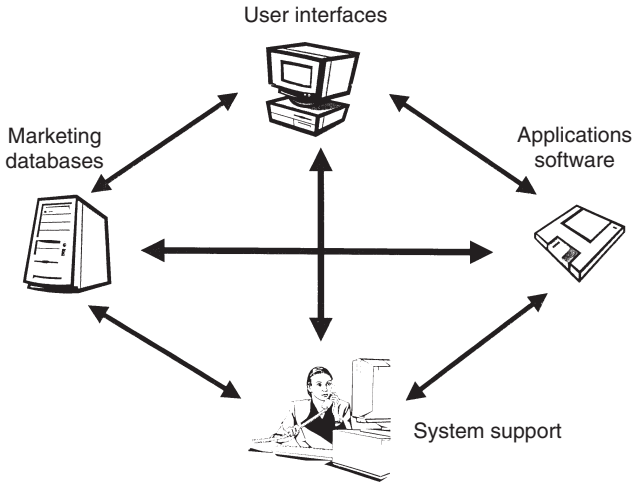


Figure 3 MkIS functional components.

II. MARKETING DECISION SUPPORT SYSTEMS

Marketing decision support systems (MDSSs) constitute a set of core applications of the MkIS. The MDSS provides computer-based tools, models, and techniques to support the marketing manager's decision process. In the general case, MDSS is optimized for queries of historical data. MDSS data typically are derived from both internal and external market sources. The MDSS features inquiry and report generation functions where the manager can access marketing data, analyze it statistically, and use the results to determine an optimal course of action.

A. Marketing Decision Support System Functions

The marketing decision support system (MDSS) provides analytical models for forecasting, simulation, and optimization. MDSS tools include simple spreadsheets such as Excel, statistical analysis packages such as SPSS and SAS, on-line analytical processing (OLAP) tools, data mining applications, and neural networks. The MDSS provides the user with the ability to explore multiple options. Typical MDSS functions include models and tools for:

1. *Sensitivity analysis.* Decision makers can explore changes in a strategic variable such as price and model, its impact on demand, or competitive behavior.
2. *What-if analysis.* This analysis can be easily accomplished with a spreadsheet. Revenues and costs can be manipulated to show the impact of each variable on profits and cash flows.

3. *Goal setting.* Analysis focuses on the desired result and builds the resource base necessary to accomplish the goal.
4. *Exception reporting.* Analysis looks for results that exceed or fall short of stated goals or benchmarks—sometimes called gap analysis. Which products or segments exceeded sales forecasts?
5. *Pareto analysis.* Analysis looks for activities that generate disproportionate results. For instance, the top 20% of customers may account for 80% of sales revenues.
6. *Forecasting models.* Econometric models are used to analyze time series data for the purpose of predicting future sales and market share levels.
7. *Simulation models.* Monte Carlo simulations address marketing decision making under conditions of uncertainty. Variables such as the market price, unit variable cost, and quantity sold are not known ahead of the product investment decision. Simulation models allow the marketer to analyze risk and assess the probabilities of likely outcomes of their decisions.
8. *Scorecards and dashboards.* Scorecard systems present a consistent framework for tracking the effectiveness of marketing activities. They often have different modules for senior executives, marketing managers, product managers, and customer service managers. Scorecard systems allow the user to "drill down" on an analytic and workflow basis to determine the status of any strategic initiative. Dashboards allow frontline managers to monitor their critical performance indicators. These systems are often used in conjunction with "best practice" standards for call-center-based customer support.

B. MDSS Analyses

Marketers typically use MDSS models and tools to analyze markets, customers, competitors, and internal operations. The following list presents some of the most common types of issues targeted by MDSS analyses.

1. *Market segment analysis.* Use of modeling techniques to identify segments and analyze economic trends, demographics, and behavior.
2. *Market share analysis.* Analyze trends and determinants of market share.
3. *Competitor analysis.* Analysis of competitors' market positions, economics customer base, and marketing strategies.
4. *Pricing analysis.* Identifies and analyzes the factors that influence a firm's ability to set prices, including price elasticity and demand analysis. Includes internal economics and market-related factors.
5. *Cost analysis.* Studies a firm's overall cost structure and its impact on product cost. Margin analysis combines cost analysis with pricing analysis. Variance analysis looks for explanations of costs overages and underages.
6. *Sales analysis.* Studies the distribution of a firm's sales by region, product, brand, sales territory, etc.
7. *Sales forecasting.* Develops estimates of sales potential by product, region, sales territory, brand, etc.
8. *Sales force productivity.* Studies sales force effectiveness and efficiency and contributing factors.
9. *Advertising analysis.* Analyzes advertising effectiveness, media choices, and brand awareness.
10. *Distribution.* Analyzes channel decisions from economic and strategic perspectives.
11. *Simulation.* Simulates decision making under various strategic scenarios.
12. *Customer satisfaction.* Analyzes issues concerning the customer's expectations and outcomes with the product.

C. Data Warehousing

IBM defines a data warehouse as a place that stores enterprise data designed to facilitate management decisions. In essence, a data warehouse provides the basis for an analytical system where periodic data points are collected and stored at specified times for future analysis. Data warehousing enables marketers to capture, organize, and store potentially useful data about customers and markets for decision making purposes.

Every record of a transaction or interaction with a customer, supplier, channel member, or sales person is an opportunity to create knowledge. Firms collect data from these day-to-day business operations. In order for this data to be useful, it is often organized and stored in a data warehouse. Simply put, a marketing data warehouse is a repository for data that has been collected from internal and external data sources. Each customer generates a stream of transaction records over time. Data sources may include scanner data, billing records, applications, registration forms, warranty forms, call reports, customer service in-

quiries, and Web site data. Data warehousing enables the firm to organize and store this data for analysis purposes. By careful analysis of this and other data, firms can design more effective and efficient ways to serve their customers. Data warehouses exist to support the decision making process by providing ready access to market and customer data.

The Internet has shifted power away from marketers to customers by lowering search costs and providing greater choice. Competitors are only a "click" away. Consequently, marketers need to work smarter to create and manage the one-to-one relationships that customers desire. This goal requires increased knowledge about customer preferences, behaviors, and value expectations in order to create better products and services. Conversely, marketers need to understand which customers are the most valuable to them over time. They need to regularly monitor and evaluate the "lifetime value" of each customer in order to determine which customers to continue investing in and which ones to drop.

a. Data Warehouse Processes

Data warehouses can provide narrow or broad strategic views of key marketing activities for the purpose of generating higher customer value and business returns. Data warehousing activities commonly support customer relationship management (CRM), product development, and customer service delivery. Data warehouse architectures are based on the following three processes:

1. *Data collection.* Processes need to be developed for capturing data from the appropriate internal and external sources and then organizing, verifying, integrating, and otherwise preparing it for loading into the database.
2. *Data management.* This step involves formatting and storing that data for easy access by data warehouse users.
3. *Data access.* This process involves specialized tools to query the data, analyze it, and create and distribute useful business reports.

Data warehouses are an integral part of the MDSS. They provide the ability to access data for creating marketing operations reports; analyzing sales results over time; identifying and mapping patterns and trends that may be emerging in the market; and enabling the development of new products, pricing, market segmentation strategies, marketing communications campaigns, and distribution channels.

D. Data Marts

A data mart is a scaled-down version of a data warehouse. It is more focused, less complex, and holds a subset of the entire data, often in summary form. It is usually designed for a smaller number of users. Data marts provide fast, specialized access and applications. They are sometimes call departmental data warehouses.

Data marts are useful when it is not feasible for a data warehouse to meet the needs of all potential users. Data marts enable a more limited number of users to exert greater control over the data they need. Data marts enable increased data access speed and minimize preemption of user queries. They also minimize performance sacrifices inherent in large data warehouses by enabling small groups to get the data they most need when they need it. They may also include specialized data sets that can be analyzed with statistical or data mining tools. Not all the data in the data mart need come from the data warehouse. A marketing researcher may overlay customer purchase histories from the data warehouse with geographic information system (GIS) data from a commercial service and store it in the data mart.

E. Data Mining

Internet-based marketing strategies generate extremely large data sets from customer interactions. Purchase histories, financial records, customer service records, and Web site usage are just some of the data that reside in customer databases. In order to transform this mountain of diverse data into operationally useful information, marketers are increasingly using *data mining* procedures. Data mining is the computer-based exploration and analysis of large quantities of data in order to discover meaningful patterns and rules for the purpose of improving marketing, sales, and customer support operations. The combination of data mining procedures with data warehousing enables the MDSS to move beyond just support for the operational processes in the marketing organization and to focus on actual customer behavior. Data mining and data warehousing provide the means and the infrastructure for extracting strategic opportunity from knowledge of the customer.

a. The Data Mining Process

Large, multinational organizations produce much more marketing data per day than its managers can assimilate. The Internet facilitates the rapid growth of

data on a worldwide basis. However, exponential growth of data can, paradoxically, lead to a situation where more data leads to less information as managers become swamped by the flood of data that defies ready interpretation. Marketers need to develop procedures for processing, filtering, and interpreting this data for strategic marketing purposes. Data mining is essentially the engine for a knowledge-based marketing strategy. It provides the ability to collect, process, disseminate, and act upon information more rapidly than the competition which is essential for the creation of first-mover advantage.

The first step in the process is to collect data on what the customer does. On-line transaction processing (OLTP) systems do precisely that. Virtually everything a customer does when purchasing a product or service generates a string of transaction records. If the customer calls an "800" number to order a product, the phone company will capture data on the time of the call, the number dialed, and the duration of the call. The marketing company will generate similar data in addition to that on products and services purchased, catalog referenced, special offers, credit card number, order size, and time since last purchase. Further transactions are generated by the order entry, billing, and shipping systems. The bank and the shipping company will log further transactions. The customer may need to call customer service to solve post-purchase problems. Internet transactions can generate even more data as the customer's purchase behavior can be linked to Web-browsing behavior within a site and throughout the Web. This data can then be linked to purchase histories, financial history, and other personal-identity information.

b. Data Mining Tasks

After the data have been collected and reside in the data warehouse, approaches to analyzing the data are considered. Data mining methodologies may encompass a range of approaches from rigorous scientific methodologies and hypothesis testing to qualitative sifting through massive amounts of data in search of relationships. The type of analysis typically is a function of the task the researcher wants the data mining exercise to accomplish. Typical data mining tasks include:

1. *Classification.* A predetermined classification code is assigned to a database record. Decision tree analysis techniques are commonly associated with this task.
2. *Estimation.* Input data are used to estimate continuous variables such as age, income, and likely behaviors. Neural networks are often used for estimation.
3. *Affinity grouping.* Rules of association are developed from the data and used to group variables that seem to go together. Market basket analysis is a preferred technique that analyzes the linkages between items consumers buy in a basket of items.
4. *Description.* Summary observations are made about the data that serve to increase the understanding of the phenomena that generated the data. Description often motivates further research and data analysis. Market basket analysis, query tools, and visualization (mapping) techniques are commonly used.
5. *Clustering.* Clustering is used to segment a large heterogeneous population into homogeneous clusters based on measures of similarity. The researcher must determine the meaning of each cluster. Clustering algorithms are used to analyze the data.
6. *Prediction.* Records are classified based on predicted future values or behaviors. Neural networks, market basket analysis, and decision trees are common techniques.

c. Data Mining Techniques

Researchers looking for behavioral insights in large customer databases commonly use the following data mining techniques:

1. *Market basket analysis.* This analysis searches for associations in the data such as professional women who drive 5-Series BMWs also use Web-enabled mobile phones. The weakness of the technique is that there are an infinite number of possible rules in any given database and only a few have marketing importance. The problem is to find them. Techniques work best when the researcher has an idea of what to look for.
2. *Cluster analysis.* This analysis is based on the hypothesis that customers of the same type will exhibit similar behaviors. K-means cluster analysis is the most common method. The purpose is to assign objects to groups that are relatively homogeneous within and heterogeneous between.
3. *Decision trees.* Decision trees enable classification for directed data mining. Simple rules are often used to divide the data into subsets in which key attributes can be more readily evaluated. Decision

trees can facilitate prediction by linking customer characteristics with purchase behaviors.

4. *Query tools.* Structured query language (SQL) is often used for conducting a preliminary analysis of a data set. Data summaries such as simple averages, frequencies, and cross-tabulations are useful for looking for patterns and rules that may form the basis for more structured analyses.
5. *Neural networks.* Neural networks are a class of tools that are used for classification, clustering, and prediction. These networks are computer models that simulate the neural connections in the human brain. There are two critical stages for using neural networks: the encoding stage where the network is trained to perform a task and the decoding stage where the network executes the assigned task. In practice, these machine learning tools are used to identify loyal customer clusters, find fraudulent credit card transactions, diagnose medical conditions, and predict the failure rate of aircraft engines.

F. On-Line Analytical Processing (OLAP)

OLAP is a family of analysis and report-generating tools that are used to access large databases. It enables partially aggregated data or full reports to be stored in a multidimensional format for fast, convenient access and analysis. OLAP methods are based on databases that allow multidimensional views of business data. OLAP is useful for visualization of relationships between predesignated variables.

OLAP applications are used to achieve a higher view of the data such as total sales or profitability by product line, sales territory, or market segment. The OLAP database is usually updated in batch mode from multiple sources. OLAP is optimized for analysis and reporting. In contrast, users of on-line transaction processing (OLTP) applications are involved with creating, updating, or retrieving individual customer records. OLTP databases are optimized for updating transactions.

An OLAP system essentially stores answers to predefined business questions for report generation needs. The user can choose from a predetermined set of options for types of data and display formats. Output is typically in the form of charts, graphs, tables, or maps. OLAP solves the problem of distributing information to large numbers of users with diverse reporting needs. OLAP relieves the long response times that can be encountered when many users need to repeatedly query large databases for extended periods of time.

Time series data are probably the most common dimensionality in an OLAP database. Marketers want to look at trends in all aspects of the business—sales trends, market trends, pricing trends, profitability trends, etc. OLAP can compare current results with prior periods, calculate year-to-date results, and present other comparative historical data. OLAP can also present multiple hierarchies and classes within given dimensions. For instance, data may be drilled down by "state-county-city-customer" or "sales region-sales district-sales person-customer."

OLAP may be used in conjunction with data mining, but it is not a substitute. OLAP tools are powerful and fast tools for generating reports on data, whereas data mining tools find patterns in the data. OLAP users are constrained in the questions they can ask, since OLAP only answers the questions that the data formats were designed to address. They cannot go back to the original data and search for new solutions. Therefore, data mining is more powerful than OLAP.

G. Geographical Information Systems

GIS systems enable marketers to geographically map their customers, competitors, suppliers, sales concentrations, prospects, suppliers, and partners. Site selection, trade area analysis, environmental analysis, territory design, network planning, and risk analysis are common applications. Newer systems have integrated global positioning system (GPS) capability for location reports from resources in motion such as mapping transportation fleet movements, sales representative reporting, and locating and managing key assets in real time. GIS data provide powerful new visualization opportunities for customer data that can link consumer behavior to a fixed location at a specified time.

a. Geographical Information System Marketing Applications

1. *Customer location.* Links behavioral data from customer master files, subscription lists, support and warranty claims, transaction history, and identity with time and location information. This is very powerful information for mapping and predicting consumer behavior. The advent of mobile e-commerce will enable marketers to identify and map consumers at the actual point of purchase. The wireless carriers will be able to provide this data.

2. *Geographic market information.* Links marketing data to physical maps. Data may be classified by county, city, zip code, census tract, etc.
3. *Marketing activity location.* Links POS transactions, distribution patterns, direct response results, sales forecasts, advertising expense, etc. to geographic location.
4. *Business location.* Labels business facilities on a map that can display retail density, population density, buying power, media coverage, etc.
5. *Marketing resource location.* Links assets in motion to physical location through GPS from trucks, autos, aircraft, and wireless devices.

III. CUSTOMER MANAGEMENT SYSTEMS

Companies need a method for viewing all customer- and marketing-related information in an integrated way. Often, marketing organizations maintain multiple databases for each business and marketing activity with data that is not easily integrated for strategic or operational purposes. A new generation of software that is Internet based gathers information from customer service, Web sites, direct mail operations, telemarketing, field sales, customer service, distributors, retailers, and suppliers for the purpose of managing marketing, sales, and customer service activities. The major applications families are commonly referred to as sales force automation (SFA) and customer relationship management (CRM) systems. Some CRM systems are fully integrated with SFA applications and some are stand alone. The worldwide market for such systems is projected to grow five times faster than the overall software market, from $5 billion in 1999 to more than $22 billion in 2003.

A. Sales Force Automation

SFA is a customer management tool that is one of the fastest growing elements of the MkIS. SFA applications are often integrated with the CRM system. SFA involves the application of information technology to the sales function or, more appropriately, to the activities leading to a sale. These activities include acquiring sales leads, managing the sales opportunity, closing the sale, and managing the customer relationship.

The historic role of personal selling has been to move the product—to generate transactions. As selling has become increasingly more professional, salespeople emphasize building relationships with customers that will generate loyalty-based repeat transactions over time. Relationship building often necessitates that the salesperson has consultative and advisory skills in addition to product knowledge and sales abilities. Team-based selling places emphasis on role specialization, collaboration, and coordination. Customers have become more sophisticated as well. Requirements for customized solutions, rapid response times, and the need for concurrent and postsale service have greatly increased the need for information technology in the sales process.

The goals of SFA are to increase the effectiveness of the sales organization, improve its efficiency, and create superior value for the customer. *Sales effectiveness* focuses on getting the sale by improving lead generation, qualifying prospects, coordinating sales efforts, and tracking commitments. Effectiveness is a function of improving the sales process. *Sales efficiency* is evaluated by measuring the return on sales efforts. SFA can improve sales efficiency by reducing sales cycle time, by managing workflow, and by tracking the current status of critical activities related to the sale. Proposal generation, opportunity management, fulfillment, and follow-up are facilitated by SFA. *Superior customer value* is achieved when the customer expectations are exceeded. It is a primary determinant of customer satisfaction. SFA enables better understanding of customer expectations and management of the customer account.

a. The Sales Process

The typical field sales process consists of a series of steps that are designed to lead to a sale. The typical role of the MkIS is to support the sales process steps of lead generation, sales process management, and account management.

1. *Lead generation.* Leads represent potential customers. The identification of a lead is the beginning of the sales cycle. After leads are identified, they must be qualified. This process involves gathering information about the lead and comparing the result against qualifying criteria. The lead generation process is becoming more automated with regard to obtaining more prequalifying information directly from the lead and augmenting it from commercial and other databases such as credit bureaus. The advent of Web-based technologies is driving this trend. Qualified leads are then distributed to the sales force.
2. *Sales process management.* This process starts when the salesperson receives the lead information.

The primary information system need is for a convenient method to track the process and store the data generated at each stage. There are a number of substeps to this stage of the process.

a. *Verification of the opportunity.* The salesperson usually contacts the lead and attempts to verify the existence and nature of an opportunity including its size, timing, and appropriateness of the products and services of the selling company. Salespeople will also desire to verify the lead's ability to purchase, identify the names of key decision makers and influencers, and the level of budget authority. This information is entered into the sales database.
b. *The sales call.* If the lead is amenable and the opportunity justifies it, a sales call is scheduled. Information may be sent to the prospect and a custom presentation may be created. The SFA system is used to provide a single point of interface for the salesperson to coordinate the activities leading up to the sales call. After the call, the SFA system is updated with customer requirements, new information, and commitments made by the salesperson.
c. *Opportunity management.* If the salesperson is successful, the next step is typically the receipt of a request for proposal (RFP) from the prospective customer. The RFP will generally state the customer's requirements and the date for the final submission. Follow-up visits may be necessary to clear up or identify new requirements. The prospect may want to visit the seller's manufacturing site. The final product of this stage is the creation of a proposal and price quote to be presented to the prospect. This document should build a sound economic case for the purchase. All these interactions, requirements, commitments, and competitor information are tracked by the SFA system that provides the database, tools, and templates to generate the proposal.
d. *Closing the sale.* The presentation of the proposal and the price quote is the start of the closing process. Even if the salesperson has done a good job of presenting the business case, further negotiations may be necessary to close the sale. If all objections are met and the proposal is accepted, the close is successful. If not, the sales team will need to debrief the lessons learned to determine why the proposal was not accepted. The outcomes are recorded in the SFA system.

3. *Account management.* The automation of the sales process may result in a stand-alone system that is not integrated with other management systems. However, SFA is increasingly being integrated with the overall CRM system. This is especially true of the account management function. Since the primary goal of the CRM system is to manage the customer relationship in order to generate repeat sales, a good salesperson will want to keep track of the status of a new account and how well the customer is being served. This is especially true if the account is to become "referenceable" to other prospects. The account management function of the CRM system enables the salesperson to track order entry, order processing, shipment, and installation. The salesperson may need to ensure that postsales service is delivered or to monitor the results of installation and track the customer's satisfaction. The ability to track outcomes and interact with the customer in order to reassess needs and create new opportunities is a major benefit that flows from effective account management.

b. Sales Force Automation Tools

SFA tools consist of software applications that enable the salesperson to better target sales opportunities and manage the sales cycle. The tools are increasingly available bundled as integrated suites that are Internet enabled, accessible through a browser, and linked to the CRM system. The software applications may be categorized as:

1. *Document management tools.* These tools support all stages of the sales process. They include word processors, graphics programs, spreadsheets, e-mail, expense reports, proposal generators, and "product configurators." A Web-accessible sales library or encyclopedia of previously developed product information, brochures, product demonstrations, presentations, financial information, price lists, white papers, and public relations materials is a key resource for sales force productivity.
2. *Personal management tools.* These tools focus on increasing the efficiency and effectiveness of the sales teams efforts. They typically include calendar and scheduling programs, contact management systems, and call reporting capabilities.
3. *Process management tools.* These tools are used to keep track of customer requirements and sales

commitments. They include opportunity management systems, project management systems, account management systems, order-entry systems, telemarketing systems, and team-selling systems.

Increasingly, with the rise of Web-based hosting and the application service provider (ASP) industry, SFA applications and databases are being hosted by third-party specialists and accessed remotely on the Internet. With the ASP model, client companies are essentially outsourcing all or part of there information system function. Remote hosting raises issues of security and scalability. The advantages of an ASP approach are its browser-based simplicity, rapid implementation, and lower cost of deployment. Most ASP applications use a subscription-based pricing model. Some vendors are proposing that renting the application or paying by the transaction may become the pricing model of the future.

B. Customer Relationship Management

The Internet has facilitated a fundamental shift in market power from sellers to buyers. Customers, newly endowed with the power of market information, have much different expectations than before. Customers can easily move their business to another vendor with the click of a mouse. They have access to the same cost data as their suppliers and they demand 24/7 customer service. Customers now have almost unlimited ability for interactions with organizations through the Web in addition to the traditional phone and mail methods. Company Web sites facilitate information search, shopping, and customer support. E-mail communication can target specific offers on a one-to-one basis. Understanding and meeting demanding expectations has placed a renewed emphasis on managing customer relationships.

The primary goal of CRM systems is to increase the return on marketing expenditures by enabling the understanding of the complete history of a firm's interactions with its customers. CRM applications can deliver targeted solutions that promote customer loyalty as measured by increased response to promotions, purchase frequencies, and volume and that minimize the time between orders. CRM systems are able to target marketing communications to likely buyers, facilitate sales efforts, and deliver customer service. CRM systems increase revenue, lower costs, and optimize customer lifetime value.

a. Customer Relationship Management Design Principles

The CRM field is rapidly evolving into an integrated discipline that manages all of a company's touches with the customer. Accordingly, several design principles have evolved as the foundation for CRM development:

1. The CRM system should offer the customer multiple channels for communication, such as the telephone, e-mail, fax, on-line, or some combination. Each channel should lead to an interaction that satisfies customer expectations. Customers must be able to choose the method that best meets their needs on each occasion.
2. Each CRM interaction should deliver value to the customer. The CRM system must be able to determine what value is required and deliver it quickly whether it is a product or service transaction or a customer service request.
3. The CRM system integrates the customer throughout the firm's value chain. Customers should be able to reach into all necessary functions of the organization, not just to the sales organization or customer service, but to manufacturing and even the CEO. Better integration of the customer into the process can lead to higher levels of trust, loyalty and repeat purchases. The cost of acquiring a new customer often far exceeds that of retaining existing ones.
4. Knowledge is captured during each CRM interaction. The CRM system should capture knowledge about the customer and the customer's relationship with the company over time. Sales and customer support histories should be analyzed for indications of how well the vendor is doing in meeting customers' needs and how valuable the customer is to the marketer.

b. Customer Relationship Management Functions

CRM applications provide the customer-related information that is necessary to drive a firm's e-business activities. Better customer information enables more effective demand forecasting, product launches, and marketing campaigns. CRM functions are sometimes referred to as the company's "front office." CRM functionality includes:

1. E-commerce support
2. Sales force automation

3. Telesales and call center automation
4. Direct mail and catalog sales
5. E-mail and e-newsletter response
6. Web sales and personalization
7. Analysis of Web generated data
8. Traditional customer support and service
9. On-line support and customer service
10. Mobile support through laptops, handheld devices
11. Training

Modern CRM systems are fully integrated with the "back office" elements of the ERP system, such as accounting, manufacturing, project management, and human resources. This integration enables the sharing of customer information that can impact the operations of the enterprise and vice versa. Common customer definitions, price lists, employee definitions, service requests, call histories, order histories, contracts, and service level agreements are accessible by all who need them. Correspondingly, marketing can access information on items such as new product development, product costs, order status, delivery dates, and backorders.

IV. PRIVACY AND THE MARKETING INFORMATION SYSTEM

Privacy in the Internet Age has become a major concern for customers and marketers alike. Web-enabled marketing information systems have the potential to collect highly sensitive and voluminous information on customers, competitors, suppliers, and channel members. The speed in which this information can be collected, analyzed, and acted upon can provide a real competitive advantage to the user. This information can be used to better serve customers, streamline a company's operations, and develop better strategy. Some marketers use the data to better target advertising and marketing plans. Others sell or trade the data with other Web sites. Unfortunately, guidelines for how this information is collected, how it may be used, and how it is to be protected are not widely accepted.

Since most non-information system people use privacy, confidentiality, and security interchangeably, it is important to clarify some definitions before moving on. Privacy, as marketers define it, is the right not to be contacted or marketed to unless the customer has granted specific permission or "opted in." Security means keeping unauthorized persons from gaining access by breaking into the information system. Confidentiality is defined as maintaining appropriate controls on information so that only those authorized can access or use it. A marketer may respect a customer's privacy and only contact him or her after gaining permission, but they may have an insecure system that allows hackers to access credit card transaction data. Alternatively, a marketer may respect privacy rights and have a very secure system, but inadvertently releases a patient's health care record to the news media. In order to avoid these and other pitfalls, marketers must act to increase system security, establish confidentiality procedures and training, and implement privacy policies. The remainder of this section will focus on privacy issues as a representative case.

A. Tracking Internet Users

It is axiomatic that no one is anonymous on the Web. At the current state of technology, marketers are able to develop customer profiles and track their behavior over the Web. Soon marketers will be able to track the locations and behaviors of Web-enabled cell phone and other wireless device users. The public uproar over such capabilities has caused several Web advertising networks to abandon efforts to link Web-use profiles to personal information.

In order to track consumers on the Web, enabling information system infrastructure has been developed that links a surfer's Web behavior to his or her Internet protocol (IP) address. Once that occurs the information gathering begins. The following infrastructure elements are essential to the information model of the Internet:

1. *Advertising networks.* Web-based advertising agencies such as DoubleClick, Engage, and 24/7 create profiles of Web surfers by tracking their on-line behaviors. Ads are customized and presented on-line to those most likely to buy when they visit supported Web sites.
2. *Cookies.* A tiny text file is placed on a consumer's hard drive when they first visit a site. It allows the site to track the user's behavior on the site. Most Web users are not knowledgeable about cookies. If a consumer removes the cookies from the browser, most membership-based sites will prompt for user names and passwords and install another cookie.
3. *IP address.* A number used to identify a consumer's personal computer so that Web data can be sent. These addresses enable profiling and

ad targeting. IP addresses can be associated with behavioral and personal information.
4. *On-line profiling.* Web sites build consumer profiles of page viewing, time spent, and shopping behavior by combining cookies with other information. This sounds ominous, but the ostensible reason is to provide better automated service.
5. *Personal information.* This includes the consumer's name, address, credit card number, driver's license number, social security number, and consumer behavior that is linked to customers personal identity. Most consumers are still reticent about e-commerce because they fear for the safety of their personal information. Identity theft is a rising problem.
6. *Referers.* This is information that a Web browser passes along as a surfer moves from site to site. Referers are commonly collected and used to target advertising.
7. *Registrations.* This is personal information that is collected when a consumers fills out a sign-up form on a Web site. Data can be sold or shared. Many firms do not closely follow their stated privacy policies with regard to selling customer data.

B. Developing a Privacy Policy

Forward-looking Web marketers are at the forefront of the consumer privacy movement. Advertising network firms, such as DoubleClick, were caught on the wrong side of this issue and have suffered greatly in terms of company image and market capitalization. There is a growing realization that Web marketers should post disclosure notices on what information a company collects and how it is used or shared. Currently, privacy policies vary widely from marketer to marketer and there is no monitoring of their effectiveness.

Concerned marketers are reacting to privacy concerns in an attempt to ameliorate consumer fears and to head off potential government regulation. The major actions marketers are taking include:

1. *Disclosure of privacy practices.* Smart marketers should clearly and prominently explain what information is collected, how it is tracked, how it is used, and under what conditions it is disclosed to partners. New technology may help solve the privacy problem. Soon the P3P software standard will allow consumers to set privacy preferences in their browsers that will alert them when a site's privacy policy differs from their desires.
2. *Ask permission.* Web users should be able to opt-in, especially if the data to be collected is sensitive. Sensitive data include medical information, financial information, insurance claims, and consumer behavior that can be traced to the individual by name. Opt-in means that the site cannot collect personal information unless the user gives permission. Opt-out is a default situation where the site collects personal data and is free to use or sell it unless the consumer specifically denies permission.
3. *Allow access.* Consumers should be allowed access to the personal data each site has collected on them in a manner similar to the access they have to their credit report. Perhaps this can be accomplished by enabling consumers to obtain the aggregated profiles the advertising network firms have constructed. Consumers should be allowed to correct or delete inaccurate data.
4. *Establish industry-wide privacy rules.* The On-line Privacy Alliance is an association of leading global companies and trade associations dedicated to the protection of privacy on-line. They are working to educate on-line businesses on the need for a standardized privacy policy to inform customers about what information is being collected, how it is used, how it is protected, and how people can access the information to prevent errors and how to opt-out.

V. CONCLUSIONS

The rapid adoption of Internet-based technologies and the attendant development of e-business and e-commerce applications are having a revolutionary impact on the marketing discipline. Marketing information systems, in particular, are being transformed as these new technologies are enabling the integration of marketing, sales, and customer service activities. The primary drivers of this shift are the promises of delivering increased value to the customer more rapidly and at less cost. Future implementations of MkIS will increasingly involve the customer in the value creation process and work to more effectively align the enterprise and its supply chain on rapidly changing market opportunities.

SEE ALSO THE FOLLOWING ARTICLES

Advertising and Marketing in Electronic Commerce • Cost/Benefit Analysis • Data Warehousing and Data Marts • Electronic Commerce • Electronic Payment Systems • Enterprise Resource

Planning • Geographic Information Systems • Marketing • On-Line Analytical Processing • Procurement • Research • Supply Chain Management • Transaction Processing Systems

BIBLIOGRAPHY

Berry, M. J. A., and Linoff, G. (1997). *Data Mining Techniques for Marketing, Sales and Customer Support.* New York: Wiley.

Bidgoli, H. (1997). *Modern Information Systems for Managers.* San Diego: Academic Press.

Hansen, W. (2000). *Internet Marketing.* Cincinnati, OH: South-Western Publishing.

Kachur, R. (2000). *Data Warehouse Management Handbook.* Paramus, NJ: Prentice Hall.

Khandpur, N., and Wevers, J. (1998). *Sales Force Automation Using Web Technologies.* New York: Wiley.

Kotler, P. (1997). *Marketing Management: Analysis, Planning, Implementation, and Control,* 9th edition. Upper Saddle River, NJ: Prentice Hall.

Mougayar, W. (1998). *Opening Digital Markets.* New York: McGraw-Hill.

Shepard, D. (1998). *The New Direct Marketing.* New York: McGraw-Hill.

Shim, J., *et al.* (1999). *Information Systems Management Handbook.* Paramus, NJ: Prentice Hall.

Web Resources

Business Geographics Magazine. Portal site for GIS systems information and suppliers. www.bg.geoplace.com.

CIO Magazine. Enterprise resource planning center, Online ERP information resources. www.cio.com/forums/erp/.

CRM Magazine. Leading CRM industry publication. www.CRM-Magazine.com.

CRM Community. Web community portal for customer relationship management. www.CRMCommunity.com.

Privacy Alliance. An alliance of over 100 leading businesses and trade associations dedicated to protecting privacy on-line. www.privacyalliance.org.

Supply Chain Council. Portal site for supply chain management information, conferences, and solutions. www.supply-chain.org.

Medicine, Artificial Intelligence in

Horia-Nicolai Teodorescu
Romanian Academy and University of South Florida

Abraham Kandel
University of South Florida

GLOSSARY

assistive devices A device used to assist impaired people in daily life activities.
bedside monitor Equipment including various units used to monitor the vital signs, like respiratory and cardiac signals of a patient under intensive care or medical surveillance.
chaos A type of dynamical regime of deterministic nonlinear systems characterized by apparent randomness of the behavior; unpredictability.
electrocardiography A set of methods to determine the electrical activity of the heart and to diagnose abnormalities of heart operation; the medical field including the above.
electroencephalography A set of methods to determine the electrical activity of the brain, to determine the state of the patient, and to diagnose brain activity abnormalities.
fractal dimension Generic term designating a family of measures of the complexity of the nonlinear dynamic behavior and of fractal objects.
intelligent system Any system that includes some form of artificial intelligence, typically able to adapt, learn, and evolve.
nonlinear dynamics theory Theoretical domain studying the dynamic behavior of nonlinear systems, especially the dynamics that are nonperiodical.
rehabilitation A compound of methods used to restore to a normal or closer to normal state an impaired person.

A large range of theoretical and software tools, based on artificial intelligence (AI), have been developed and various methods have been adapted to improve the diagnosis, treatment, rehabilitation, prevention, screening, and hospital management. In this article, we overview the major applications of AI in the medical and bio-medical engineering fields. We present the use of the methods and techniques of knowledge-based systems, decision-support systems, expert systems, neural networks, fuzzy logic systems, evolutionary algorithms, data mining, virtual reality, robotics, pattern recognition, and other artificial intelligence techniques. AI applications in medicine range from classical domains, such as electrocardiography and electroencephalography, to automated diagnosis, medical knowledge discovery, medical image bases management, and intelligent techniques in rehabilitation.

The specificity of the AI applications in medicine is addressed in the first section. In the second section, we review the employment of the main AI techniques adopted in medicine, while in the third section we briefly deal with the AI applications to several major medical fields, as seen from the medical perspective. In the last section we briefly discuss the perspectives and present a few conclusions.

I. SPECIFICITY AND CHALLENGES FOR AI IN MEDICINE

A. Background

AI-based systems, frequently named intelligent systems, have witnessed a rapid growth in medical applications during the last two decades. An impressive number of devices and medical equipment, not to mention the number of papers and volumes published in this field,

have been issued. Now, AI-based systems represent a powerful tool in all medical fields. The applications of AI in medicine evolved on both the horizontal dimension, with a constantly increasing number of types of equipment, and on the vertical dimension, by including more capabilities ranging from intelligent sensors to neuro-fuzzy systems, intelligent agents, image automated analysis and recognition, and decision-making and expert systems.

AI and intelligent technologies started to make inroads in medicine several decades ago, when medical practitioners recognized expert systems and knowledge-based systems as a valuable asset. The threshold level was passed when general practice equipment, such as electrocardiographic (ECG) equipment, started to include various forms of pattern recognition and signal classification abilities, as well as "diagnosis advice" facilities. Today, tomography is unconceivable without some form of intelligence in dealing with images, while information systems used in hospitals "intelligently" perform many tasks of searching, classification, data compression, and data mining. The main dimensions of medicine, namely prevention, diagnosis, therapy, surgery, and rehabilitation, have all extensively benefitted from AI techniques.

The degrees of intelligence imbedded in the medical equipment may vary from adaptive control to the extensive use of medical knowledge, pattern recognition, learning, and self-evolving abilities. As the medical technology evolves, so does the meaning given to "intelligent equipment" and no consensus exists on the nomenclature. "Intelligent systems" and "intelligent technologies" are not satisfactorily defined terms. As in many new fields, the meaning of the terms and the coverage of the field are continuously evolving. Therefore, we shall adopt here "inclusive" definitions and boundaries for the related fields. In this article, we discuss applications involving at least an adaptability capability performed through a knowledge-based system, a neural network, or a similar new, essentially nonlinear technique. Most advanced medical equipment today may involve adaptive control, domain-related knowledge, data fusing capabilities, inference ability, pattern recognition (in bio-electric signals, images), decision-making, learning capabilities, and elementary self-evolving abilities.

B. Complexity of the Domain

The living human body and its functions are extremely complex. To their understanding concur various classic and newer, interdisciplinary sciences. Medicine is plagued by huge amounts of data, large knowledge bases, incompleteness, imprecision, inconsistency in data and knowledge, various ways of reasoning, including qualitative reasoning, heuristics, and subjective reasoning, and decision-making processes that escape classical reasoning. Modeling parts of the human body is still at its infancy and there is no model, except possibly a few mechanical models, that is close enough to the modeled part of the human body. There is a definite lack for good models in many areas of medicine, yet it is supposed that diagnosis and treatment are based on good models. Moreover, medicine extensively uses system identification, system control, decision-making, and reasoning, all performed by human experts in ways that frequently escape the existing theoretical frameworks. Because medicine has not completely—if it ever will—evolved from a qualitative to a purely quantitative domain, and because of the complexity of its tasks, AI-based systems find a vast application area in medicine. Methods that include both qualitative and quantitative representations, such as fuzzy logic and neural networks, fill the gap between the needs of the medical science and traditional modeling and inference methods in precise sciences.

C. Data and Data Sources

In medical practice, data come from different sources, and the imperfections of the data may include ambiguity, uncertainty, imprecision, and incompleteness. In general, it is difficult to know the reliability of data sources; modeling the sources has been proposed and applied in various ways, most often by weighting or multicriteria preference ranking. In dealing with imprecise data, modeling the imprecision is also a disputable issue. The final goal of data sources evolution is to combine the available information to derive the best data.

D. Challenges for AI in Medicine

There are several challenges for AI in dealing with the complexity of medical problems, among others:

- The amount and diversity of data collected in medicine
- The uncertainty of data, for instance personal and family history information and clinical assessment of the patient by medical inspection, sound hearing, etc.

- The uncertainty of biological processes
- The lack of knowledge of biological processes
- The uncertainty of how medical doctors derive their knowledge, how they perform inferences, and how they perform decision-making

Under these constraints imposed on the use of the technology, it is not surprising that AI has produced specific developments in medicine to help cope with the complexity of the task. The AI community has also learned that medicine is more cautious and more traditional in many respects than other application domains.

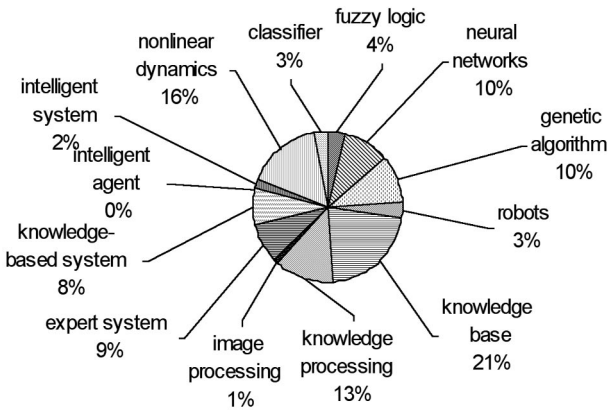


Figure 1 Representation of several AI techniques in medical research journal papers.

II. TECHNIQUES

AI techniques applied in medicine have attracted considerably different interest from the medical community. We will use as a measure of the medical community interest in a specific AI field the number of journal papers on topics related to specific AI techniques. More precisely, we use the number of papers reported in the MedLine database (see http://www.ncbi.nlm.nih.gov/PubMed/; throughout this article, statistics are based on MedLine database for the period 1951–2001). We prefer this index instead of the number of papers published in technical and computer science journals because not every paper in these journals has relevance to the medical community.

The degree of penetration of various domains of AI into the field of medicine differs largely, according to the number of published research papers. Overall, knowledge bases, knowledge-based systems, and knowledge processing dominate, with a combined 42%, followed by nonlinear dynamics with 16%, while fuzzy logic, classifiers, robots and intelligent agents are referred to in only a few of the papers (see Fig. 1.)

A. Knowledge Based Systems, Expert Systems, and Decision Support Systems

The introduction in 1972 of MYCINE, the first commercially available and successful expert system in the field, gave an impetus to the use of experts systems in clinics, as a support to the medical staff's daily activity. Expert systems, knowledge based systems, and decision-support systems are currently used in almost any field of medicine. The main issues in developing such systems for medicine are the acquisition of domain-specific data and knowledge, use of appropriate data representation, structuring and processing methods, incorporating appropriate expert models, and providing user-oriented explanations, as well as result assessment and the predictive power of the results.

Data and knowledge acquisition, management, and representation for the diagnosis support are essential tasks in all medical fields and in AI-based systems for medicine. The development of expert systems and decision-support systems asks for a great amount of medical data to be acquired and transferred to the machine. Input data may range from clinical to combined clinical, laboratory data, tests, and imaging results. Also, a large amount of medical knowledge has to be transferred from the human expert to the machine. The foundation of a medical expert system is the appropriate analysis of various medical aspects, including the way medical doctors are reasoning during the diagnosis and prognosis processes. The complexity and specificity of the knowledge—statistical, heuristic, imprecise—about various diseases and conditions raises difficulties in acquiring the expertise from medical doctors. Data and knowledge acquisition and structuring is a highly time consuming task, credited with up to 70% of the overall time needed to develop the expert system. The use of Internet platforms has been suggested and partly demonstrated for helping in this purpose. There are cases when an AI-based system must perform with a limited amount of data to minimize the cost of medical investigation, or because the data is unavailable. Moreover, data may be distributed between several information systems, as in dealing with hospital-acquired infections. To cope with this difficult task, hierarchical systems, based on several engines and data dictionaries, have been proposed.

All successful techniques for knowledge-based systems are applied to medical systems as well. However,

case-based reasoning has been considered only partly appropriate to the field and found fewer applications, possibly because generalizations from single cases to general rules are needed in medical domains. Also, intelligent agent-based approaches have not yet found a significant use in medicine. On the other hand, fuzzy logic proved attractive and great efforts have been devoted to the realization of fuzzy expert systems for medical use, including diagnosis, cure, surgery, and preventive medicine.

It is considered that the success of diagnostic support and expert systems highly depends on knowledge representation, while the quality of workbenches, reasoning explanation, and interfaces plays a major part in the acceptance by the medical community of such systems. Indeed, it is recognized that one of the essential problems in developing expert systems is to show *why* and *how* results have been obtained; this is particularly true for medicine. Moreover, these explanations must be provided in a way suitable for expert human thinking. For this purpose, the expert system must incorporate models of knowledge processing by the human experts. Providing clear explanations of the results obtained, in a way suitable for easy understanding by the expert, proves to be far from a trivial problem; in fact it may become more difficult than obtaining the result itself.

The assessment and validation of knowledge-based clinical expert and decision-support systems is a critical phase in the development of these systems. Inconsistencies, biases, and limitations in the practical application of the AI-based systems must be carefully evaluated before use. Clinical validations of the systems on large databases provide reliability indexes varying in the range of 60 to 90% of cases, depending on the system, the problem in hand, and the amount of data input to the system. Because of the limited reliability, the final decision is achieved under the user's responsibility. The system design and the methodology applied for upgrading such systems should benefit from the early stages of development from tests performed on the validity of the system response in clinical trials.

One of the problems faced by expert systems, decision-support systems and control systems is the prediction power. Related to this is the stability problem of the overall system, comprising the AI system, the practitioner and the patient or the population. This problem can be stated as follows: find the advice and the explanation to the practitioner such that the evolution for the specified patient or population in a given time frame is the best according to a given criterion. Most AI systems providing advice and explanations may be considered static, as they assume the explanation and cure process as a one-step process. Such systems are developed without any consideration for the stability of the medical care process in the long run.

B. Artificial Neural Networks

Artificial neural networks (ANNs) are nonlinear systems able to adapt by mimicking the natural neural processes. ANNs are able to perform nonlinear signal processing, feature extraction, pattern recognition, prediction, and memory processes. Therefore, they found extensive use in medical applications, including control, image processing and analysis, decision-making, and signal processing. However, the use of ANNs is plagued by the lack of a direct relationship between the weight values and the task performed; this lack obstructs building models that are interpretable by the medical expert. This is a serious issue in some medical applications, where results must be validated by the expert.

ANN-based techniques have received considerable attention for building prediction models from data because of their advantages over standard statistical methods, like their ability to model non-linear relationships. Signal prediction is a domain where the meaning of the weight values is not an issue, as signal samples have little meaning to the medical experts. Therefore, ANNs have found many applications in this field, from predicting electrocardiographic and electroencephalographic signal trends to predicting tremor movements. Also, risk assessment in various diseases, including survival prediction, is a field where ANNs have found extensive use.

Neural networks are a common way to deal in an efficient manner with pattern recognition functions, control, and advanced signal processing in prostheses and rehabilitation systems. Whenever the meaning of the processing is not an issue in the medical field, ANNs may help in developing parts of expert and decision-support systems too.

ANNs have been frequently applied in computational models of neurobiological theories and have helped provide insights into the operation of small parts of the nervous system. Apart from the explanation of several processes, like memory processes, visual perception, learning, and cortical maps, ANNs have helped correct theories in the medical field. This is one of the cases where cross-fertilization between medicine and AI is most manifest and productive. Also, recent developments in cognition sciences have

shown that ANN models may help understanding behavioral patterns, organization processes, and complex processes like learning, speech, and vision.

C. Fuzzy Logic, Fuzzy Systems, and Neuro-Fuzzy Systems

Fuzzy logic was developed in the 1960s and rapidly attracted the attention of the medical community and computer scientists involved in medical applications as a powerful tool in modeling imprecise reasoning and decision-making processes. Currently, fuzzy logic is extensively used in fuzzy knowledge bases, fuzzy expert systems, and image processing, while fuzzy systems are used in many control applications. Fuzzy logic and fuzzy systems demonstrated advantages as a developing method for nonlinear systems, and in combination with neural networks and appropriate training algorithms, fuzzy systems have been applied in adaptive signal processing systems, control systems, classifiers, prostheses, and rehabilitation systems.

A typical fuzzy system is composed of several elements that are specific to many knowledge-based systems. In the first place, it has an elementary knowledge base, including knowledge related to the variables, namely the related membership functions, a rule base, and an inference system performing the inference under various conventions regarding the logic operations. Moreover, a fuzzy system includes a defuzzification interface, which can be seen as an elementary decision block. Therefore, the architecture of a fuzzy logic system is similar to that of elementary decision-support systems. The ability to provide explanations on how the result has been obtained is a feature easy to implement for fuzzy systems, because of the one-step inference performed. In fact, many fuzzy systems have the facility of showing the rules applied to derive results and the graphical representation of membership functions as well. The flexibility of fuzzy systems and part of their popularity in many domains, including medicine, comes from the capability of performing logic operations (AND, OR, NOT) according to various types of logic; that is, implementing a logic that best fits the problem in hand. Gradualness of the truth values in fuzzy logic fits the needs of medical assessment and human expertise and encourages the use by medical experts who are accustomed to referring to belief and possibility degrees as much as to probabilities. Also a benefit is the ability of fuzzy logic—in fact, of any nonbinary logic—to deal with conflicting rules that are inconsistent under the binary logic.

Diagnosis and cure are currently employing fuzzy logic approaches to automate medical practice. Some fields of medical practice where extensive qualitative reasoning is not yet possible and where statistics are only partly relevant seem particularly suited to the use of fuzzy logic in an effort to go from almost completely empirical knowledge to qualitative reasoning, thus filling the existing gap. Typical examples of such medical-related domains are nutrition, psychiatry, and psychology, but almost any representation of the clinical data and diagnosis act involves a large part of qualitative data and inference that may be brought to a quantitative replacement using fuzzy representations. Patient monitoring and anesthesia control are two other tasks where fuzzy logic has been successfully applied by several teams. This topic will be dealt with as an example in Section III.B.

Neuro-fuzzy systems evolved as a fusion of neural networks and fuzzy logic and their use in medicine is rapidly growing in all applications today involving neural networks and fuzzy systems in an independent manner. The advantage of neuro-fuzzy systems consists in higher flexibility and increased modeling power. Neuro-fuzzy systems have also been demonstrated in modeling of natural neural networks.

D. Evolutionary Algorithms

Evolutionary algorithms, composed of genetic programming, genetic algorithms, evolutionary programming, and other similar methods, have recently made inroads in diagnosis—including differential diagnosis and diagnosis support systems—prognosis, drug discovery, data fusing, signal processing optimization, signal classification, and pattern recognition, control, and image processing. Moreover, hospital planning and scheduling tasks have started to benefit from these techniques. Genetic algorithms have also been used in optimizing fuzzy, neural, or neuro-fuzzy systems for data processing, pattern recognition, and control in medical applications. In adaptive systems, such as ANN-based or fuzzy logic based signal processing or control systems, evolutionary algorithms have the advantage of frequently working more reliably than classic adaptation methods, like gradient methods, moreover, they perform faster than algorithms based on classical statistical search methods. Evolutionary algorithms demonstrated excellent capabilities in knowledge discovery, with good predictive accuracy of the discovered rules and high accuracy rate for the overall rule sets. Critical aspects in their application are the choice of the fitness function

and the choice of the population. Most applications of this type have been reported in diagnosis. Although at the beginning of their use in medicine, these algorithms show a great potential in medicine.

E. Robotic Systems and Virtual Reality

Robotic systems gained extensive use in hospital laboratory automation, as well as several other applications like rehabilitation technology, assistive devices, and surgery.

Virtual reality (VR) methods found significant inroads into surgery planning, rehabilitation, and teaching in medicine. In these domains, VR tends to become a classic tool. The most popular applications today are by far teaching aids and surgery planning. However, virtual reality has demonstrated high potential in rehabilitation applications too. VR has been applied in several rehabilitation systems to decrease rehabilitation time, to help control postural balance, to help limit tremor movements, and to help recover from various disabling diseases. The addressed groups of people are motory impaired (either by handicap or by accident, including persons showing spastically moving limbs), visually impaired, or dyslectic people in controlling their environment and learning. Several projects have aimed to establish new methodologies combining virtual reality techniques and rehabilitation methods and to develop systems for helping the rehabilitation of specific groups of impaired people, namely those with motor troubles. VR techniques create a richer environment during rehabilitation sessions, provide feedback during the rehabilitation exercises, and introduce new feedback means. Such systems may offer tactile, visual, and sound feedback information in an integrated environment, allowing improved self-control by the subject during the rehabilitation training. Moreover, advanced processing of several signals from the subject allows the recognition of the state of the subject for adapting the rehabilitation training to the state of the patient.

VR systems must also embody vast knowledge bases on human movements, including movement skills and related cognitive tasks, for supporting haptic interfaces for interaction and other means related to the rehabilitation strategy implementations. The motor skill knowledge representation is unusual for classic AI applications, because of its specificity and complexity. The domain knowledge refers in this case to the human body kinematics, sensorimotor data, and sensorimotor knowledge representation, and it involves various models of the human. The problems faced by current VR system developments include improved representation of spatial objects, real-time computation of movements of the virtual objects, real-time monitoring of the patients, and efficiently controlling the remote objects that may be included in the system.

F. Nonlinear Dynamics Theory

Biological systems are reputedly nonlinear, memory-driven processes, with a large number of parameters, inputs, and outputs. They typically are composed of several subsystems. Therefore, simple linear theory does not match the demands of modeling and understanding biologic processes. Since the end of the nineteenth century, the development of the theoretical framework for nonlinear differential equations, nonlinear discrete processes, and fractal objects allowed for the understanding of several biologic processes in the second part of the twentieth century. Today, the common terms used to refer to this theoretical and application domain are nonlinear dynamics, chaos theory, and complexity theory (different from the algorithm complexity theory in computer science). Early modeled processes in the frame of the nonlinear dynamics theory are the prey–predator dynamic equilibrium and the logistic growth process. Now it is known that most equilibria in biological systems are dynamic equilibria and may develop patterns of evolution. While the number of patterns is limited, a precise state of the system is never reproduced and they are unpredictable in the long run, while they are predictable as patterns of evolution. The trajectory of a continuous-time system evolving in a chaotic manner is, in the state space, an open curve confined to a bounded region of the space. This contrasts with the trajectory of a periodical system, which is a closed curve. The trajectory has several other specific properties that have elicited the name of "strange attractor." Among others, it covers a bounded space region in a "dense" manner. Consequently, measuring the trajectory by covering it with infinitesimal space volumes will differ from the coverage of a typical curve. To make a distinction between the trajectory of such a system and the trajectory of a classic system, the quantity D named "(fractal) capacity dimension" is defined as

$$D_{capacity} = \lim_{\varepsilon \to 0} \frac{\log(N(\varepsilon))}{\ln(1/\varepsilon)},$$

where $N(\varepsilon)$ is the number of space elements with dimension ε needed to cover the trajectory. A closed

curve in the plane has the dimension 1, and a region of the plane has dimension 2, as expected. A chaotic trajectory in the plane may have the capacity dimension anywhere in the interval from 1 to 2.

Another salient property of the chaotic systems is that two trajectories starting very closely diverge exponentially. Because of their irregular evolution, it is not surprising that any small error in the knowledge about their initial position (condition) will develop large errors in determining the future state. This makes the prediction of chaotic processes virtually impossible in the details. The measure of divergence is provided by the Lyapunov coefficients. Briefly, the Lyapunov exponent λ determines the exponential increase in distance between two trajectories, according to

$$\lambda = \frac{1}{\Delta t} \log \frac{\delta(\Delta t)}{\delta_0},$$

where $\delta_0 << 1$ is the initial distance between two trajectories, and $\delta(\Delta t)$ is the distance between the trajectories after a time Δt. When $\lambda > 0$, the trajectories are said to diverge exponentially. A positive Lyapunov exponent proves the chaotic character of the process. Extensions of the Lyapunov exponent concept provide a series of Lyapunov coefficients that can be used to further characterize the chaotic behavior and to define another fractal dimension, the Lyapunov dimension. The property of trajectory divergence of the nonlinear systems, while the trajectories remain bounded, is known as *sensitivity to initial conditions* and makes it difficult to predict the evolution of such systems. Although such systems are unpredictable, various methods for chaos control have been devised to force specific patterns of behavior of the system, to stabilize the system, or to force it to follow a specified chaotic system.

Now nonlinear dynamics is employed in almost all fields of medicine. It found applications in explaining dynamics of the processes that appear at various levels in biological structures, including simple natural neural networks, the brain, heart, and speech. The understanding and modeling of the dynamics allow the experts to better characterize the behavior by means of the above mentioned fractal dimensions in view of diagnosis and to predict the processes in view of treatment, as in epileptic seizures. Nonlinear analysis has already been applied in forecasting cardiovascular activity, in assessing the risk for epileptic seizures, in assessing tremor movements, and in the early detection of several diseases, like schizophrenia. The attractor shapes may considerably differ between various diseases, despite lack of other indications in the analyzed signals. Determining the fractal dimensions for these attractors is evidenced in a quantitative manner by the differences. Although not yet generalized in diagnosis, there seems little doubt that nonlinear dynamics analysis will become an indispensable tool in the near future. Chaos control is also seen as a potential tool in withstanding critical heart activity by controlling it, while the tools of nonlinear dynamics theory help assess the correctness of proposed models for biological processes.

G. Data Mining and Knowledge Discovery

The amount and diversity of data in medical practice are huge. Data range from personal data and family history to working and living conditions, clinical observation data, physical examination data, and laboratory measurement results. Laboratory measurements include biochemical, molecular genetics results, imaging data obtained by various techniques (X-rays, ultrasonography, magnetic resonance, positron emission), and vital signals (cardiac and respiratory signals mainly and other biological signals). These data may refer to the present moment as well as previous time frames in the history of the patient when he or she was under monitoring, medical surveillance, or cure.

Knowledge discovery and data mining are two techniques that are intimately related. Data mining is a sum of methods used to discover and classify facts in large databases of heterogeneous facts. Data mining may use statistical methods, evolutionary algorithms, clustering, fuzzy logic, and ANNs. A special emphasis is on extracting facts from heterogeneous, often irregularly constituted data, where statistical methods may be inappropriate.

Data mining attracted interest in information retrieval in large medical databases, in genomics research, in drug research and assessment, in image processing, and in helping diagnosis (see Fig. 2). Also, data mining has found applications in identifying the most relevant signals to be acquired for the purpose of diagnosis, in the selection of the tests to be performed in view of diagnosis, in optimizing the electronic patient record, in identifying types of diagnoses and treatments that better fit specific cases, as well as identifying which plan can be generated and systematically developed.

An application where data mining shows promise is the reduction of data analyzed or stored in monitoring systems. Either in the bedside monitoring systems, where huge amounts of data are collected and the computational effort may be tremendous, or in portable monitoring systems which have an inherently

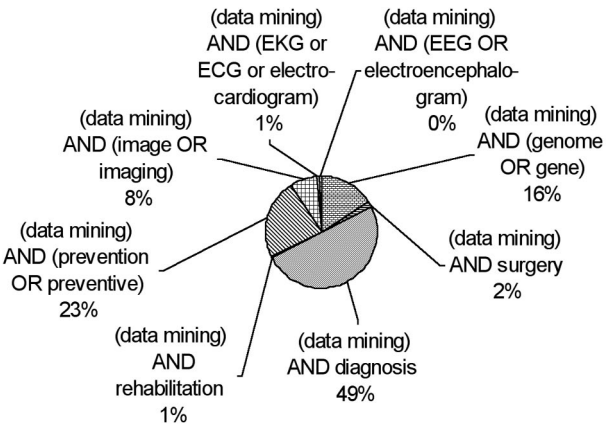


Figure 2 Applications of data mining in various medical fields.

limited power of data collection and analysis, finding the relevant data is an essential task. As the capabilities of the signal and image acquisition equipment increase, data dimensionality and correspondingly the multivariate analysis of the data becomes a huge computational task, increasing the needed storage, decreasing the retrieval of information speed, and increasing the overall costs. When the data processing must be performed quickly, as in intensive care medicine where the trends of a patient's state trends must be determined, the computational speed may also become an issue. In these cases, data mining may help by deriving the essential signals to be processed at a given moment of time and by determining the relevant patterns in the data.

III. APPLICATIONS

In this section, we review the impact of AI techniques on some of the principal fields of medical practice.

A. AI in Signal Processing and Analysis in Medicine

Signal processing and analysis are used in virtually all fields of medical practice, hence their treatment here as an independent section. There are significant differences between the application of various AI techniques to specific medical fields. To start with two classic domains, electrocardiography (ECG) and electroencephalography (EEG), the AI-type techniques most used in these domains are nonlinear dynamics related techniques, which lead with more than a third of the total number of papers quoting an AI technique. General AI methods follow closely, and neural networks are credited with 15%. This situation is explained by the importance of the nonlinear dynamic processes in heart dynamics, and by the successful use of ANNs in electrocardiographic (ECG) signal processing (Fig. 3.a). The situation is comparable for electroencephalography (Fig. 3.b).

One of the most successful applications of AI in medicine is medical image processing and analysis. Fuzzy logic, genetic algorithms, neural networks, knowledge bases, classifiers, and chaos theory all have been applied to improve image preprocessing, feature extraction, data compression, information retrieval, and image recognition (see Fig. 4).

B. Hospital-Based Care

While the general practitioner is frequently limited to the use of inexpensive decision-support systems, expert systems, and low-cost diagnosis tools, hospitals

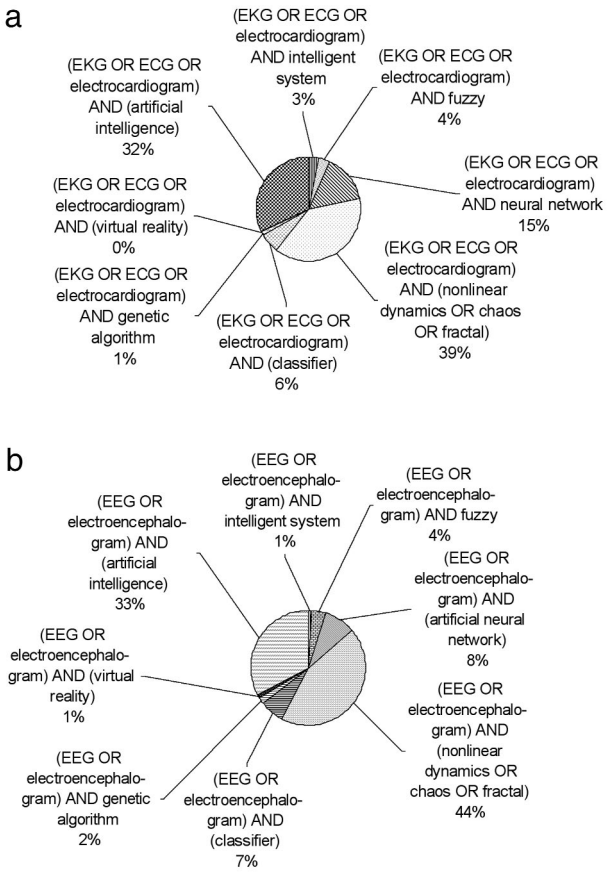


Figure 3 AI techniques in applications related to (a) electrocardiography and (b) electroencephalography.

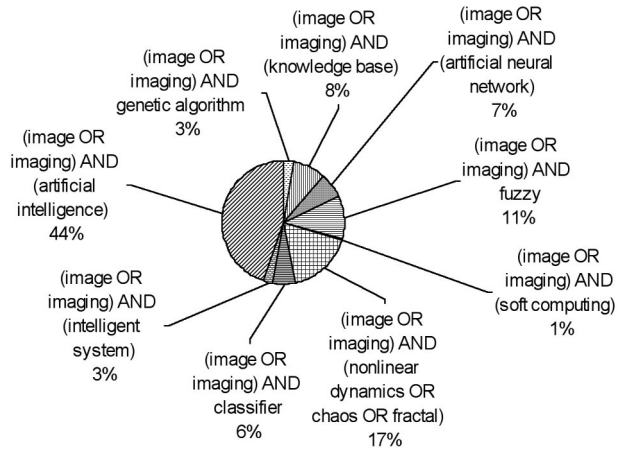


Figure 4 AI techniques in applications related to image processing, as reflected in medical research journal papers.

can afford expensive systems to help improve medical care and increase productivity. New technology has turned the hospital into a computerized system with data, knowledge user, and organizer. Image processing methods combined with tomography have revolutionized diagnosis. Classification and pattern recognition methods are used in many medical units. Some of the hospital units are more demanding than others, for instance the intensive care units (ICU), and require extensive use of intelligent equipment. Technology is at a point where much hospital equipment has some kind and degree of intelligence as well as intercommunication capabilities; we may expect that the ability to communicate with medical experts and patients will be added soon to many clinical units.

As an example, we briefly present here the use of AI techniques in a typical hospital application, namely anesthesia. As already mentioned, a popular application of fuzzy logic is fuzzy control. One of these applications is anesthesia and drug delivery control. The topic was approached in the early 1990s and has seen successive refinements. Anesthesia is a demanding situation for the medical team supervising the patient. A large number of signs and measurements must be supervised and decisions must be made accordingly, in view of predicting the output of the control performed on the patient.

Anesthesia is a critical task currently performed by highly trained medical personnel. However, the load of supervising and controlling a great number of parameters during long time periods is high; therefore human attention limits and fatigue become critical factors. To reduce the load, various systems to control the anesthesia substance delivery and to support the human decision process have been proposed.

The aim of anesthesia controllers is to keep under control the behavior of the cardiovascular system during operation. The controlled parameter is the quantity of delivered anesthesia drug per unit time. The aim is basically to keep the blood pressure in a given range. Such systems should exhibit a high adaptability to a large range of patient sensitivity and responsiveness to anesthesia substance, as well as sudden changes in the response of a given patient, depending on her/his state.

As simulations have shown, simple controllers cannot match this challenging application. Indeed, there is no way to ensure knowledge about a patient such as building an optimal, pretuned controller. Adaptation capabilities are needed because of the wide range of sensitivities to anesthetic agents among different patients and because of the time varying sensitivity of individuals. Every anesthesia controller should be adaptive and the adaptation process should be fast enough to prevent the patient slipping to a critical state, or to irreversible transitions of the patient's state. The learning ability of the controller and its real time response are crucial in this application. To cope with these difficulties, various types of adaptive intelligent controllers were proposed. Linear combiners, neural networks, fuzzy systems and neuro-fuzzy systems, also involving various adaptation algorithms such as genetic algorithms, have been tested on benchmarks consisting of specified response functions of the patients under clinical conditions. The preliminary results reported so far are encouraging. As an example, we focus subsequently on fuzzy controllers.

Fuzzy control is essentially a nonlinear control implemented through rules extracted from a knowledge base. The fuzzy controller is frequently seen as an expert system approach. The rules have fuzzy antecedents and may have either a fuzzy or numerical ("crisp") consequence, according to the type of fuzzy controller (Mamdani type or Sugeno-Takagi type). The numerical output is obtained by aggregating the consequences of the rules in a "defuzzification" block of the controller. The defuzzification method varies from design to design and may be essential in building an appropriate input-output characteristic of the nonlinear controller. The input variables are related to the hemodynamic response of the patient, typically the mean arterial pressure and the cardiac output. The output of the controller represents the drug infusion rate. The rule base reflects the heuristic rules typically considered by anesthesiologists. The fuzzy system control is dependent on the drug used. Several drugs may be used at the same time. Self-organizing fuzzy control systems able to adapt to the patient's

changes in sensitivity to drugs in real time have been reported to attain an impressive tracking ability: the fuzzy controller allows an adaptation to patient's sensitivity of 5:1 during a simulated 4-h operation.

C. Preventive Medicine

Professional diseases are still a major concern in medicine, in spite of the strict regulations most governments and companies are introducing to limit the impairment risk related to professional activities. The increasing average age of the workers as well as the increasing life standard puts new demands for limiting the risk of impairment at higher ages. Expert systems and decision-support systems help to reduce the work and time consumption for health condition screening and for assessing the professional health risks of workers subjected to high noise levels, noxious gases and fumes, or high levels of psychical stress. As an example, typical requirements for such a system used in assessing the risk of hearing loss include the ability to take into account a huge number of factors affecting hearing loss and the ability to determine the risk of specified levels of hearing loss after large time periods. Requirements also include the capability of processing fuzzy information presented by the physician (during examination, e.g., "the tympanum color is reddish"), or by an engineer when describing the nature of the noise ("the noise is impulsive," "the average spectrum of the noise is close to that of the pink noise"). In addition, requirements include the ability to take into account additional risk factors that may increase the potential of hearing loss, such as mycelia in the atmosphere.

Also in preventive medicine, various AI-based models and prediction tools successfully complement and compete with statistical tools today.

D. Rehabilitation

The new rehabilitation technologies, like robot technology and VR-based techniques, face several hurdles before they can be widely deployed in hospitals and then in homes, where the need is highest. A serious potential problem is that if the medical community and medical assistance providers are not prepared—technically and financially—to correctly support the new technology, users will not be able to benefit from it in a significant manner and quantity. There is still a long way to go, from technical solutions to manufacturing, medical assessment, standardization, user education, and financial affordability for many AI-based rehabilitation systems and devices advocated today.

E. Telemedicine

Telemedicine systems supporting prevention, diagnosis, cure, and even telesurgery needs as well as supporting the daily practice of family doctors are rapidly developing with a potentially huge impact on medicine. Basic telemedicine services encompass telemonitoring of patients, teleconsulting, and telerehabilitation (including applications of virtual reality). Basically, a telemedicine system includes the teleconsulting center, the remote patient's units, and the practitioner equipment (workstation, monitors, appropriate communication systems). The teleconsulting center is composed of servers with appropriate data bases, information retrieval systems, knowledge bases to help the practitioner, inference engines, etc. Moreover, experts are available in teleconsulting centers to advise the practitioner, suggest diagnosis, or treatment. The practitioner may stand at the patient bedside or may telemonitor patients using cable or radio connection, a personal computer, and monitoring devices. According to the level of the provided medical care, telemedicine applications may marginally include AI systems or may be highly demanding in such systems. Higher end telesurgery and intensive care applications to come will need large communication bandwidths, powerful computing systems, several AI-based systems, and sophisticated remote equipment. At midway, telemonitoring is composed of the acquisition of the patient's vital signals; depending on the patient condition, the signals may include respiratory flow, blood pressure, and electrocardiographic and electroencephalographic signals. Telemonitoring capabilities include signal processing, determining when alarm thresholds are attained, starting communication with the practitioner, and providing advice and diagnosis. In all these applications, AI is involved.

IV. CONCLUSIONS

Medical science progress and the future of medical practice are strongly dependent on the progress made by applying AI systems to medicine. Salient developments may be produced in the foreseeable future in the field of telemedicine, where progress in the capability of communication and intelligence may alloy to make the field viable, in embedding intelligence in prostheses and portable units, and in relation to the

genetic medicine. Genetic medicine may prove to rely on data mining, knowledge representation and discovery, evolutionary methods, and possibly on nonlinear dynamics.

The potential of artificial intelligence applications in medicine relies on developments in both artificial intelligence and medical sciences. To successfully apply knowledge representation and knowledge processing methods, raw medical knowledge is first needed. The tendency to use raw medical data and to discover new knowledge will be reinforced during the years to come, mainly because of the lack of reliable medical knowledge in several domains. This will increase medical knowledge and possibly open new ways in medicine. Data mining and knowledge discovery are fields where huge progress is expected because of the need of this technology in medicine as well as progress AI.

Since the inception of AI, several of its branches, such as neural networks, semantic networks, knowledge representation, expert systems, and fuzzy logic, have been credited at various times with important roles in the development of medical applications and of medicine as a science. Although out of proportion, these statements carry the truth that AI actually contributed to the development of the medical field. Yet, AI experts had to learn that medicine is continuously evolving and is increasingly demanding on the technical level. While essential progress has indisputably been achieved, it is only small steps along an evolution of matching medical challenges.

SEE ALSO THE FOLLOWING ARTICLES

Artificial Intelligence Programming • Engineering, Artificial Intelligence in • Ethical Issues in Artificial Intelligence • Expert Systems Construction • Health Care, Information Systems and • Hybrid Systems • Industry, Artificial Intelligence in • Intelligent Agents • Neural Networks

BIBLIOGRAPHY

Cios, K. J., and Kacprzyk, J. (Eds.). (2001). *Medical data mining and knowledge discovery*. Heidelberg: Physica–Verlag.

Hudson, D. L., and Cohen, M. E. (1999). *Neural networks and artificial intelligence for biomedical engineering*. New York: IEEE Press.

Lavrac, N., Keravnou, E. T., and Zupan, B. (Eds.). (1997). *Intelligent data analysis in medicine and pharmacology* (Kluwer International Series in Engineering and Computer Science). Boston: Kluwer Academic.

Metin, A. (Ed.). (2000). Nonlinear biomedical signal processing: Fuzzy logic, neural networks, and new algorithms, Vol. 1, in *IEEE Press Series in Biomedical Engineering*. New York: Engineering in Medicine and Biology Society/IEEE Press.

Teodorescu, H. N., and Jain, L. C. (Eds.). (2000). *Intelligent systems and technologies in rehabilitation engineering*. Boca Raton, FL: CRC Press.

Teodorescu, H. N., Kandel, A., and Jain, L. C. (Eds.). (1998). *Fuzzy and neuro-fuzzy systems in medicine*. Boca Raton, FL: CRC Press.

Teodorescu, H. N., Kandel, A., and Jain, L. C. (Eds.). (1999). *Soft-computing in human-related sciences*. Boca Raton, FL: CRC Press.

Mobile and Wireless Networks

Dong-Wan Tcha
Korea Advanced Institute of Science and Technology

GLOSSARY

access point (AP) An integration point which serves network connectivity between a distribution system (DS) and a basic service set (BSS).
base station controller (BSC) A network component in the PLMN with the functions for control of one or more BTS.
base transceiver station (BTS) A network component which serves one cell.
basic service set (BSS) A set of WLAN components which directly communicate with each other.
code divisional multiple access (CDMA) A multiple access technology which allows many users to share a common frequency for transmission; the user signals are distinguished by spreading them with different codes.
distribution system (DS) A backbone network which connects BSSs.
extended service set (ESS) A large logical BSS which consists of BSSs, DS, and APs.
frequency divisional multiple access (FDMA) A multiple access technology which assigns individual channels to individual users; each user is allocated a unique frequency band or channel.
frequency division duplex (FDD) A duplexing method which provides two distinct bands of frequencies for every user.
global mobile satellite personal communication system (GMPCS) Global mobile personal communications by satellites.
home location register (HLR) The location register to which a mobile subscriber is assigned for record purposes such as subscriber information.
industrial, scientific, and medical band (ISM band) 2.4 Ghz band under no regulation.
interworking function (IWF) A functional entity which provides the functionality necessary to allow interworking between a PLMN and the fixed networks.
mobile-service switching center (MSC) A switching device which performs all necessary functions in order to handle the calls to and from the mobile stations between the radio system and the fixed networks.
mobile terminal (MT) The physical equipment used by a PLMN subscriber.
plain land mobile network (PLMN) A mobile network which is established and operated by an administration or Recognized Private Operating Agency for the specific purpose of providing land mobile telecommunication services to the public.
private automatic branch exchange (PABX) A switching device which connects the telephones in a private organization and a number of outgoing lines.
public switched telephone network (PSTN) The voice-oriented public telephone networks, both commercial- and government-owned. It's also referred to as the Plain Old Telephone Service (POTS).
radio transmission technology (RTT) Data transmission technology used in wireless link, such as IS-95, WCDMA.
time divisional multiple access (TDMA) A multiple access technology which divides the radio spectrum into time slots. In each slot only one user is allowed to either transmit or receive.
time division duplex (TDD) A duplexing method which uses time instead of frequency to provide both a forward and reverse link.

user equipment (UE) The physical equipment used by a user to access a network.

visitor location register (VLR) The location register used by an MSC to retrieve information for, e.g., handling of calls to or from a roaming mobile station currently located in its area.

WIRELESS AND MOBILE NETWORKS are rapidly becoming the major type of access network for telecommunication services. This article summarizes the traditional approaches and the current trends in performance modeling for design and operation of such networks. We first present the structure of a general communication network, based on which wireless and mobile networks are classified, into four types. Each type is characterized with focus placed on the air interface, since all other domains are not significantly different from one another. With attention then shifted to designing and managing such networks, the performance measures for that purpose are listed, and the salient features of the mobile air interface are examined. Then epitomized are key classic Operations Research/Management Science (OR/MS) approaches, both probabilistic and deterministic, used for traffic modeling and optimizing wireless and mobile networks. Realizing the significant change in traffic characteristics for Internet traffic, provided at the end of this paper is why mobility management becomes the single most important performance issue for future global multimedia networks.

I. INTRODUCTION

Wireless is becoming the major form of access to the Internet, the growth of which has been phenomenal, doubling in size worldwide every year for the last decade. The number of wireless mobile terminals is predicted to exceed that of wired lines in several years, which has already happened in Korea and Scandinavia. Even before the inauguration of the IMT-2000 service, image traffic is rapidly catching up with voice traffic over the wireless Internet access via mobile terminals.

In this article, we first provide conceptually the domain structure of a general communication network, based on which systems are classified into four types. Each type is characterized, targeting readers who are not well versed in radio communication technologies. With attention then shifted to designing and managing such networks, the performance measures for that purpose are listed, and the salient features of the mobile air interface are examined. Key probabilistic models for traffic modeling and analysis are visited. We also present some deterministic network models in Operations Research/Management Sciences, which can be used, frequently with the probabilistic models above, for designing and operating telecommunication networks in general. Taking note of both salient characteristics of air interface of mobile environment and other complex system features, it is pointed out that the traditional type of performance studies hitherto made based on evaluating performance measures would be no longer effective for future globalized mobile and wireless networks. We elaborate on the issues of mobility management and provide reasons why they are the most important ones for future networks.

II. GENERAL STRUCTURE OF MOBILE NETWORK—CONCEPTUAL DESIGN

A. Domain Split

We start with showing the main constructs, called domains, of the system structure that are common to all wireless and mobile networks in Fig. 1.

The system architecture is basically split into the user equipment (terminals) and the infrastructure, resulting in two domains: the user equipment domain and the infrastructure domain. User equipment has an air interface to the infrastructure. The infrastructure is a shared network resource that provides services to all users within its coverage area. The user equipment domain encompasses a variety of equipment types with different levels of functionality. The equipments, performing radio transmission and containing applications, may be compatible with one or more existing access (fixed or radio) interfaces. A re-

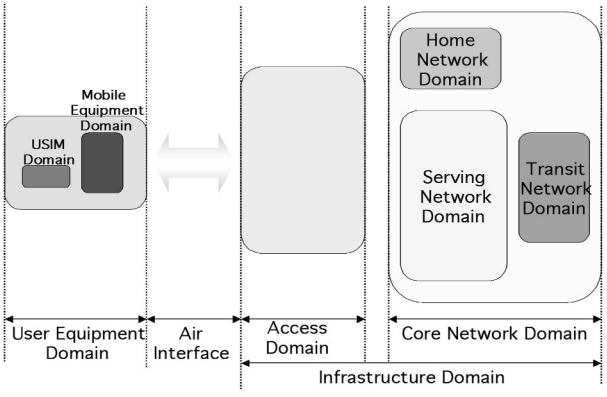


Figure 1 Domain split. (Note: 3GPP technical specification 23.101.)

movable smart card for user identification may also be used in different equipment types.

The infrastructure domain is split into the access network domain and the core network domain, based on the access-related functionality. The access network domain allows the user to access the core domain via a network employing a specific access mechanism. On the other hand, the core network domain can be connected to any type of access network. The core network, in addition to transferring (switching and transmitting) control signal and user-generated information, supports such functionalities as the management of user location information, and the control of network features and services. The core network domain is further divided into the serving domain, the home domain, and the transit domain. The serving domain is the part to which the access network is connected, thus its location changes as the user moves. The home domain represents the core functions that manage user-specific data at a permanent location regardless of the location of the serving domain. The transit domain is the part located on the communication path between the serving domain and the remote party.

Any mobile and wireless communication system could be described as a specific implementation of this domain structure. Figure 2 illustrates a typical example of such physical implementation. Each domain is then characterized by listing the functionality of its key entities.

B. Core Network

The core part, besides its main role of switching and transmitting data, supports functions like managing user information and other network services and features.

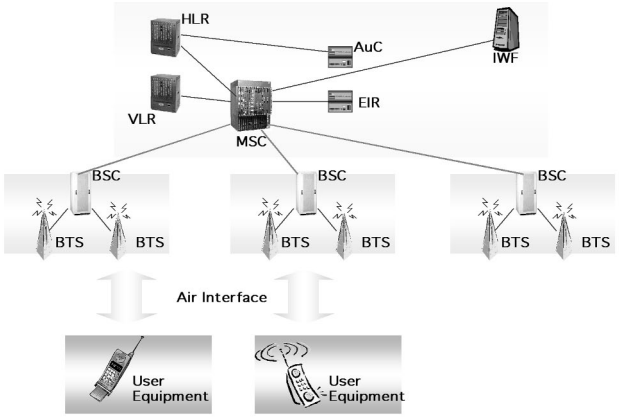


Figure 2 An exemplar mobile communication system.

1. Location Register

There are two types of location register: home location register (HLR) and visitor location register (VLR). An HLR is a database for mobile subscriber management. Stored therein are the subscription information and some location information enabling charging and routing. At least one identity attached to each subscriber and piece of equipment should be kept, respectively, at the authentication center (AuC) and at the equipment identity register (EIR), usually located in HLR.

Mobile terminals (MTs) roaming in an MSC area, which will be defined shortly, are tracked by the VLR in charge of the area. When an MT enters a new location (or MSC) area, it starts a registration procedure by reporting this to the VLR. If this MT is not yet registered, the VLR and the HLR exchange the information required for the proper call management.

The information for the calls set-up or received by the MTs in an MSC area is stored in the database of the VLR. For some supplementary services other than the routine call management, a VLR may have to obtain additional information from the HLR. A VLR may be in charge of one or several MSC areas.

2. Mobile-Service Switching Center (MSC)

An exchange which performs all the switching and signaling functions for MTs located in an area, called an MSC area, is referred to as the mobile-service switching center. It, unlike an exchange in a fixed network, should consider the scarcity and propagation characteristics of radio spectrum, based on which such key procedures as location registration and hand off should be managed. The MSC should be equipped with a functional entity designated as the interworking function (IWF), which allows interworking, via protocol conversion, between a radio network and the existing fixed networks.

C. Air Interface

Air interface, often referred to as radio interface, connects between user equipment (UE) and access domain through radio transmission, which characterizes the wireless network itself. There are many radio transmission technologies, which can be grouped into two lines, ones associated with spectrum issues and the others with transmission technologies.

1. Spectrum Issues

Frequency spectrum is a scarce resource to be efficiently used. For system design three important issues must be considered: regulation on the use of the band of interest, propagation characteristics such as attenuation, reflection, and multipath behavior, and technological impact on equipments like antennas and transceivers.

Another technical item is whether both up- and downlink transmissions are provided via a paired band allowing frequency-division duplexing (FDD) or via a single band only, thus restricted to time-division duplexing (TDD). A general trend on the spectrum issues is that as the frequency increases, more expensive technology is required from the increase in propagation attenuation.

2. Modulation, Channel Coding, and Multiple Access

Modulation is the process of encoding information for transmission. It involves translating a baseband (or simply original) message signal to a bandpass signal at much higher frequencies. The translation of the message signal is performed by varying the amplitude, phase, or frequency of a high-frequency carrier in accordance with its amplitude. Demodulation is the process of extracting the baseband message from the received carrier signal.

Channel coding is the process of adding redundant bits to the transmitting message to improve the link performance. The coded message, containing a larger number of bits than originally contained in the message, is modulated for transmission in the wireless channel. Some errors introduced by the channel can also be detected or corrected at the receiver side. Channel codes generated for error detection are called error detection codes, while those for both detection and correction are called error correction codes.

Multiple access schemes allow the scarce resource of radio spectrum to be shared by many mobile users. The sharing of spectrum must be done without severe degradation in system performance for high-quality communications. Frequency division multiple access (FDMA), time division multiple access (TDMA), and code division multiple access (CDMA) are the three major access techniques in a wireless system. These techniques can be grouped as narrowband and wideband systems, depending upon the size of bandwidth to be shared.

III. WIRELESS AND MOBILE COMMUNICATION SYSTEMS

A. Overall Description

Wireless and mobile networks, since their infrastructure domains, particularly the core domains, are not significantly different from one another, are generally classified according to the air interface into the following four types:

1. *Cellular network:* A cellular network, in comparison with its predecessor with a single large zone, consists of many cells with small coverage. Each frequency channel is used repeatedly by multiple cells, sufficiently far apart to be interference tolerant from each other. This frequency reuse is a key enabler to support a huge number of subscribers. The hand-off mechanism is another important feature, by which the frequency assigned for a call is seamlessly changed as the MT crosses cell boundaries. This requires the MT to frequently change frequencies as needed.
2. *Cordless telephony:* Cordless telephony is basically not much different from the cellular system, but more characterized by low mobility (small range and low user speed) and small power (i.e., limited coverage). Starting from a simple application at private residence, it has evolved to a telepoint access system supporting the unidirectional calling capability only. Customer needs transformed the system to have two-way calling capability, and a wide service area was partitioned into a number of small cells, rendering the system to have similar characteristics with the cellular network above.
3. *Wireless LANs:* Wireless LANs (WLANs) are preferred to their wired counterparts for situations in which wiring is difficult or impractical, or some degree of mobility is needed. Many of WLAN standards use the ISM band, and some architectures even support ad hoc implementation, whereby terminals communicate directly with each other (peer to peer) without the mediation of a fixed base station. A group of WLANs under the common architecture can form a large logical network with a wide service area, in which case mobility management becomes a key issue.
4. *Communications satellites:* Communications satellites are everywhere these days, utilized for fixed wireless access, broadcast, and mobile

services. The classic geostationary earth orbit (GEO) systems mostly offer broadcast services while those on low/mid-earth orbit (LEO/MEO) are for the so-called global mobile satellite personal communications (GMPCS).

The categorization into four types, having similar core network functionalities, arises from their differences in the air interface and some access network functionalities.

B. Cellular Network

1. Access Network Part

A base station system consists of a collection of equipment (transceivers, controllers, etc.), for communicating with MTs in a certain area. A BSS has one base station controller (BSC), and one or more base transceiver stations (BTS) controlled by the BSC. A base transceiver station (BTS) is a network component that serves one cell. A base station system expands the so-called base station, in charge of a single cell in the early implementation stage, into a two-level hierarchy covering multiple small cells.

2. History and Deployment

The cellular configuration started replacing the single large cell topology with the inauguration of the improved mobile telephone service (IMTS) in the early 1960s. But most of the important concepts of the cellular system were established in the 1980s when the first commercial mobile telephony was introduced. This first generation cellular telephone was based on the analog technology with FM modulation. A typical example is the advanced mobile phone services (AMPS) still operative in North America. The second generation (2G) wireless system employs digital modulation, using spectrally efficient radio transmission schemes, the TDMA or the CDMA, in comparison to the analog FDMA scheme previously employed. In addition to this substantial increase in system capacity, another inherent benefit of the digital 2G system is incorporating a wide variety of integrated speech and data services. Representative 2G wireless systems are the global systems for mobile communication (GSM), TDMA IS-54, IS-136 and CDMA IS-95 standards in the United States, personal digital cellular (PDC) in Japan.

The third generation (3G) systems, aiming at providing universal access and global roaming, are expected to support multidimensional high-speed services. They will indeed be an important milestone for ubiquitous personal communications. Introduction of wideband packet data services up to 2 Mbps is probably the main attribute of 3G systems.

a. The Second Generation Cellular Network

Table I shows the air interface characteristics of four key 2G systems.

b. The Third Generation Cellular Network

The 3G systems are intended to offer a wide range of services in many different radio environments, with the quality of wireline networks. Key elements characterizing 3G systems are in the radio transmission technology (RTT). Many RTT proposals converge to two reliable candidates: one is W-CDMA of Europe and another is cdma2000 of North America. Table II shows the air interface characterizations of these two 3G systems.

C. Cordless Telephony

Starting with a fixed location like a cordless handset at home, this line of systems was originally intended to support an island-like isolated zone with high

Table I Air Interface Characteristics of 2G Systems

	GSM	IS-54/136	PDC	IS-95
Region	Europe	North America	Japan	North America
Frequency band (kHz)	900/1800/1900	800/1900	700/1500	800/1900
Multiple access	F/TDMA	F/TDMA	F/TDMA	F/CDMA
Carrier spacing (kHz)	200	30	25	1250
Modulation	GMSK	OQPSK	OQPSK	BPSK/QPSK
Channel coding	Convolution code Rate 1/2	Convolution code Rate 1/2	Convolution code Rate 1/2	Convolution code Rate 1/2 or 1/3

Table II Air Interface Characteristics of 3G Systems

	WCDMA	cdma2000
Multiple access	CDMA	CDMA
Carrier spacing (MHz)	5/10/20	1.25 : 1x, 5 : 3x
Wideband implementation	Direct spread	Multicarrier
Inter BS synchronization	Not required	Required
Frame length	10 ms	5/20 ms

subscriber density with little concern for the subscriber mobility. As the diversity as well as the volume of subscribers increase, the service area has become large to consist of multiple small zones, which in turn requires the subscriber mobility to be supported. This is in good contrast to the cellular network, the original form of which is a single large zone with maximum support of the mobility of subscribers therein. The whole service area then becomes cellular to accommodate a rapid increase in the subscriber population, still with high concern on the terminal mobility. Despite the functional differences, particularly on the mobility, of two systems at their inception stages, both network configurations are of cellular type with few significant structural differences in core and access network domains. However, there are some notable differences in the air interface between the two systems, which are listed in Table III.

D. Wireless LAN (WLAN)

The fundamental unit of WLAN architecture is the BSS, defined as a group of stations that are under the direct control of a single coordination entity. The area covered by a BSS is called a basic service area (BSA), which is analogous to a cell in a cellular network. All stations in a BSS are supposed to be able to directly communicate with all other stations in a BSS.

Table III Comparison of Cordless and Cellular Systems

Characteristics	Cordless	Cellular
Cell size	Small (50–500m)	Large (0.5–30km)
Mobility speed	Low (<6km/h)	High (<250km/h)
Coverage	Zonal	Wide area
Handset complexity	Low	Moderate
Duplexing	TDD	FDD
Speech Coding	32 kbps ADPCM	8–13 kbps vocoder

An ad hoc network is a grouping of stations into a single BSS without any aid from outside, which is illustrated in Fig. 3. Access points (APs) act as the integration points necessary for network connectivity between multiple BSSs, thus forming an extended service set (ESS). The ESS can thus be viewed as a large logical BSS, enabled by a common distribution system (DS) interconnecting APs. The DS can be thought of as a backbone network, the implementation of which can be done independently via either a wired LAN or other wireless LAN.

An ESS can also provide gateway access to a wired network via a device, a logical entity, known as a portal. Figure 4 illustrates a simple ESS developed with two BSSs, a DS, and a portal access to a wired LAN.

1. Access Network Part

An AP has two interfaces: one is the wireless connection with user equipments, and the other is the wireline connection with the DS.

2. Air Interface

Two of the most representative air interface standards for WLAN and IEEE 802.11 and Bluetooth. The IEEE 802.11 standard defines three different physical layer implementations: frequency hopping spread spectrum (FHSS), directed sequence spread spectrum (DSSS), and infrared (IR).

The FHSS utilizes the ISM band. Hopping sequences (channels) different from each other enable multiple BSS to coexist in the same geographical area. They are generally grouped into a couple of hopping sets, in order to avoid prolonged collisions between hopping sequences. The access rate is normally 1 Mbps using two-level gaussian frequency shift keying

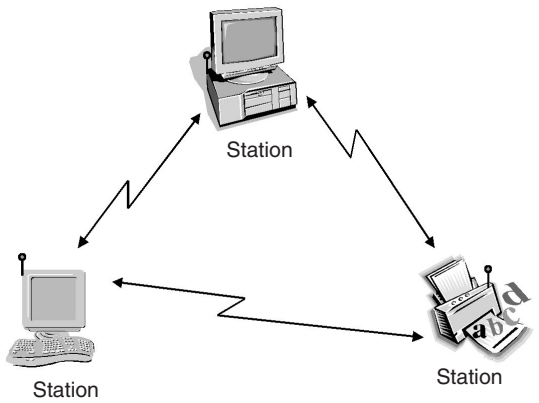


Figure 3 Sketch of an ad hoc network (case of a single BSS).

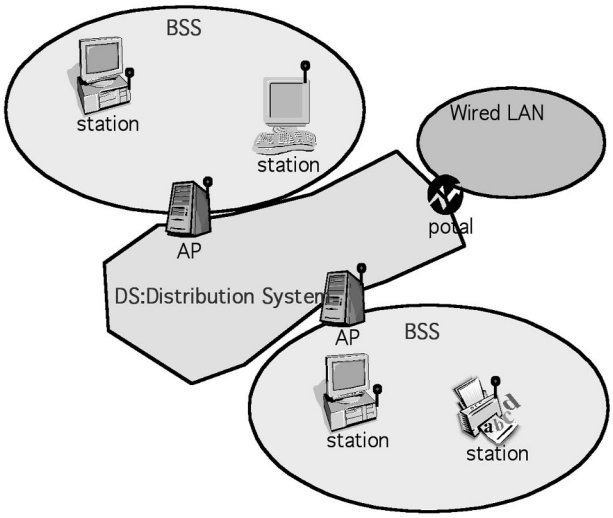


Figure 4 Sketch of an infrastructure network.

(GFSK), while the enhanced access rate of 2 Mbps uses the four-level one.

The DSSS also uses the 11-Mhz bandwidth in the ISM band, on which 11 channels are spread. For each channel, the 1-Mbps basic rate is encoded using differential binary phase shift keying (DBPSK), and a 2-Mbps enhanced rate uses differential quadratic phase shift keying (DQPSK).

The IR specification, using a wavelength range from 850–950 nm, is designed for indoor use only, and allows both line of sight and reflected transmissions. Encoding of the basic access rate of 1 Mbps is obtained by 16-pulse position modulation (PPM), where 4 data bits are mapped to 16 coded bits for transmission. The enhanced access rate (2 Mbps) is performed using 4-PPM modulation, where 2 data bits are mapped to 4 coded bits.

The Bluetooth wireless technology allows users to make instant connections between various communication devices. The Bluetooth radio is built into a small microchip, operates in the ISM band, and uses a fast acknowledgement and frequency hopping scheme to make the wireless link robust. Specified in the present standard are the gross data rate of 1 Mbps, time-divisional duplexing, and packet switching. Each packet, nominally covering a single slot, is transmitted in a different hop frequency. Two power levels are defined in the specification to cover from a small area like a room to a medium range such as within a home. Software control including identity coding built into each microchip makes the communication devices under the full control of their owners. Supporting both point-to-point and point-to-multipoint connections, the current specification allows up to seven "slave" devices set to communicate with a "master" radio in one device. Multiple "piconets" thus established can be linked together in ad hoc "scatternets," rendering a continually flexible network configuration.

E. Satellite Communication Network

A mobile satellite system (MSS) consists of satellites, system control center, gateways, and terminals as shown in Fig. 5. Satellites directly open channels via

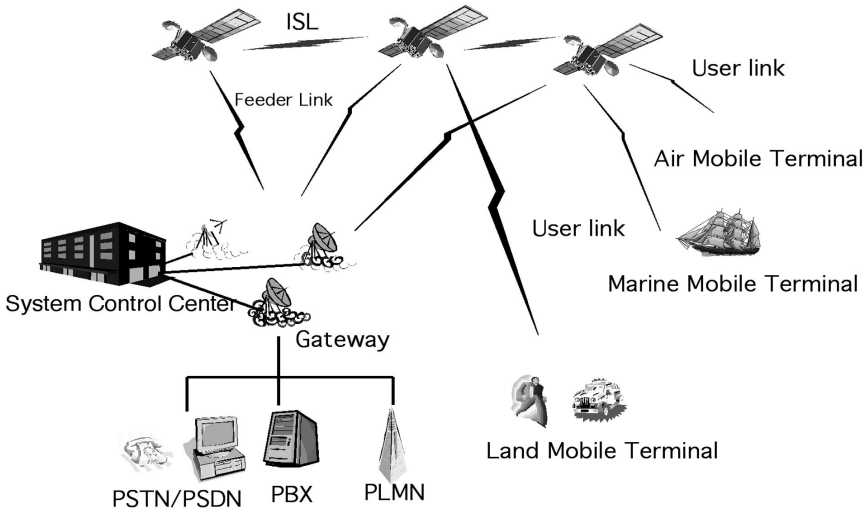


Figure 5 Sketch of a general mobile satellite network.

user links to terminals, and connect via feeder links to various land networks like public switched telephone network (PSTN) or plain land mobile network (PLMN) through gateways. They may also create intersatellite links (ISLs) for intersatellite direct communications. The overall satellite network is monitored and managed by a system control center. There are a wide variety of terminal types: handset, vehicle attached terminal, pager, and mobile computing equipment, etc.

1. Access Network Part

In MSS, the satellite is the only network access component, performing almost the same function of base station in a cellular network. According to the level of functionality, satellites are classified into two types, intelligent and dumb. An intelligent satellite can open intersatellite links and perform switching functions. A dumb satellite performs only the simple relaying function with other functions like switching relegated to the system control center.

2. Air Interface

Several satellite systems on LEO/MEO operating at the L/S-frequency band have been or are being deployed to provide the so-called GMPCS. The classic GEO systems are not addressed here owing to the narrow scope of its services like international broadcast relaying and international telephoning. Focusing on multimedia personal communications, only the air interface of the GMPCS system is examined, the summary of which is given in Table IV.

IV. DESIGN AND ANALYSIS

We first describe the characteristics of mobile and wireless networks, and their idiosyncrasies with respect to traditional, hardwired networks. We then list both quantitative and qualitative performance measures that can be used for the design and operation of any type of communication network. Traditional traffic modeling and analysis using probabilistic, particularly queueing models, is reviewed with concern about evaluating key performance measures. Representative deterministic network models in Operations Research/Management Science are briefly visited to glimpse how they, often together with the performance models, can be used in design and operation of telecommunication networks.

We move on to examine how the characteristics of information traffic have changed in accordance with the evolution of network services and environments. Evidence has been provided that the traditional approach of evaluating performance measures is no longer appropriate when mobile and wireless networks are integrated and globalized. The mobility management is persuaded to be the most comprehensive performance criterion, related to most of the aforementioned key measures, for a future globalized network. Finally, the issues of mobility management are categorized and described.

A. Characteristics of Mobile Air Interface

Wireless mobile networks have ever-changing topologies, mostly composed of bandwidth-constrained wireless links. Nodes are equipped with wireless transmit-

Table IV Air Interface Characteristics of Satellite Systems

			Iridium	ICO	Globalstar
Services (common: voice, data, fax)			Paging, location information	Paging	Paging, location information
Data rate (kbps)			2.4	2.4	7.2
Keying method			QPSK	QPSK	QPSK
Orbit/altitude (km)			LEO/778	MEO/10355	LEO/1390
Number of satellites (primary + backup)			66 + 6	12 + 2	48 + 8
Number of gateway			15 ~ 10	12	100 ~ 210
Multiple access method			TDMA/TDD	TDMA/FDD	CDMA/FDD
Frequency	User link	Up(kHz)	1621.35 ~ 1626.35	1980~ 2100	1610 ~ 1621.35
		Down(kHz)		2170 ~ 2200	2483.5 ~ 2500
	Feeder link	Up(kHz)	1940 ~ 1960	5100 ~ 5200	5091 ~ 5250
		Down(kHz)	2910 ~ 2930	6875 ~ 7055	6875 ~ 7055

ters and receivers, sometimes with a platform like a router. We list several salient characteristics of the wired ones

- **Bandwidth-constrained, sometimes variable capacity links.** Wireless links have lower capacity than their hardwired counterparts. In addition, due to the effects of multiple access, fading, noise, and interferences, the actual throughput is often much lower than the specified transmission rate. Aggregate application demands will likely approach or exceed network capacity frequently, rendering the resulting congestion and the variable capacity the norm than the exception.
- **Limited physical security.** They are more prone to physical security attack than wired ones, due to the effects of eavesdropping, spoofing, and denial-of-service attack.
- **Energy-constrained operation.** User nodes rely on batteries or other exhaustible means for their energy requirement. Energy conservation is thus one of the most important system design criteria.
- **Changing topology or coverage.** Nodes representing MTs are free to move arbitrarily, resulting in geographically varying traffic density. Thus cell or basic service areas, and thus the overall network topology, may have to change randomly and rapidly to accommodate this traffic variation.

B. Performance Measures for an Individual Network

The performance measures, both qualitative and quantitative, are used to evaluate the suitability/performance of the structure/protocol of a network. We first briefly list quantitative measures and then qualitative ones for a general network. It is relatively obvious to find which measures are more affected by the abovementioned characteristics of wireless mobile interface. We first list the quantitative measures.

1. *End-to-end data throughput and delay:* The most popular measure indicating the network's overall effectiveness from the external perspective
2. *Blocking/outage probability:* Call blocking occurs when the offered traffic load is close to nodal or link capacities; call outage, almost synonymous with blocking, is for CDMA wireless link
3. *Session setup time:* A particular form of external end-to-end delay measurement of call setup, of particular concern to a wireless LAN of ad hoc type
4. *Percentage out-of-order delivery:* Of concern to a network adopting connectionless transmission, since this causes an extra alignment procedure
5. *Proportion of data (control) bits transmitted to data bits delivered:* A typical metric for network efficiency
6. *Bit error rate (BER):* Probably the most important measure, particularly for mobile and wireless environment, which is closely related to most of the above measures

Besides the important criteria like security and power consumption, there are the following qualitative ones: centralized or distributed operation, occurrence of loop-free routes, and traffic pattern on a demand basis or control basis. Packets spinning around in a network, particularly in a wireless LAN of ad hoc type, should be avoided, or at least be minimized. Worth noting is the general performance trend with which the demand-based approach can utilize network energy and bandwidth resources more efficiently, at the cost of increased route discovery delay, compared to the control based approach.

Owing to the adverse and uncertain environment of mobile air interface, each of the listed performance measures is more sensitive to a mobile network than to fixed networks. Evaluating network performances, particularly for mobile ones, is thus recommended with the more sensitive measures. When selecting a performance measure, the handling capability of the associated evaluation has also to be counted, not to mention its fundamental importance. Discretion is further called upon to which class the network traffic under consideration mainly belongs from among four different cases: conversational (e.g., voice), streaming (video), interactive (web browsing), and background (e-mail download).

C. Traffic Modeling and Network Models

In the face of rapidly advancing and complicating information technologies and services, professionals and specialists are summoned to the design and management of the associated infra-networks. Required for such design and management decisions are prediction and evaluation of some key network performance measures for various potential network alternatives. Using or being aided by these results, one can then apply the existing optimization tools for systematically evaluating alternative solutions for network design and operation.

1. Probabilistic Models

Many performance models addressing such measures as delay, throughput, blocking, etc., are based on queuing theory. In the generic queuing models, customers randomly arrive at a facility with service requirements that may be random in duration. The theory attempts to find probabilistic descriptions of such quantities as the size of waiting lines, the delay experienced by an arrival, and the availability of a service facility. In the voice telephone network, demands for service take the form of subscribers initiating a call. Erlang, a Danish mathematician, found that given a sufficiently large population, the random rate of such calls can be described by a Poisson process. In this application the duration of the call was found to have an exponential distribution. An important result obtained by Erlang, which was used until lately in engineering a voice network, is the probability of a busy signal, or call blocking, as a function of the traffic level and the number of lines that have been provided.

Though such queuing models were developed for voice traffic, they have widely been used for data communications as well. The generation of data packet at a station corresponds to the initiation of voice calls to the network, while the time required to transmit the packet over a link to the required time of service is a function of both the packet length and the transmission rate. The associated queuing models for both voice and data communications, with a few exceptions, are based on the assumption that customer arrivals follow a Poisson process. But the recent explosive growth in the Internet users makes the commonly assumed models for network traffic, e.g., the Poisson process, be challenged. Were traffic to follow a Poisson process, it would have a characteristic that burst lengths would tend to be smoothed by being averaged over a long enough time scale. Construction and operation of many traditional voice and data networks has actually been founded on this phenomenon.

Measurements of real Internet traffic, however, indicate that significant traffic variance (burstiness) is present on a wide range of time scales. That is, in many real multimedia network traffic, we observe the property that is associated with one type of fractal—an object whose appearance is unchanged regardless of the scale at which it is viewed. Traffic having this property—bursty on many or all time scales—can be described statistically using the notion of self-similarity. Recently reported is evidence that web traffic, in addition to Ethernet and other LAN traffic, exhibits behavior that is consistent with self-similar traffic models. This implies that the smoothing effect of Poisson models commonly adopted hitherto can no longer be guaranteed for today's multimedia network traffic. From now on, acute discretion is thus required on various performance modeling by only selectively employing Poisson assumptions.

The conventional probabilistic models, because they deal mostly with voice and simple data traffic, could yield some analytic results, though approximate ones in many cases. These results have frequently been incorporated to render a comprehensive optimization model for network design and management, for the solution of which various Operations Research/Management Science tools as shown in Table V are utilized. On the contrary, this development can no longer be expected for modern-day mobile and multimedia networks, due

Table V Basic OR Models

Basic OR model		
Shortest path	Description	When the length of each link is specified in a given network, we wish to run as fast as possible between two specified nodes, a start node and a terminal node
	Applications	(Shortest path) routing
Maximum flow	Description	In a capacitated network, we wish to send as much flow as possible between two specified nodes, a source node and a sink node
	Applications	Available # of circuits between two nodes
Minimum spanning tree	Description	We wish to find a spanning tree (a connected acyclic subgraph spanning all the nodes) that incurs the smallest total cost
	Applications	Least-cost construction of a network
Multicomodity flow	Description	When there are multiple flow classes (commodities) sharing the network resource together, we wish to calculate a least-cost flow solution satisfying all commodity demands
	Applications	Design and operation of a general communication network

to the scarcity of analytic performance results which has been reasoned above. Simulation seems at the moment the only viable technical means for evaluating network performances, which in turn explains why we cannot find any notable comprehensive optimization models for design and operation of today's mobile and multimedia networks.

2. Optimization Models

The key to analyzing and designing networking strategies is the efficient use of sophisticated network design tools that employ both optimization and simulation methods. These tools are essential to evaluate alternative networking solutions while attempting to optimize communications resources. They, once appropriately applied with the aforementioned performance results, enable the establishment of a cost-effective infrastructure while satisfying various Quality of Service (QoS) requirements.

For a glimpse of such an example, consider a BTS location problem in which the given service area is to be covered with the minimum number of BTS, a solution of which is illustrated in Fig. 6.

The problem can be modeled as a set covering problem, one of the most basic OR problems, for which fast heuristics are reported in the literature. To make the model realistic, its variant can be formulated by adding such QoS requirement constraints as BER on a mobile link.

Designing networks is a complex task that can be addressed only by segmenting the big problem into smaller, more manageable subproblems. These small manageable problems quite frequently take the form of such well-known mathematical models as the shortest path problem, the minimum spanning tree problem, the maximal flow problem, the minimum cost multicommodity problem, etc. Several representative mathematical models are listed with their description and usage, though the description is given with attention on network design. All these basic OR models and solution methods can also be effectively used for operating and managing networks as well.

The reader's attention is summoned to the fact that the use of these OR/MS models in network design and operation has significantly been reduced for today's mobile networks with two complicating factors hard to overcome: the existence of multiple traffic classes and the difficulty in traffic modeling at air interface.

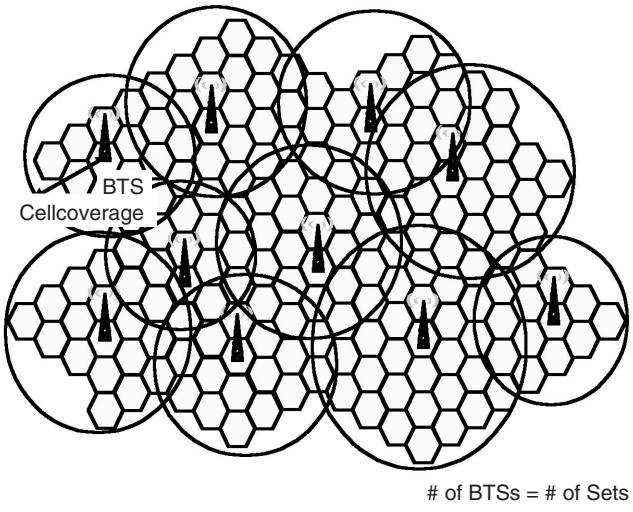


Figure 6 A solution of BTS location.

3. Change in Traffic Characteristics

Since voice traffic has been, until recently, the major constituent of network traffic, all traffic parameters are traditionally obtained from analysis of voice traffic, characterized by Poisson arrival and exponential duration time. As the main service migrates from voice to packet data services like web browsing, file transfer, e-mail, etc., research has been directed to develop multimedia capabilities on both wired and wireless computing environments. These new major data traffic have different characteristics than that of the traditional voice traffic. Figure 7 shows the general traffic pattern of this data traffic.

The main characteristics of this new traffic pattern are listed below.

Data burstiness: Session arrivals can be viewed the same as the Poisson arrival pattern of voice service. But the constituent packets of each session arrive in a bursty manner and the arrival points appear to form visual clusters. That is, interarrival times tend to run such that a series of several relatively short interarrival times are followed by a few relatively long ones.

Heavy-tailed service duration: Data session durations tend to have heavy-tailed distributions such as the Pareto and Weibull distribution, which has infinite variance.

Strict QoS requirement: Generally accepted BER of data is 10^{-6} which is stricter than the 10^{-3} of voice. The signal to interference ratio, generally represented by E_b/I_o, for the CDMA air interface, for instance, must be 12 dB, which critically limits the system capacity.

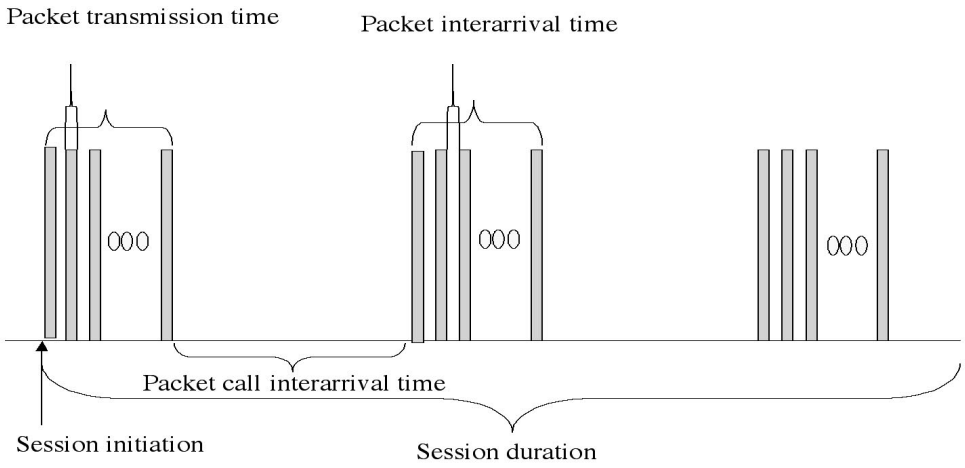


Figure 7 General packet traffic pattern.

Table VI shows typical parameter values of two classes of packet transmission: high multimedia and medium multimedia.

4. Performance Studies on Future Nerworks

Under the environment of changing traffic characteristics, it is not easy to maintain a satisfactory level in each key QoS measure through the life of call (or session) connection. Further complicating factors are adverse effects brought forth by the aforementioned salient characteristics of mobile air interface. In most classic performance studies, a single or a few measures from among the listed performance measures above are analyzed to evaluate the effectiveness of alternative network systems. But this type of performance analysis becomes harder to conduct or even loses its significance as the wireless services become prevalent, i.e., as a huge wired and wireless communications infrastructure is about to offer heterogeneous services to users that may roam across various geographical and network boundaries. Quantifying such a performance measure may be meaningless in a global network encompassing a variety of different wireline and wireless networks, since it cannot be a basis for enhancing the overall performance of such a worldwide network.

Table VI Traffic Characteristics of General Packet Service

	Downlink HMM and MMM	Uplink HMM and MMM
Number of packet calls per session (NPCPS)	5	5
Number of packets per packet-call (NPPPC)	25	25
Number of bytes per packet (NBPP)	480	90
Packet-call interarrival time (PCIT) (s)	120.00	120.00
Packet interarrival time (PIT) (s)	0.01	0.01
Transmission rate (kb/s)	64	64
Total session time (s)	4.88E + 02	4.88E + 02

D. Mobility Management

As regional and national network coverage becomes worldwide coverage, research on system performances becomes directed toward the comprehensive issue of mobility management. More direct means of utilizing the radio resource, the major limiting factor of the system capacity and performance, is to upgrade the air interface by incorporating the advancing radio transmission technologies. Bypassing these important engineering issues on air interface, our attention will be centered on the key issue of systems design and management—mobility management.

A wireless communication infrastructure, being either a single network or a group of heterogeneous networks, should be able to locate roaming MTs for call delivery and to maintain connections with MTs that change their point of attachment. The first process is called location management, and the second hand-off management. At this point, it would be worthwhile to note how much each and every perfor-

mance measure listed above depends upon these two processes. For instance, for a key measure of setup time, consider its relation with locating a roaming terminal. Also consider how critically the measure of delay and throughput is affected by the processing efficiency of hand-offs between cell or network boundaries. This dependency would become more severe as a communication infrastructure gets globalized consisting of heterogeneous wired or wireless networks. This explains why the traditional type of performance studies focusing on evaluating some measures for an individual network gives way to the comprehensive issue of location management as networks gets integrated and wireless.

1. Location Management

Location management is a two-stage process: the first stage is to discover the current attachment point of the mobile user for call delivery, as shown in Fig. 8, and the second one is call delivery. The associated protocol deals with querying and storing information in location databases and sending paging signals to locate the user within the network. Key research areas are the design of database architecture to reduce query traffic, streamlining location update signaling, and terminal paging schemes. Figure 8 associates these research issues including the one on security with their respective location management operation. Since location management deals with database and signaling issues, many of the issues are not protocol dependent and can be applied to any of the mobile networks.

2. Handoff Management

Hand-off (or handover) management for maintaining connection with mobile terminals is a three-stage process. The first stage involves initiation of identifying the need for hand-off, and at the second stage new resources for the hand-off connection must be found and any additional routing operations must be performed. Under the network-controlled hand-off (NCHO), or mobile-assisted hand off (MAHO), the network takes control of the procedure while the mobile terminal does the job for the mobile-controlled hand-off (MCHO). The final stage is data-flow control, where the delivery of the data from the old connection path to the new connection path is maintained according to agreed-upon service guarantees. The handoff management operations are presented in Fig. 9.

Hand offs are of two kinds depending upon the condition under which they take place: intracell hand off and intercell hand off. Intracell hand off occurs when the user's call has to be transferred to a new radio channel of appropriate strength at the same BTS due to the deterioration of the present channel's signal strength. Intercell handoff occurs when the user moves into an adjacent cell and all of the terminal's connections must be transferred to a new BTS.

Handoffs can be distinguished between the soft and hard based on the signaling diversity kept throughout the operation. During the soft hand off, a terminal is connected to multiple BTS simultaneously and uses some form of signaling diversity to combine the multiple signals. In the hard hand off, on the other hand, a terminal stays connected to only

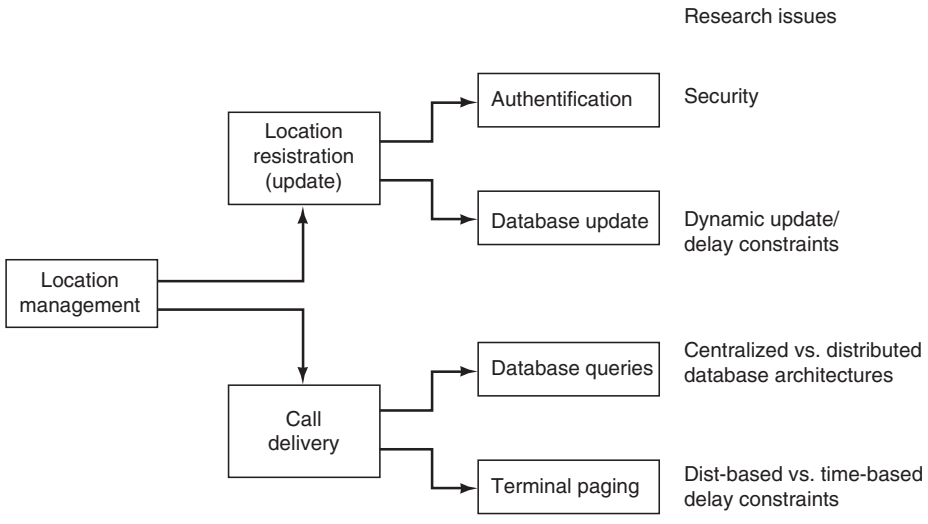


Figure 8 Location management operations. (Note: Akyildiz, "Mobility management in next generation wireless system.")

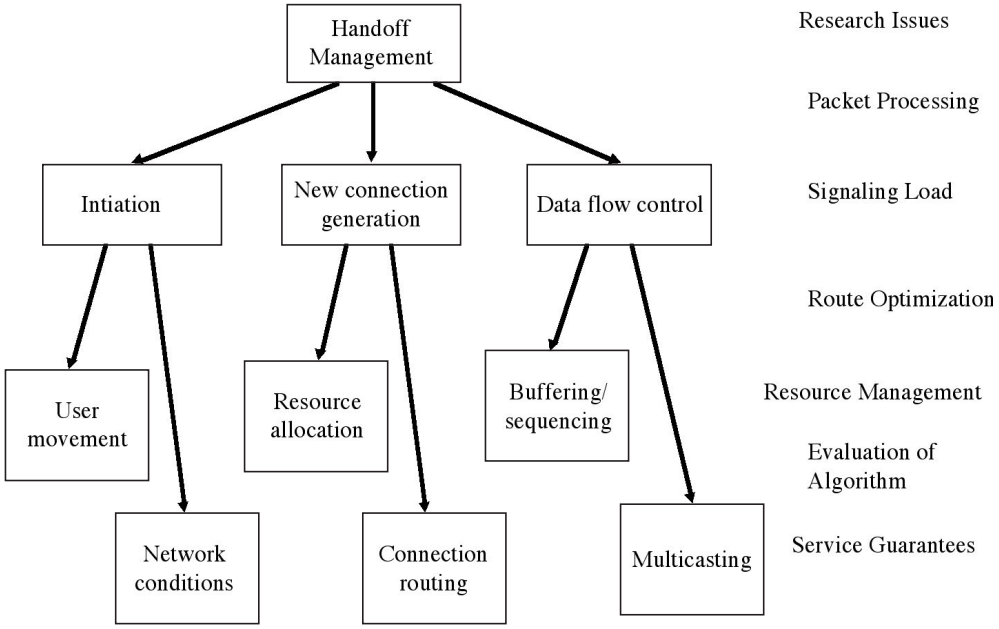


Figure 9 Handoff management operations. (Note: Akyildiz, "Mobility management in next generation wireless system.")

one BTS at a time, clearing the connection with the former BTS immediately before or after establishing a connection with the target BTS.

Research issues on systems management for hand off are minimizing the signaling load on the network, optimizing the route for each connection, and efficient bandwidth reassignment. These issues including the ones on the air interface are listed in Fig. 9.

3. Global Roaming and Other Issues in Future Networks

Future networks will begin to implement personal mobility and service provider portability in addition to terminal mobility. Personal mobility is the ability of the user to access their personal services independent of their attachment point or terminal. Service provider portability allows the user and/or the terminal to move beyond regional mobile networks. The user will be able to receive his personalized end-to-end services regardless of the current network, within the limits of the visited network's service offerings. This freedom then requires the coordination of a wide range of service providers, compatibility of backbone networks, and network operator's agreements.

This level of global mobile freedom also requires the evolution in network architecture and structure to support radio environments that range from high-capacity picocells, to urban terrestrial micro- and macrocells, to large satellite cells.

This evolution then brings forth renewed interest in channel or bandwidth allocation on which the level of cells are to be assigned for terminals of varying mobility and functionality in addition to the traditional channel allocation under the homogeneous cell environment. Various agreements among many service providers for global roaming require further development of location management operations like accessing and updating databases and paging.

V. CONCLUSIONS

This paper reviews modeling approaches for design and operation of various mobile and wireless communication networks. Basically for the reader who is not well versed in radio communication technologies, mobile and wireless networks are classified into four types based on the common domain framework, and each type is reviewed with an accent placed on the air interface. With appropriate performance measures for such networks mentioned, we examined some key quantitative models, probabilistic and deterministic, along with their usage in the design and operation issues. We then observed the significant change in traffic pattern for the emerging multimedia networks which are both globalized and integrated. After finding that the classic performance modeling approaches may not be effective any further, the paper ends with accentuating why the issue of the mobility manage-

ment should become the single most important performance criterion for design and operation of future wireless and mobile networks.

ACKNOWLEDGMENT

The author is grateful to Jae-Hoon Kim, his Ph.D student, for sharing his insight and providing access, data, and feedback, without which this work might not have been completed in time.

SEE ALSO THE FOLLOWING ARTICLES

Electronic Mail • Future of Information Systems • Global Information Systems • Internet, Overview • Local Area Networks • People, Information Systems Impact on • Telecommunications Industry • Telecommuting • Voice Communications

BIBLIOGRAPHY

Akyildiz, I. F., *et al.* (August 1999). Mobility management in next generation wireless system. *Proceedings of the IEEE,* Vol. 87, No. 8.

Alwan, A. *et al.* (April 1996). Adaptive mobile multimedia networks. *IEEE Personal Communications.*

Baugh, C. R. (July 1998). Personal access communications system: Fixed wireless local loop and mobile configurations and services. *Proceedings of the IEEE,* Vol. 86, No. 7.

Berruto, E., *et al.* (February 1998). Research activity on UMTS radio interface, network architectures, and planning. *IEEE Communication Magazine.*

Cook, C. I. (1994). Development of air interface standards for PCS. *IEEE Personal Communications.*

Dravida, S., *et al.* (May 1998). Narrowband and broadband infrastructure design for wireless networks. *IEEE Communications Magazine.*

Gardiner, J. G. (April 1990). Second generation cordless (CT-2) telephony in the UK: telepoint services and the common air-interface. *Electronics & Communications Engineering Journal.*

Geiger, R. L., *et al.* (1997). Wireless network extension using mobile IP. *IEEE Micro.*

Honcharenko, W., *et al.* (January 1997). Broadband wireless access. *IEEE Communications Magazine.*

Jabbai, B., *et al.* (January 1995). Network issues for wireless communications. *IEEE Communications Magazine.*

Korinthios, J. A., *et al.* (April 1996). Numbering and addressing aspect of the UMTS's integration into the fixed network infrastructure. *IEEE Personal Communications.*

Mitrou, N. M., *et al.* (February 1993). Voice and data integration in the air-interface of a microcellular mobile communication system. *IEEE Transactions on VT,* Vol. 42, No. 1.

Noerpel, A. R., *et al.* (June 1996). PACS: Personal access communications system—A tutorial. *IEEE Personal Communications.*

Noerpel, A. R., *et al.* (October 1996). PACS: Personal access communications system: An alternative technology for PCS. *IEEE Communications Magazine.*

Ojanpera, T., *et al.* (September 1998). An overview of air interface multiple access for IMT-2000/UMTS. *IEEE Communications Magazine.*

Orlik, P. V., *et al.* (July 1998). Traffic performance and mobility modeling of cellular communications with mixed PlatforMT and highly variable mobilities. *Proceedings of the IEEE,* Vol. 86, No. 7.

Phalavan, K., *et al.* (September 1994). Wireless data communications. *Proceedings of IEEE,* Vol. 82, No. 9.

Shafi, M., *et al.* (October 1997). Wireless communications in the twenty-first century: A perspective. *Proceeding of IEEE,* Vol. 85, No. 10.

Steels, R., *et al.* (January 1995). System aspects of cellular radio. *IEEE Communications Magazine.*

3GPP technical specification 23 series.

3GPP technical specification 25 series.

Zanders, J., *et al.* (August 1997). Radio resource management in future wireless networks: Requirements and limitations. *IEEE Communications Magazine.*

Model Building Process

Robert Blanning
Vanderbilt University

GLOSSARY

analytical model A model of a real-world process, as opposed to a model of the reasoning processes of an expert in a real-world problem domain.
brittleness The extent to which a model, especially an intelligent model, substantially loses accuracy when its inputs fall outside of certain boundaries.
insightfulness The degree to which a model reveals useful qualitative insights about the real world, such as the insensitivity of certain outputs to certain inputs or oscillations in one or more outputs, regardless of any predictive inaccuracies.
intelligent model A model of the reasoning processes of a knowledgeable and experienced specialist in a particular problem domain, such as a financial analyst or a production planner.
model A symbolic representation of reality that allows one to manipulate certain symbols, the inputs, and to calculate the resulting values of other symbols, the outputs. A model may be an analytical model or an intelligent model.
model development life cycle (MDLC) A set of activities, largely sequential, that leads to the development of a validated model.
model management The exercise of managerial and technical oversight in modeling.
predictive accuracy The degree to which the outputs of a model correspond to outcomes in the real world.
symbolic Relating to numerical variables, Boolean (yes/no) variables, or more abstract (e.g., high, medium, low) variables.
theoretical validity The degree to which the structure of a model is supported by a generally accepted theory of real-world entities and processes.
validity A combination of a model's predictive accuracy, insightfulness, and theoretical validity.

A MODEL is a symbolic representation of reality. The symbols may be numerical, Boolean (yes/no), or more abstract (e.g., high, medium, low). A model may be an analytical model that relies on operations such as simulation or constrained optimization, or it may be an intelligent model that imitates the reasoning processes of knowledgeable and experienced specialists in a particular problem domain. Model building differs in each of these two cases, but there are important similarities. The principal similarity is that there is a life cycle in model building, a model development life cycle (MDLC), similar to the systems development life cycle for information systems. In this article we examine the MDLC and then describe the building of each of these two types of models in terms of the MDLC.

I. MODELS AND MODEL BUILDING

Before discussing the process of model building, it is instructive to ask why models are built in the first place. In other words, what type of people build models and why do they build them? The question is even more complicated, since the people who build models and the people who use them may be different. So

we ask, what do the users of models do with the models that they build or that others build for them?

There are two principal categories of builders and users. The older category is scientists, such as physicists, chemists, and biologists. Scientists build models in order to better understand reality. They construct theories of reality (e.g., theories of subatomic particles or of chemical reactions) and they construct models (primarily, quantitative models) describing their theories. They then use the models to predict what would happen under certain conditions and compare the results with real-world observations. If they find that the predictions of the model do not correspond with their observations, then there was something wrong with their theories. If they do correspond, then the scientists are given confidence to proceed further in developing their theories.

A more recent category of model builders and users, one that we will be concerned with in this article, are people who wish not only to understand the real world but also to control (or influence) it. These people are engineers, operations researchers, financial analysts, and the like. For example, an operations researcher may build a model of a production process that describes the flow of transactions (e.g., raw materials, piece parts, subassemblies, and finished goods) through the production process. The model might calculate any delays caused by long queues forming in front of some of the production facilities. The operations researcher would then change some of the features of the production process, such as the processing capacities of certain facilities or the production scheduling rules, and see if the delays are reduced. The advantage of the model is that one can experiment with the model without incurring the cost and inconvenience of conducting experiments in the real world.

Models of this second type, called decision models, have two types of input and one type of output. The inputs are decision variables (such as factory schedules) and environmental variables (such as forecasted demand for finished goods). The outputs are measures of cost or performance, such as delays, throughputs, and total production cost. There may also be parameters in the model, such as unit production costs or constraints on production capacity. Parameters are similar to environmental inputs, except that they are programmed into the model and are assumed to change less frequently.

We will be concerned here with this second class of users/purposes. In other words, when we discuss models we will assume implicitly that they are decision models and not purely scientific models. However, many of the comments made here will apply to scientific models as well. There are still other types of symbolic models, such as the models of real or artificial worlds used in computer games, but we will not be concerned with them in this article.

We now turn to the model-building process. The principal issue here concerns the sequence of stages that one will encounter throughout the life cycle of a model's development and implementation. This is similar to the well-known systems development life cycle (SDLC) used in other types of systems development efforts. We will present an MDLC, similar to the SDLC, and demonstrate its applicability to model building.

We begin in Section II by describing the MDLC. The MDLC consists of four stages, and we identify and explain each of them. Then in Section III we describe the building of analytical models and the role of the MDLC. In Section IV we describe the building of intelligent models and the role of the MDLC. We conclude in Section V by examining briefly the concept of model management.

There are four ancillary issues not addressed in this article. The *first* is model ergonomics—that is, the design of user-friendly front ends that (1) receive data and instructions from the user, (2) pass them on to the model, (3) receive outputs from the model, and (4) provide these outputs to the user. These are often Graphical User Interfaces (GUIs). The issues here include the design of input forms and output reports, both interactive (screen layouts) and hard copy (printed reports). A related issue is the design of command languages that users can employ to provide instructions to model solution algorithms and user interfaces.

The *second* issue is computational feasibility. This refers to the demand placed on computational resources, primarily processing capacity and memory. Processing capacity is typically the speed of such calculations as floating-point additions and multiplications. Memory may be primary or secondary. Primary memory is internal to the CPU—that is, RAM in microcomputers or core memory in larger machines. Secondary memory is external to the CPU—that is, disk or tape. Of course, direct access secondary memory (i.e., disk) will trade off for primary memory whenever components of a program are paged from disk into core, resulting in virtual (primary) memory. For small models this is seldom an issue. Problems usually arise in connection with large models.

In other words, the issue here is scalability. Much theoretical research has been done on the scalability of certain types of algorithms (e.g., the simplex method of linear programming) or problems (e.g.,

linear programming problems more generally). This area is called computational complexity. In addition, some scientists have focused on issues of computational feasibility and scalability in their own disciplines. One example is computational economics, which is devoted to the solution of large-scale economic models by efficiently inverting large matrices. Another example is numerical relativity, which is devoted to the efficient solution of the extremely complex partial differential equations that arise in general relativity theory.

The *third* ancillary issue concerns the building of models that are not symbolic. There are three types of nonsymbolic models:

1. *Physical models:* The physical models most widely used in industry are pilot plants, which are small-scale models of large plants. Pilot plants are most commonly used in continuous processing industries, such as the chemical and petroleum industries. Since the catalysis of certain chemical reactions can often be scaled up from small to large settings, pilot plants can be useful prototypes in the design of large chemical plants.
2. *Analog models:* These are models wired into analog computers. They simulate real-world variables such as material flows with electronic properties such as electrical current. Although analog computers were a serious alternative to digital computers fifty years ago, this is no longer the case. However, they may be appropriate for very specialized problems.
3. *Iconic models:* These are diagrammatic models, such as program flowcharts, data flow diagrams for business processes, and entity-relationship (ER) diagrams for databases. Iconic models are visually appealing, but they do not allow the more precise calculations provided by symbolic models. However, they are often useful in documenting symbolic models or in describing the real-world systems to be modeled.

The *fourth* ancillary issue is the construction of computational procedures that are similar to decision models but are not decision models themselves. Typically these are statistical procedures used for estimation or hypothesis testing. There are three ways in which statistical procedures may be used in building decision models. First, they may be used to provide estimates, such as sales forecasts, that will appear as model inputs. Second, they may provide estimates, such as unit costs, that will appear as model parameters. Third, they may be used to test hypotheses that are useful in model building. For example, if the (multivariate) correlation or (nonparametric) degree of association between two variables is not statistically significant, then a model builder will probably decide to simplify the model by eliminating consideration of any interaction between these variables.

These four ancillary issues would certainly come into play if we were to take a more enlarged view of model building, but we will not address them here. To do so would dissipate our discussion beyond the bounds appropriate for an article of this length. Therefore, we now turn to the MDLC, after which we examine its application to analytical models and intelligent models.

II. THE MODEL DEVELOPMENT LIFE CYCLE

We begin by describing life cycles as they apply to information systems in general. A life cycle is a sequence of activities that collectively makes up the systems development process. A typical example is as follows: requirements analysis, feasibility study, analysis, design, implementation, and maintenance. There are two reasons for identifying life cycles in information systems development. The first is to guide the early stages of the systems development process. Problems that become manifest during the later stages of the life cycle often have their roots in the earlier stages, and it is useful during the earlier stages to understand and anticipate the later stages. For example, mistakes made in requirements analysis for a system may not become apparent until the system is implemented. Similarly, ambiguities and errors in system documentation may not be recognized until a problem arises and it becomes necessary to modify the system.

The second reason for identifying life cycles is that they offer a foundation for project documentation and control. In general it is necessary that information systems be described in appropriate documentation, such as systems manuals and user manuals. But it is also necessary, or at least useful, to document systems development efforts as well. The point at which one stage, such as analysis, ends and another stage, such as design, begins is called a project milestone. This often requires documentation of the current status of the development effort in the form of a milestone status report, which the users of the system must formally approve before the next stage can commence.

It is widely recognized that a major cause of information systems failure is inadequate user involvement in the development process. Insisting that users

formally approve milestone status reports before projects can continue is an important method for encouraging users to become involved in systems development efforts and also in encouraging systems developers to solicit user involvement.

Of course, no sequential listing of project activities is exact. Earlier stages may be revisited during later stages as problems arise. However, a sequential listing of stages, in the form of a life cycle, is frequently a useful guide to information systems development, resulting in an SDLC. There are several versions of the SDLC, but we will not address them or their differences here. Rather, we progress directly to the MDLC.

The MDLC consists of four stages, as follows (see Fig. 1).

A. Stage I: Problem Recognition and Identification

People often overlook this first stage of the MDLC because it takes place before most of the "real" work—or the technical work—is performed. But one should understand how model development efforts are initiated, if only so that one can encourage or discourage their initiation in the first place.

Model development efforts can be inspired by two types of sources, internal and external. *Internal* sources are typically specialists in model building, such as specialists in linear programming or expert systems. They are sometimes thought of as "people with a hammer looking for a nail," but that is often an unfair characterization. It is certainly reasonable for specialists to seek out problems that require their specialty. If an organization hires people proficient in a particular form of modeling, such as linear programming and systems dynamics, nobody should be surprised if they begin to look for problems for which their knowledge and experience are appropriate. In fact, that is probably why they were hired in the first place.

External sources differ in they are in a large part beyond the control of an organization. However, organizations can do a lot to encourage or discourage external influences. Specialists—such as engineers, financial analysts, and market researchers—often maintain contact with their counterparts in other organizations, such as corporations, professional organizations, government agencies, and universities. This is done through professional societies, conferences, and publications, as well as by less formal means. These specialists are frequently aware of modeling efforts in comparable organizations, and these efforts can be a productive stimulus for modeling innovations.

During this initial stage of the MDLC it is necessary to identify the problem domain. This consists of identifying those aspects of reality that are to be modeled

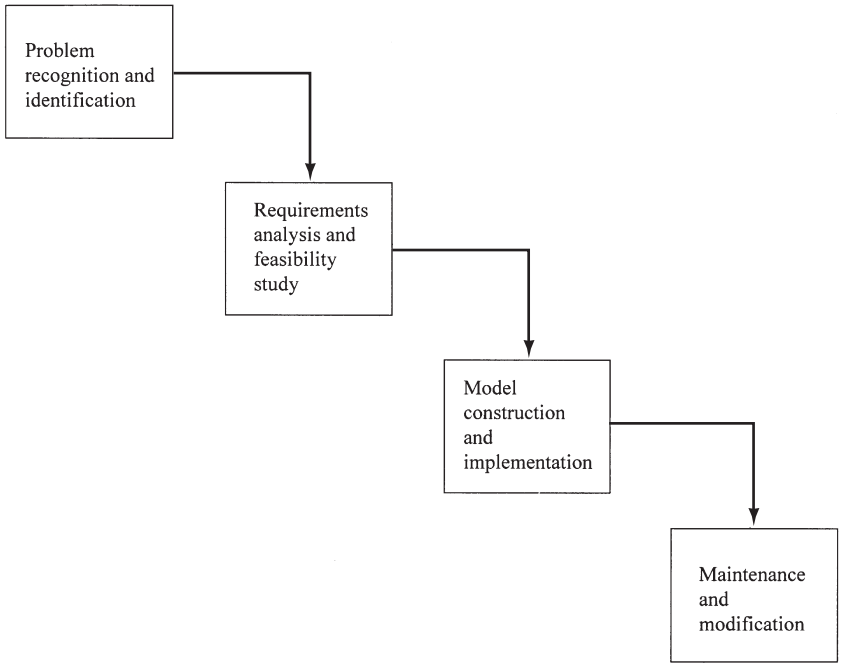


Figure 1 The model development life cycle (MDLC).

and therefore, to recognize explicitly those aspects that are not to be modeled. For example, a model of certain parts of a supply chain, such as production, may be expanded to include other parts of the chain, such as purchasing or distribution. This requires judgment as to (1) what can reasonably be accomplished and (2) the value, in terms of improved decision making, of such an expansion.

A common problem in decision modeling, as in other areas of systems development, is "mission creep." This means a gradual expansion of the boundaries of the system being modeled to include other aspects of reality. This can cause a modeling effort to expand to the point that it is out of control because the size of the effort becomes too large or too complex to be managed well. Explicit identification of the problem to be solved may be helpful in preventing, or at least mitigating, mission creep.

B. Stage II: Requirements Analysis and Feasibility Study

This stage of the MDLC includes a determination of what the potential users of a model would like the model to do for them. It also includes a determination of what information a modeling effort might provide. Thus, it will result in a compromise between what the users want and what can realistically be accomplished.

In order to do this it is necessary to specify in very general terms the type of model to be developed. For example, it may be decided that the model will probably be a simulation, but it may not be clear whether the simulation will be deterministic or whether it will be stochastic—that is, whether the simulation will incorporate random events.

An important function that should be performed during this stage is the determination of risk. This includes the possibility that the system will not function properly, that the users will find that the system does not meet their objectives, or that the systems development effort will be late or will run over budget. There are three determinants of risk. The first is the size of the model. Large models usually present problems not found in smaller models. These include computational feasibility and problems in project coordination. The second determinant is the degree to which the objectives of the modeling effort have been clearly identified at the start of the effort. Its purpose is to avoid confusion and disagreements later on and to prevent mission creep. The third determinant is the modeling technology, and especially the familiarity of the model builders with the type of model to be constructed.

In addition, there is an important characteristic of a model development effort that affects project risk. This is the degree to which the eventual users of the model are informed and consulted throughout the MDLC. If the eventual users are also the builders, then there is no problem. Otherwise, communication between builders and users is important, and often essential, for project success.

Yet another task performed during this stage is a feasibility study. The term "feasibility" can be misleading, because it suggests a Boolean classification (i.e., a model-building effort is either feasible or infeasible). Rather, the purpose of a feasibility study is to determine how reasonable the modeling effort is from several points of view. Five points of view are generally considered, as described by the acronym TELOS, taken from the first letter of each of the five types of feasibility.

The first type of feasibility is *technical feasibility*. With small analytical models this may not be a problem. But with large models and intelligent models it may become a serious problem.

The second type of feasibility is *economic feasibility*. The model development effort may be technically feasible but too costly. This is often true of very large models or models that require large databases.

The third type of feasibility is *legal feasibility*. Legal issues have seldom arisen in modeling efforts in the past, with the exception of models embedded in military weapons systems where the system resulted in legal conflicts between the government and the (private) defense contractor. However, there may be more legal conflicts in the future, especially with regard to intelligent models used in life-threatening situations, such as medical expert systems.

The fourth type of feasibility is *operational feasibility*. This refers to the acceptability of the modeling effort in a particular organization. This will include any Luddite resistance to technological innovation as well as political resistance to any new organizational structures that may result from the modeling effort. Luddite resistance will sometimes arise with regard to intelligent models, and political resistance will sometimes arise with regard to large models and therefore, large modeling efforts.

The fifth type of feasibility is *schedule feasibility*. Some modeling efforts have flexible milestones, and a limited degree of schedule slippage will not create serious problems. But in some cases, a model is needed for a particular decision that must be made by a certain date, for example, the decision to accept or reject an offer from an outside supplier or customer. The question here is whether time is a binding constraint and if so, whether the constraint can be met.

Finally, we ask the following question: How does a model builder reconcile any differences that might arise between user needs and desires? Suppose the stated desires of the potential users of a modeling effort differ from what the model builders believe the users really need. This problem can arise in any staff support operation. A user may ask for something unreasonable, for example, asking a model builder to model a process for which accurate data cannot be found. Or the user may impose constraints, such as economic or schedule constraints, that render the modeling effort infeasible. Or the user may instruct the model builder to ignore certain relevant issues, such as legal issues.

A model builder should always remember that he or she is a professional and can withdraw or resign if asked to behave unprofessionally, but it will seldom come to that. A more common, and more reasonable, solution is to establish communication with the user so that problems of this type can be resolved by less drastic means. This communication should be established as early as possible in the MDLC and should be continual, that is, it should not be allowed to lapse.

C. Stage III: Model Construction and Implementation

This is the "technical" part of the MDLC, and as far as some model builders are concerned, it is the entire MDLC. It consists of four substages, as follows.

The first substage is *elaboration of model structure.* This consists of two tasks. The first task is a more detailed specification of the model type, such as whether a constrained optimization model should be linear or nonlinear or whether a simulation will be deterministic or stochastic. This will be discussed in the following two sections. The second task is to decide whether the model is to be deterministic or stochastic. Some models, such as queueing models, will be stochastic. But most models can be either deterministic or stochastic. For example, in a time-step simulation the calculations within time stages could be deterministic, in which case only one run need be made, or they could be stochastic, in which case multiple runs must be made with different random numbers and the results averaged. In addition, variation about the mean (e.g., in the form of a standard deviation) may also be calculated and reported.

The second substage is *parameter estimation.* Each relationship in a model will be one of three types. The first is definitional, for example, net income is defined as revenue less expense. The second is statistical, for example, a demand schedule may be obtained by regressing volume on price, and possibly on other variables as well. The third is subjective, for example, certainty factors in rule-based expert systems are usually estimated subjectively.

The third substage is *model implementation.* There are two types of implementation, computational and organizational. Computational implementation consists primarily of coding and system testing. Organizational implementation includes user testing, documentation (i.e., the preparation of user manuals and system manuals), and training. The training will generally be user training, but in the case of large models, it will also include training the computer specialists who will execute the models and the modeling specialists who must maintain the models.

The fourth substage is *model validation.* There are three criteria for model validity. Not all of them need be present, since one or even two of them may be irrelevant in a particular instance. The first criterion is predictive accuracy. This is generally measured by (1) taking events not used in the construction of the model, (2) running the model, (3) comparing the model prediction with reality, and (4) calculating a measure of error, such as a mean square error or a mean absolute deviation.

The second criterion is insightfulness. The question here is whether the model reveals any interesting qualitative insights about the real world. For example, the model may suggest that there will be discontinuities or cycles in certain phenomena, or it may suggest that one or more of the outputs will be insensitive to one or more of the inputs. This could turn out to be useful information, even if the predictive accuracy of the model is low. For example, it may be useful to know that certain variables are cyclic, even if the model fails to predict accurately the frequency or amplitude of the cycles.

The third criterion is theoretical validity. The question here is whether the model is based on an accepted theory of reality. For example, a model of consumer behavior will contain relationships between certain inputs (e.g., product attributes and consumer demographics) and outputs (i.e., purchasing decisions). However, the relationships may be based either on theories of consumer psychology or on ad hoc speculations of the model builder. In some cases the resulting relationships may be the same, but users will generally have more confidence in the outputs of the models if there is a proper theoretical justification for them.

D. Stage IV: Model Maintenance and Modification

Some models are used once or a few times and then discarded. An example is a model that is built to support a one-time decision, such as the construction of a major production facility. Other models continue to be used for years or even decades after they are built, and these models must be changed over time in order to adapt to new circumstances. There are two types of change, maintenance and modification.

Maintenance consists of gradual changes in a model made in response to gradual changes in the environment. Examples are minor errors discovered in the model and changes in external conditions that affect the model (e.g., new government regulations that affect a personnel planning model). Modification is similar to maintenance except that the changes are more disruptive. An example is the addition of a new production facility that must be reflected in a logistical model.

Maintenance and modification are of special interest. For many models the vast majority of the expense and effort in the MDLC is incurred in this final stage. Therefore, models should be designed for maintainability. For example, they should have easily understandable variable names, well-documented programs, and for that matter, well-conducted feasibility studies. All of these occur in the first three stages of the MDLC. The major point here is that mistakes made or sloppy work performed during the first three stages of the MDLC may not become apparent until the fourth stage, and this is where most problems will become manifest.

III. BUILDING ANALYTICAL MODELS

Let us suppose that we wish to build a production-scheduling model. The model would assist a production scheduler in (1) preparing production schedules and (2) modifying the schedules in response to unanticipated events. Examples of unanticipated events are equipment breakdowns and delayed arrival of raw materials. The inputs to the model would be (1) proposed production schedules and (2) environmental conditions, such as consumer demand and descriptions of any unusual events. The output would include such variables as the cost of production and the throughput of the factory. The user of the model would experiment with the model to find a suitable production schedule.

There are two ways to build such a model (see Fig. 2). The first is to gather real-world data about the production process and to write a program that simulates the process. The real-world data used to estimate the model parameters might include cost data, the times needed to perform various factory operations, and the probabilities that certain subassemblies will fail a quality inspection and will require rework. Such a model is called an analytical model.

There is, however, a second approach. Instead of gathering data about the real world, we may instead identify one or a few successful production schedulers and attempt to model them. That is, we may build a model of the reasoning processes of people who are skilled at production scheduling and at reacting to unanticipated events. Identifying such people in a particular organization is seldom difficult; everyone knows who they are. Such a model is called an intelligent model, or sometimes an expert system or a knowledge-based system. In this section we discuss the building of analytical models. In the next section we discuss the building of intelligent models.

Simulations, such as the production-scheduling simulation described above, are typically dynamic. That is, they describe a process, such as a production process, as it evolves over time. However, there are several ways in which time can be represented. It can be represented as a series of time stages at fixed intervals, such as once per minute, day, or year. This is called a time-step simulation. The output of each stage (e.g., inventory level, working capital, etc.) becomes the input to the next stage. These variables are called state variables because they describe the state of the process at the interface between two successive time stages. The simulation calculates the changes in the state variables that take place during each of the time stages. Most time-step simulations of this type are programmed in general-purpose scientific languages, such as FORTRAN, BASIC, or Pascal.

There is another type of time-step simulation. In the example described above, time was a discrete variable. However, it can also be continuous. Of course, in a digital computer, as opposed to an analog computer, time must be represented in discrete form. But it is possible to make the time stages sufficiently brief that for all practical purposes they represent a continuous process. The model is then a system of simultaneous differential equations, usually nonlinear and occasionally discontinuous and with time lags. An example is the discipline of industrial dynamics, which describes industrial processes in terms of differential equations and simulates them. There are related

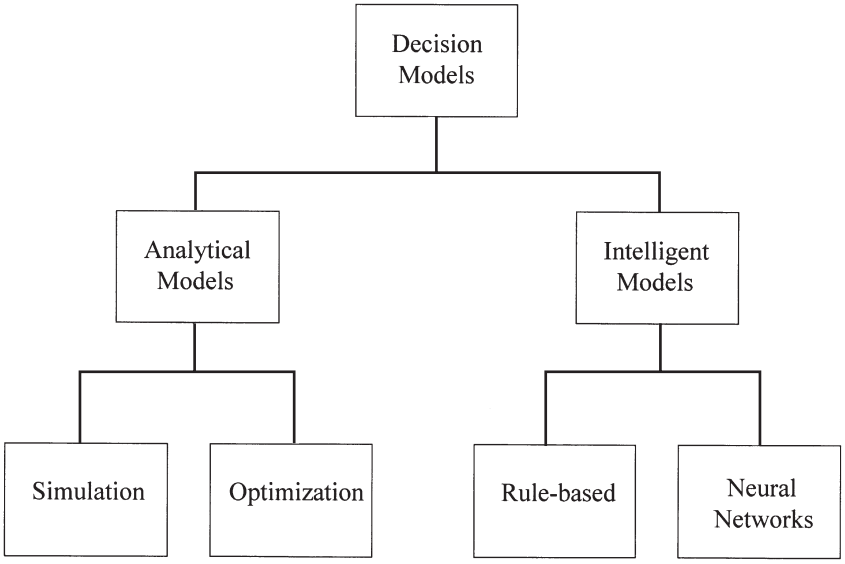


Figure 2 Types of models.

disciplines such as urban dynamics, world dynamics (which was used in the infamous "limits to growth" study), and more generally systems dynamics. Models of this type can be programmed in general scientific languages, but they are often programmed using special-purpose languages such as DYNAMO or packages such as STELLA.

Yet another type of simulation is an event-step simulation, in which the time stages are of variable length. These are typically queueing simulations of service facilities, such as vehicle maintenance facilities. Vehicles arrive at random times and are serviced (e.g., inspected, repaired) at service facilities for which the service times are also random. There are special computer languages and packages, such as General Purpose Simulation System (GPSS), for this type of simulation. Event-step simulations are typically stochastic. That is, random numbers are generated representing such phenomena as arrival and service times, and multiple simulations are performed to calculate average behavior and measures of variation (such as standard deviations). Simulations that make use of random numbers are called Monte Carlo simulations.

An alternative to simulation is optimization. The most common type of optimization is linear programming. In linear programming there are (1) a set of decision variables (e.g., components of a production schedule), (2) a single objective function that is to be maximized (e.g., profit) or minimized (e.g., cost), and (3) a set of equality or inequality constraints that must be satisfied. For example, there may be upper bound constraints on the availability of raw materials or lower bound constraints on required output levels. Linear programming models are often found in continuous processing industries, for example, in oil companies and chemical companies. In many cases they are quite large and are executed using specialized packages such as GAMS or MPSX. For smaller problems the linear programming routine in Excel is often used. The discipline of structured modeling sometimes provides a useful framework for describing the components of the model and the relationships among these components.

Another type of optimization is combinatorial optimization, along with the related areas of integer programming and heuristic search. In this case the decision variables are discrete and the constraints require that only certain combinations of these variables are feasible. Combinatorial optimization is used most often in the design of transportation networks and telecommunication networks.

Although linear programming and combinatorial optimization are the most commonly used forms of optimization, other methods are occasionally used as well. One is nonlinear programming, in which the objective function and/or the constraints are nonlinear. Another is dynamic programming, which is similar to time-step simulation except that the purpose is to find an optimal sequence of decisions. Yet another is goal programming, in which there are several objective functions and the purpose is to find a compromise among them. In chance-constrained programming the constraints are random variables that must be satisfied with a certain

probability. In stochastic programming the constraints are random variables that need not be satisfied, but there are costs associated with failing to satisfy them, and these costs appear in the objective function.

The difference between simulation and optimization is illustrated in Fig. 3. Consider an organization producing a product in which the decision variable is the sale price of the product (Price) and the environmental variable is the demand schedule (Demand). A parameter programmed into the model is the cost schedule (Cost). The user of the model wishes to price the product so as to maximize profit (Profit). A simulation would have as its inputs both the decision and the environmental variables, and the user would experiment with different prices so as to maximize profit. In the case of optimization, the user would enter only the environmental variable, and the optimization procedure would do the experimenting to produce both the optimal price and the resulting profit. Simulation and optimization procedures can also be used to perform sensitivity analysis that measure trade-offs between selected variables.

Here are a few general comments on how the MDLC may apply to the building of analytical models. These comments may not always be true. Model builders should examine their specific circumstances to determine which issues in the MDLC are most important in individual cases.

Stage I: Problem recognition and identification— Since the selection of a model type is often postponed until stage II, there are few model-specific issues here. However, in some cases the model type is obvious at the beginning. That is, it may be clear that if a model is to be built, it will be a certain type of model. In that case, internal and external investigations of the model type will take place during this stage.

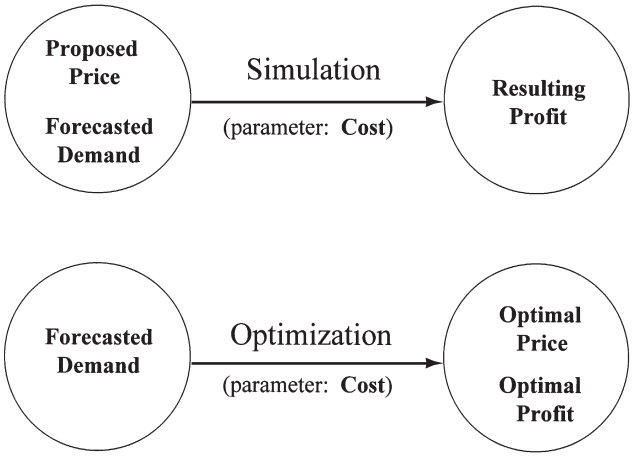


Figure 3 Simulation and optimization example.

Stage II: Requirements analysis and feasibility study—With regard to risk assessment, the principal source of risk will be size and structure, since some analytical models are quite large. With regard to feasibility studies, the principal problems will probably be economic and schedule feasibility.

Stage III: Model construction and implementation— An important issue here is whether the model will be deterministic or stochastic. In some cases, such as queueing models, the model will definitely be stochastic. But in most cases, there is a choice to be made. Model relationships will generally be definitional and statistical. Criteria for model validation will usually be predictive accuracy and theoretical validity, although in industrial dynamics models insightfulness (e.g., the presence or absence of cycles) will often be a major consideration.

Stage IV: Maintenance and modification—The models with the longest life spans (years or even decades) are usually large analytical models. It is not unusual for bugs and data errors to be found years after initial implementation. For these models, clear documentation is especially helpful.

IV. BUILDING INTELLIGENT MODELS

An intelligent model is a model of the reasoning processes of a specialist in a particular problem domain. Most intelligent models are rule-based models. An example of a rule is "if the shipment of sheet steel is more than two days late, then shut down the line that produces shelving." The "if" part of the rule is called the antecedent, and the "then" part of the rule is called the consequent. Most rules are more complicated than this in that their antecedents and consequents contain several variables. When the modeling software is presented with a particular problem instance, it attempts to link the rules (so that the consequents of some rules match the antecedents of other rules) to establish a chain of reasoning leading to a conclusion (such as a production schedule). The software can also explain how it arrived at its conclusion by displaying the chain of reasoning, that is, the chaining of rules that led to the conclusion.

The builder of an intelligent model must identify the variables that will appear in the antecedents and consequents of the rules and must then construct the rules. This is done by taking protocols of specialists while they are performing a task (e.g., preparing a production schedule). A protocol is a detailed verbal description of the details of the task given by the specialists and elicited by the model builder. After constructing

the rules, the model builder may also decide to elicit certainty factors, which describe the degrees to which the specialists believe that the various rules are valid. These are then used to calculate aggregate certainty factors for the output of the model. Thus, the model builder must be skilled at drawing out specialists, that is, at getting them to explain what they are doing while they are doing it.

There is a second type of intelligent model—a neural network. This type of intelligent model was inspired by the connectivity of neurons in the human (and animal) brain. Neural networks are presented with previous problem instances and attempt to learn which solution strategies work and which ones do not. Neural networks do this by changing the degrees to which some of the artificial "neurons" (called "processing elements") affect other processing elements. The advantage of this approach is that model builders need not spend their time constructing rules and specialists need not spend their time explaining what they are doing. This is called "overcoming the knowledge acquisition bottleneck." On the other hand, there are two disadvantages to neural networks. First, neural network algorithms often consume a substantial amount of computer time, and they may not converge on a solution. Second, neural networks, unlike rule-based models, cannot explain how they arrived at their conclusions.

A disadvantage of intelligent models is that in general they are quite brittle. That is, they may work well within restricted problem domains but are inapplicable outside of these domains. In other words, they are based on certain assumptions (about production processes, cost relationships, etc.) and give wildly improbable answers whenever these assumptions are not valid. Of course, this is true of analytical models as well. But the limits of analytical models are usually clear, whereas the limits of intelligent models are often fuzzy. This makes it especially important to define the problem to be solved as early as possible in the MDLC.

Here are a few general comments on how the MDLC may apply to the building of intelligent models. These comments may not always be true. Model builders should examine their specific circumstances to determine which issues in the MDLC are most important in individual cases.

Stage I: Problem recognition and identification—The issues that arise here are different from those that arise in analytical modeling. In some cases model builders begin by considering analytical models and turn to intelligent models when they find that they are more appropriate for their particular problem. However, there is one thing that model builders can do at this stage to facilitate the development of intelligent models. This is to define the problem very carefully in order to reduce the likelihood of mission creep and the resulting problem of brittleness, which is especially likely with intelligent models.

Stage II: Requirements analysis and feasibility study—With regard to risk assessment, intelligent models are usually not very large, so the principal source of risk is technology, not size. With regard to feasibility studies, technological and operational feasibility are usually paramount, the latter because some people may be unwilling to accept the notion of an intelligent model. Legal feasibility may be an issue when intelligent models are used in healthcare systems.

Stage III: Model construction and implementation—The deterministic/stochastic issue will revolve around the issue of certainty factors. Model relationships will generally be subjective, stemming from protocols elicited from specialists by the model builder, although a few relationships may be definitional and statistical as well. As a general rule, the greater the emphasis on subjective relationships, the more likely it is that certainty factors will be used. The reason is that users of models containing many subjective relationships would like to know whether and how much they should trust the outputs of these models. The principal criterion for validity will be predictive accuracy. However, insightfulness (e.g., explaining unanticipated periodic or discontinuous behavior) may be a side benefit of rule-based models.

Stage IV: Maintenance and modification—Intelligent models typically have a shorter life span than analytical models, but in some cases maintenance and modification may become issues as well. As usual, clear documentation is helpful or essential.

V. MODEL MANAGEMENT

Organizations contain many resources that must be managed effectively. These include fixed assets, working capital, human resources, and the like. This has led to the development of such areas as fixed asset management, working capital management, and human resource management. Thirty-five years ago it was suggested that stored data was also such a resource, and this has led to the established discipline of data management.

It is now clear that decision models are also an important organizational resource that must be man-

aged effectively, and this has led to a fledgling discipline of model management. Several frameworks have been proposed for model management. One is relational model management, in which models are viewed as virtual relations. Another is metagraphs, which are similar to digraphs except that they are set-to-set, rather than point-to-point, mappings. Yet another framework is object-oriented modeling, in which models and their supporting data are viewed as information objects, resulting in a modeling framework called UML (Universal Modeling Language). Each of these provides a framework for integrating collections of models with data bases and rule bases.

Frameworks such as these may eventually be helpful in managing large collections of models. They may also lead to interesting theoretical insights into model building processes and especially into the MDLC.

SEE ALSO THE FOLLOWING ARTICLES

Data Modeling: Entity-Relationship Data Model • Data Modeling: Object-Oriented Data Model • Optimization Models • Statistical Models (Non-Optimization Models)

BIBLIOGRAPHY

Basu, A., and Blanning, R. W. (1995). Metagraphs. *Omega,* Vol. 23, No. 1, 13–25.

Blanning, R. W. (1993). Model management systems: An overview. *Decision Support Systems,* Vol. 9, No. 1, 9–18.

Harmon, P., and Watson, M. (1998). *Understanding UML: The developer's guide.* San Francisco, CA: Morgan Kaufmann.

Pidd, M. (1998). *Computer simulation in management science.* Chichester, UK: Wiley.

Winston, W. L. (1991). *Operations research: Applications and algorithms,* Second Edition. Boston, MA: PWS-Kent Publishing.

Zahedi, F. (1993). *Intelligent systems for business: Expert systems with neural networks.* Belmont, CA: Wadsworth Publishing.

Monte Carlo Simulation

Krishnamurty Muralidhar
University of Kentucky

GLOSSARY

ABC classification An inventory classification system that identifies those groups of items that comprise the bulk of the total inventory's value.
bootstrapping A statistical procedure that uses resampling to construct an empirical sampling distribution of a statistic.
inventory The stock of an item or resource used by an organization.
Monte Carlo methods Those techniques that involve the use of random numbers.
simulation The reproduction of a real-world process or system.

MONTE CARLO METHODS are a class of simulation techniques that offer the ability to analyze and understand complex problems. Their potential applications include decision making under uncertainty, inventory analysis, and statistical analysis. The application of Monte Carlo methods in practice has been rather limited. This paper outlines the possible reasons for such limited applications. This paper provides illustrations of potential situations where Monte Carlo methods can be applied effectively. The paper also discusses the bootstrap method, a new statistical resampling technique that uses Monte Carlo methods extensively. Finally, this paper discusses the impact of recent advances in information technology and changes in the paradigm used to teach quantitative methods can have on Monte Carlo methods.

I. DEFINITION AND INTRODUCTION

Simulation models can be generally classified into one of three major types, namely, continuous event simulation, discrete event simulation, and Monte Carlo simulation or Monte Carlo methods (MCM). In simple terms, MCM may be any procedure that uses randomly generated numbers to solve a problem. By this definition, any simulation model can be considered as using MCM. Law and Kelton use the term Monte Carlo Method to define a procedure that uses randomly generated variables to solve problems in which the passage of time is not of consequence (i.e., the problem is static in nature). Pritsker (1995) defines the aim of discrete and continuous event simulation as reproducing the activities of the entities in a system over time, to better understand the behavior and performance of a system. We use both these definitions to classify MCM problems as those simulation problems that either do not involve the passage of time and/or do not involve entities and activities.

The earliest use of MCM can be traced to mathematicians in the 17th and 18th centuries interested in investigating games of chance. Later developments included the use of MCM for solving complex differential equations and for solving combinatorial problems. Monte Carlo methods are often applied in situations where the objective is to evaluate the integral of a function in a given region. In many instances, the properties of such functions are complex and/or unknown and hence, it may be difficult to develop closed form solutions. Numerical techniques (other

than MCM) also may not be useful because of the complexity of the function being evaluated. In such cases, MCM offers a simple yet effective tool for solving the problem. Problems of the type described above are found in practically every scientific discipline. It is not surprising then that MCM is used effectively in many scientific disciplines including chemistry, engineering, operations research, physics, and statistics.

Of the disciplines mentioned above, the two that are of relevance in an organizational decision-making context are operations research and statistics. In operations research, MCM is used mainly in analyzing queuing systems (discrete event simulation). Monte Carlo methods are also used in for some optimization problems. However, a discussion of this topic is beyond the scope of this paper. In statistics, as in other disciplines, MCM is used to solve problems that are not analytically tractable. It is also used to understand the behavior of statistical techniques when the assumptions underlying such techniques are violated. They are also often used to determine the power of statistical tests for which analytical derivation is not possible. Recently developed re-sampling techniques also use MCM. Thus, it is clear that MCM plays a large role in evaluating statistical techniques.

Monte Carlo methods have been and continue to be used extensively as analytical tools in statistical research. Even a cursory glance at the leading journals in statistics (such as, *Journal of the American Statistical Association, Annals of Statistics,* and *Communication in Statistics*) reveal a large number of papers that use MCM. Quantitative journals in business (such as *Management Science* and *Decision Sciences*) as well as journals in operations management (such as *Journal of Operations Management* and *International Journal of Production Research*) also contain numerous papers that use MCM as a tool for analysis. Given the wide acceptance of MCM by researchers both in statistics and business, one would *expect* that MCM would be used extensively in an organizational decision-making context as well. This, however, does not seem to be true. An analysis of practitioner-oriented journals (such as *Interfaces*) reveals that there are few papers that use MCM in an organizational decision-making context.

There are several possible reasons for this situation. One possible explanation is that, unlike other statistical techniques, MCM does not receive extensive coverage in courses that teach basic quantitative methods. Typically, extensive coverage of MCM is provided in graduate and doctoral level classes, and in some specialized undergraduate classes. In a general quantitative methods class, coverage of MCM is very limited. Even in classes on simulation, while MCM is often mentioned as a potential tool, more emphasis is placed on development of discrete event simulation models rather than MCM models. A second possible reason is that, in the past, implementing MCM required extensive programming effort, specialized software, and the use of mainframe computers.

Thus, while it is clear that MCM has the potential to be an extremely effective analytical tool for organizational decision making, it has currently not reached its potential. More importantly, unless more attention is paid to informing and educating decision makers regarding MCM, it will not realize its full potential. Hence, the objective of this study is to illustrate the applicability of MCM in an organizational decision-making context. The next section illustrates different scenarios that lend themselves as MCM applications. The third section provides a description of the use of bootstrapping (a resampling technique that relies heavily on the concepts of MCM) and its application in organizational decision making. The final section provides the conclusions.

II. POTENTIAL APPLICATIONS OF MCM

As observed earlier, MCM has the potential to be an effective analytical tool in organizations. In this section, we identify some examples of problems that lend themselves to such analysis. The examples provided in this section are by no means intended to be an exhaustive list of potential applications of MCM, only a small selection of such applications.

A. Decision Making under Uncertainty

Monte Carlo methods can be used effectively in any situation where there is inherent uncertainty and the problem complexity is such that an analytical derivation is extremely difficult or even impossible. For the purposes of illustration, consider the news vendor problem. In its simplest manifestation, the problem can be described as follows. The objective of the problem is to determine the number of newspapers to purchase so as to maximize expected profit. The decision is to be based on a historical, simple, discrete demand distribution for newspapers, the purchase price, and selling price of the newspaper. In the simple case, the number of potential alternatives being evaluated (the number of newspapers to purchase) is small. The profit function is also a simple linear function of the number of newspapers sold, number of newspapers not sold, purchase price, and selling price. In this sim-

ple case, an analytical solution can be easily derived. The news vendor problem is often used to illustrate the basic concepts of expected values, probabilities, and decision making under uncertainty.

In its simplest form, the news vendor problem has few direct analogies in an organizational context. In an organizational context, problems involve:

1. A large number of potential alternatives
2. A complex demand distribution
3. A complex pay-off function
4. Non-mutually exclusive alternatives
5. Multiple, related alternatives for related products
6. Decisions to be made at multiple stages

When a real-life problem consists of one or more of the above, an analytical solution to the problem may not be available.

Consider for the sake of illustration, the pay-off function. In the simple news vendor problem, the pay-off function is linear and is computed as the (newspapers sold $\times$ Profit per unit) $-$ (newspapers unsold $\times$ Loss per unit), where profit and loss per unit are simple functions of cost, selling price, and some simple penalty for unsatisfied demand. Let us consider the last factor, namely, the penalty for unsatisfied demand. An organization may wish to evaluate, when demand exceeds supply, the impact of a policy of providing unsatisfied customers with a coupon toward their next purchase. In such a case, it would be inappropriate to consider the value of the coupon itself as the cost since it is unknown whether the customer will actually use the coupon. Marketing research indicates that whether a coupon is redeemed or not may be affected by the coupon proneness of the customer, the attractiveness of the coupon, coupon expiration dates, etc. The models for coupon redemption are, in most cases, complex and not easily tractable analytically. In situations such as these, the only feasible analytical tool may be MCM. The organization could use MCM effectively to analyze the impact of this policy. In addition, the organization could also easily conduct sensitivity ("what-if") analysis to study the impact of changes in the characteristics of the customers and/or coupons on the solution.

It is easy to visualize other complexities in the problem as well. For instance, the pay-off function in the simple problem was simply the profit resulting from the sale of the newspapers. In real-life situations, decisions such as these may result in revenue flows over periods of time. The organization may then wish to evaluate the not just profit, but the net present value of these revenue flows or the internal rate of return. Unless these revenue flows have well-defined mathematical properties, it may not be possible to analytically identify the best alternative. Similarly, a complex demand distribution may also prevent the analytical identification of the best alternative. Considering a family of substitutable products will also complicate the problem further, as would the case where competing products are considered. Finally, decision makers may also want to consider decision criteria other than profit maximization.

Thus, although the simple news vendor problem may not have direct use in an organizational context, adding even small complexities to the simple model makes it more relevant in an organizational context. When such complexities are added to the model, analytical derivations may not always be possible, and MCM may present the only viable alternative method of analysis. Evans and Olson provide an excellent example of the application of MCM for the evaluation of a new energy service offered by the Cinergy Corporation. The example illustrates the effectiveness of MCM as an analytical tool for evaluating alternatives and/or for performing sensitivity analysis for such problems.

B. Inventory Analysis

Elements of statistical analysis also play a crucial role in inventory analysis. Organizations often implement policies and procedures that reduce time, effort, and expense, required to monitor and control for low value, high volume inventory items (often referred to as type "C" items). The EOQ model (or a variation of the EOQ model) is often used to determine the order quantity for such items. Continuous review is used in conjunction with the EOQ model and a fixed quantity is ordered as soon as a pre-specified reorder point is reached. In some cases, the organization may implement a periodic review system where the product is automatically reordered at fixed intervals. The quantity to be ordered varies depending on the inventory on hand at the time of reordering. A comprehensive discussion of the EOQ model, periodic, and continuous review systems can be found in any introductory operations management textbook.

Monte Carlo methods are often used by researchers to analyze the impact of different statistical assumption on inventory control. A casual perusal of the leading journals that publish research in operations management reveals a large number of papers using MCM as the analytical tool for research in inventory control. While there are some papers that have

addressed the use of MCM in practice, these articles tend to provide suggestions for the appropriate use of MCM, rather than describing the actual use of MCM.

Monte Carlo methods can be used effectively to understand and implement inventory control due to the inherent nature of the problem. Statistical distributions play a crucial role in the inventory control of type C items in practice. The simple EOQ model assumes that both the demand and lead times are deterministic. It is unlikely that in real-life applications that this assumption will be satisfied. Hence, the normal distribution is often used to describe demand and/or lead time distributions in analyses, and it is possible to analytically derive the reorder point and order quantity. Just as with the news-vendor problem, however, the inventory analysis could quickly be complicated when some of the simple assumptions are relaxed.

In practice, inventory analysis rarely conforms to the simple assumptions that are used. In these cases, the distribution of demand and/or lead-time could follow distributions other than the normal distribution. This in itself should not be a problem as long as the distribution being evaluated has clear mathematical properties (such as the gamma, beta, log-normal, etc.) in which case closed form solutions are still possible. However, even in these cases MCM can be used as a sensitivity analysis tool to evaluate the impact of the selection of a specific distribution for demand and/or lead-time.

Another practical aspect of inventory analysis is the common "bundling" of products. In many instances, organizations order items not on an individual basis, but order several items from the same supplier at the same time. Organizations may "bundle" orders for several reasons including price breaks, reduction of paper work and ordering costs, and potential shipping advantages. Bundling would also imply that for some products, it would be necessary to use order quantities and/or reorder points that are not "optimal." When such complexities are included in the simple inventory model, MCM provides a simple and effective tool for analyzing a complex situation. Such analysis, in addition to providing estimates of costs, can also provide valuable insights into the process that can then be used to modify policies and procedures if the situation warrants.

C. Statistical Analysis

Organizations recently have shown a renewed interest in the use of statistical analysis for improving operations. Initiatives such as the "Six Sigma initiative" promote the use of statistical analysis for the "near elimination of defects from every product." A recent article in the *American Statistician* states:

> "Statistical and related methods are being used extensively by Six Sigma companies, and by literally thousands of practitioners systematically using statistics on focused projects yielding significant financial results."

It is perhaps in this context that the application of MCM has the greatest potential. From the statement above, it is clear that organizations are using statistical procedures. These procedures are usually based on assumptions regarding the population from which the sample is derived. Some of these procedures are robust (i.e., they are not sensitive to deviations from the underlying assumptions) while others are not. In many cases, the extent to which a given statistical procedure is robust can be found in textbooks, but not necessarily for all procedures and for all types of violations of assumptions. In cases where such information is not available, MCM can be used effectively to determine the extent to which a statistical procedure is robust to violations of assumptions.

For the purposes of illustration, consider the following scenario. Assume that an organization is attempting to determine whether the variance in processing time of an older machine is greater than that of a newly installed machine. The result of this test will be used to determine whether the older machine needs to be modified or replaced. Modifying or replacing the older machine is an expensive proposition and hence the organization would like to be sure that there is a definite need for this action before it is taken. The null and alternate hypotheses in this case can be stated as follows:

$$H_0 : \sigma_1^2 = \sigma_2^2$$
$$H_a : \sigma_1^2 > \sigma_2^2$$

where σ_1^2 and σ_2^2 are the variance in processing times of machine 1 and machine 2, respectively. Further let the specified level of significance be α. This specification implies that, under the null hypothesis (namely that the variances are equal), the probability of incorrectly concluding that the variances of the two populations are different (the type I error resulting from the hypothesis test) is approximately α. By specifying small values for α (usually 0.01, 0.05, or 0.10), the organization minimizes the probability that null hypothesis will be incorrectly rejected (and hence also minimizes the unnecessary cost of modifications to the machine when it is not necessary). Assume that the organization has conducted an experiment and

observed 50 processing times for each of the machines ($n_1 = n_2 = 50$). A descriptive analysis of the sample data has indicated that the samples, while similar in shape, are heavily skewed. A frequency distribution of the sample data for one of the machines is provided in Fig. 1.

The statistical procedure most often used to compare equality of variances is a simple F test. Using the 50 observations for each machine, the variances in processing times for machine 1 (s_1^2) and machine 2 (s_2^2) are computed. The test statistic for this procedure is the ratio of the sample variance of the older machine to that of the new machine as follows:

$$R = \frac{s_1^2}{s_2^2}$$

The null hypothesis is rejected if the value of R is greater than $F_{\alpha,\eta 1,\eta 2}$ (the critical value from an F distribution with a specified significance level of α, η_1 ($=n_1 - 1$) represents the numerator degrees of freedom, and $\eta_2 (=n_2 - 1)$ represents the denominator degrees of freedom).

One of the key assumptions of the test described above is that the populations from which the samples are derived are normally distributed. Earlier studies have shown that the test is very sensitive to the normality assumption. These studies have shown that when the samples are drawn from skewed populations, even if the significance level is specified as α, the actual level of type I error is different from the specified level. Thus, for some distributions, when the null hypothesis is rejected, it is possible that the probability that this decision is made in error is much higher than acceptable, which in turn could result in unnecessary expenses in modifying one machine. Given that the processing time data is known to be heavily skewed, the organization would obviously like to know whether this has an impact on the decision at hand (namely, whether the variances are equal). While earlier studies provide a general conclusion that for skewed populations, it is unlikely that the results for every specific case are readily available. In such cases, MCM can be used as a tool to investigate the impact of skewness on the F test described above. A description of a simple experiment using MCM is described below.

The objective of this experiment is to investigate the sensitivity of the F test for comparison of variances to samples drawn from a population whose characteristics are similar to those observed in the sample data. The frequency distribution provided in Fig. 1 shows that the sample data has an (approximate) exponential distribution. Hence, the organization would like to determine, for samples drawn from an exponential distribution, under the null hypothesis (that is both samples are drawn from the same population), the actual level of type I error that results when the specified level of significance is α. The organization would also like to investigate whether using larger samples would alleviate the problem.

In the simulation experiment, the population to be used for the experiment was specified by using the gamma distribution and specifying the shape parameter as 1.0. For the sake of simplicity and, since both samples are drawn from the same population, without loss of generality, the scale parameter was also assumed to be 1.0. Using `GammaInv` function in Microsoft Excel, n observations were generated to represent the sample of observations for machine 1 and another n observations for Machine 2. The variances of the two sets of observations and the test statistic (the ratio of the sample variance of machine 1 to the sample variance of machine 2) were computed. The *p* value resulting from the test was computed using the FDist function in Excel. The null hypothesis was rejected if the *p* value of the test was less than the α value specified. The entire experiment was repeated 10,000 times. At the end of the 10,000 replications,

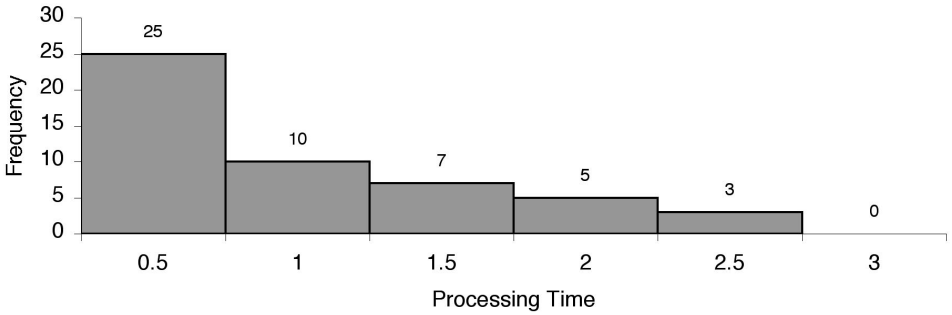


Figure 1 Frequency distribution of processing time.

the percentage of cases resulting in the rejection of the null hypothesis was recorded for the three levels of significance ($\alpha = 0.01$, 0.05, and 0.10). The entire experiment was then repeated by varying the sample size from 50–200 in increments of 50. The results of this experiment are provided in Table I.

A test would be considered robust if, when the assumptions are violated, the performance of the test will be the same as when the assumption is satisfied. One critical aspect of any statistical test is that, under the null hypothesis, the observed level of type I error (the percentage of rejections when the null hypothesis is true), must be approximately the same as the specified level of significance (α). The results provided in Table I indicate that for small sample sizes, the F test is clearly not robust when samples are drawn from an exponential population. For a sample of size 50, the observed probability of rejecting the null hypothesis when it is true (type I error) is much higher than the specified level of significance (α). Even for samples size of 100 and 150, this trend continues. However, for a sample of size 200, the observed and specified significance levels are approximately the same.

The results in Table I provide valuable information to the organization. First and foremost, the results of the experiment indicate that, for the specific case being considered (exponential population and sample size = 50), even if the null hypothesis is rejected, there is a large probability that there is no difference in the variances of the two machines. In other words, if the organization undertakes modifications/repairs to the old machine based on the fact that the F test indicated that the variance is higher, then there is a large probability (as high as about 25%), that such modifications were unnecessary. The results in Table I also suggest that the best option under these conditions would be to increase the sample size from 50 to 200. When the sample size is 200, the observed specified significance levels are approximately the same, and hence, the results of the F test can be relied upon.

Table I Specified versus Actual Significance Levels of the F Test

Sample size	Specified level (%)	Observed level (%)
50	1.00	10.64
	5.00	18.68
	10.00	24.64
100	1.00	4.23
	5.00	11.18
	10.00	17.61
150	1.00	1.74
	5.00	7.09
	10.00	12.80
200	1.00	0.86
	5.00	4.45
	10.00	9.30

It is important to note that *all* computations in the above experiment were performed on a personal computer using only Excel functions and Visual Basic as the programming language. The entire experiment required 4500 seconds to complete. Note that the number of replications was specified as 10,000. Reducing the number of replications would result in a corresponding (linear) reduction in computation time. In many cases, even 1000 or 2000 replications may be adequate. Increasing the number of replications beyond 10,000 rarely produces any improvements in accuracy. Hence, it is possible that the same results could have been achieved with even less computation time.

The situation presented above commonly arises when using statistical procedures. Real life situations rarely conform to the strict assumptions underlying many statistical procedures. In many cases, while information regarding the general behavior of a procedure to violations of the assumptions is available, information regarding the behavior of the procedure in specific cases may not be readily available. In such cases, MCM offers a simple, effective, and efficient tool either to generate new information for the specific case or to verify/validate existing information. In the following section, we describe a new MCM-based statistical procedure that provides decision makers with the ability to perform even more advanced statistical analysis.

III. BOOTSTRAPPING

Bootstrapping is a relatively new, computer-intensive statistical methodology introduced by Efron. The bootstrap method replaces complex analytical procedures by computer intensive empirical analysis. The bootstrap method relies heavily on MCM where several random resamples are drawn from a given original sample. The bootstrap method has been shown to be an effective technique in situations where it is necessary to determine the sampling distribution of (usually) a complex statistic with an unknown probability distribution using these data in a single sample. The bootstrap method has been applied effectively in a va-

riety of situations. Efron and Tibshirani provide a comprehensive discussion of the bootstrap method.

The bootstrap methodology can be described in simple terms as a resampling procedure that is used to construct an empirical distribution of the sample statistic. The empirical distribution (often referred to as the bootstrap distribution), can be used in the same way as the theoretical sampling distribution. To illustrate the bootstrap method, assume that a sample, **X** $\{= X_1, X_2, \ldots X_n)$ of size n has been collected. Further assume that the parameter of interest is θ and let $\hat{\theta}$ represent the sample statistic used to estimate θ. In the bootstrap method, a simple random sample of size n is *resampled, with replacement,* from **X.** From this bootstrap sample, the sample statistic $(\hat{\theta}_1^B)$ is computed. The process of generating the bootstrap samples and computing the sample statistic is repeated a large number (say m) times resulting in the collection $\{\hat{\theta}_1^B, \hat{\theta}_2^B, \ldots \hat{\theta}_m^B\}$. This collection allows for the construction of the bootstrap distribution of $\hat{\theta}$, which can be used to estimate standard error, construct confidence intervals, or perform hypothesis tests.

Note that the bootstrap method is free of parametric assumptions common in traditional statistical techniques. The only assumption underlying the bootstrap method is that the sample is representative of the population, a basic assumption which underlies any statistical technique. However, since the bootstrap method relies heavily on the original sample, the bootstrap method may not be effective for small sample sizes. The bootstrap method is most effective in cases where the sampling distribution of the statistic is so complex that it cannot be analytically derived and/or where the sampling distribution can be derived only under strict parametric assumptions and cannot be generalized when these assumptions are not satisfied.

Consider the example of the comparison of the variance in processing times for the two machines described earlier. Even though the performance of the F test for a sample size of 200 was adequate, the decision maker may be concerned with the violation of the normality assumption and its impact on the test. The decision maker in this case may consider the bootstrap method as a viable alternative to perform a test to check whether the variances in processing time of the two machines are different. In this example, the statistic of interest is the ratio of the variance of the old machine to the variance of the new machine. As before, let s_1^2 and s_2^2 represent the sample variance for the old and new machines, respectively. The statistic of interest is the ratio of the two variances namely R.

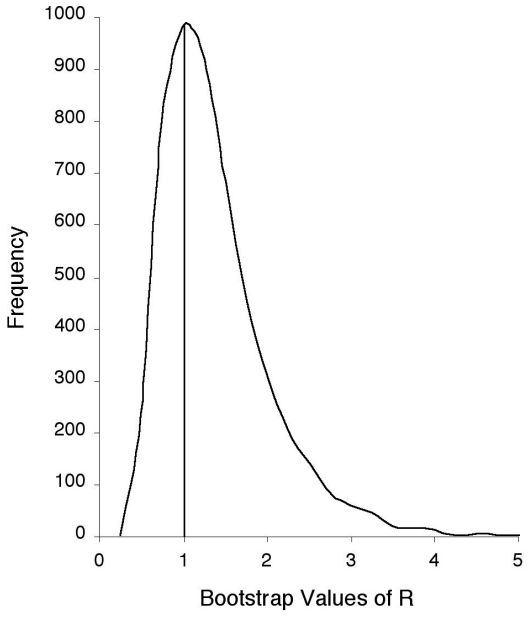
In the traditional F test, when the samples are drawn from normal populations, under the null hypothesis that the variances are equal (i.e., R = 1), the sampling distribution of R has an F distribution with η_1 and η_2 degrees of freedom. This allows a hypothesis test to be performed using the F distribution. However, when the samples are drawn from other distributions (such as the exponential) it is not always possible to analytically determine the sampling distribution of R. In these cases, the bootstrap method can be used to construct the bootstrap distribution of R.

For the purposes of this illustration, two samples of 50 observations each were generated for the old and new machine processing times from a gamma distribution with shape parameter = 1.0 and scale parameter = 1.0. Since the samples were generated from the same population, the correct decision is not to reject the null hypothesis. Using the original sample collected for the old machine (n = 50), a resample of size 50 was drawn, with replacement, from the original sample. The variance of this (bootstrap) sample $([s_1^2]_1^B)$ was computed. The process was repeated for the new machine and $([s_2^2]_1^B)$ was computed. The ratio of the variance of the old machine to the variance of the new machine was computed as:

$$R_1^B = \frac{[s_1^2]_1^B}{[s_2^2]_1^B}$$

The entire process of selecting resamples, computing variances, and computing the ratio was repeated 5000 times. Using the collection $\{R_1^B, R_2^B, \ldots R_{5000}^B\}$, a frequency distribution was constructed (Fig. 2). This frequency distribution (the bootstrap distribution of R) is an estimate of the true sampling distribution of R.

As indicated earlier, the bootstrap distribution can be used in placed of the sampling distribution of R in order to estimate the standard error, construct confidence intervals, or perform hypothesis tests. In this case, we are interested in performing hypothesis test. Figure 2 also indicates the location of the (null hypothesis) value of R = 1.0. Assume that the specified level of significance is α. Let the area of the curve to the left of 1.0, as a proportion to the total area, be represented by p. The proportion p represents the percentage of bootstrap samples where the R was less than 1.0. Conversely, $(1 - p)$ represents the proportion of cases where the R was greater than 1.0. If the value of p is very small, then it can be concluded that the variance of machine 1 was less than that of machine 2 only in a small proportion of the bootstrap samples. Specifically, if $p < \alpha$, then it implies that



R	Frequency
0.25	3
0.50	235
0.75	802
1.00	987
1.25	916
1.50	684
1.75	459
2.00	309
2.25	205
2.50	141
2.75	84
3.00	61
3.25	47
3.50	20
3.75	16
4.00	12
4.25	4
4.50	6
4.75	3
5.00	2
> 5.00	4

Figure 2 Bootstrap distribution of R.

fewer than α proportion of the bootstrap samples provided support for the null hypothesis. Hence, if $p < \alpha$, the null hypothesis can be rejected.

In this example, from the frequency distribution of R (also provided in Fig. 2), it can be seen that $p \approx 0.40$. This implies that in approximately 40% of the bootstrap samples, the ratio of the two variances was <= 1.0. Since p is larger than the usually specified levels of α (= 0.01, 0.05, or 0.10), the null hypothesis cannot be rejected. Thus, the data in this case does not support the statement that the variance of machine 1 is greater than the variance of machine 2.

The above illustration shows the ability of the bootstrap method to provide a distribution-free means for performing a hypothesis test regarding the variance of the two machines. As with the earlier example, all computations were performed using only Excel functions and Visual Basic. The total computation time to perform the hypothesis test using the bootstrap method was approximately 50 seconds.

VI. CONCLUSIONS

Just a few years ago, the experiments described above could be performed only using a mainframe computer and specialized software. Recent advances in technology have made these requirements unnecessary. All computations in this paper were performed on a personal computer (Dell Dimension with a 450-MHz processor). The only software used was Excel and Visual Basic was used as the programming language. While it is true that some specialized knowledge (specifically, programming in Visual Basic) is still needed, the fact that this software is available even in the most basic computer makes it easier for decision makers to use MCM. It is also important to note that the times reported in this study were not CPU times, but were actual elapsed times.

Educators are also adapting to the advances in information technology to enhance the capabilities of their students. There has been a recent trend toward teaching quantitative methods courses using spreadsheets. The objective of this approach is to treat students (decision makers) as the "end-users" rather than the traditional approach treating students as "an-informed-consumer-of-MS-consulting." In order to achieve this objective, this approach focuses on the use and application of statistical and management science techniques using spreadsheets. This approach reduces the focus on numerical aspects of the techniques that is often found in the traditional method of teaching such courses. With greater acceptance of this approach, recent textbooks also present the relevant information in a spreadsheet environment.

The major impact of this trend has been that techniques and procedures, such as MCM, that were, un-

til recently, considered too advanced and/or complicated to be covered in introductory classes, are now receiving coverage. Recent textbooks on quantitative methods now provide coverage of MCM. Several Excel add-in packages (such as @Risk and Crystal Ball) have also been developed for implementing MCM.

In conclusion, MCM remains a simple yet powerful alternative for analyzing complex decision making problems involving uncertainty. However, the application of MCM in practice does not seem to match the potential of MCM. This can be attributed mainly to the fact that, in the past, application of MCM required special technology and software. In addition, a lack of coverage of the topic in quantitative methods courses meant that only students who specialized in quantitative methods were likely to use MCM. These problems have been alleviated, thanks to the recent advances in technology and a shift in the paradigm for teaching quantitative methods. With these changes, it is likely that the application of MCM in practice will finally reach the potential it offers for decision makers.

SEE ALSO THE FOLLOWING ARTICLES

Continuous System Simulation • Decision Support Systems • Discrete Event Simulation • Executive Information Systems • Game Theory • Optimization Models • Simulation Languages • Strategic Planning, Use of Information Systems for

BIBLIOGRAPHY

Efron, B. (1979). Bootstrap methods: Another look at the jackknife. *Annals of Statistics,* Vol. 7, No. 1, 1–26.

Efron, B., and Tibshirani, R. (1993). *An introduction to the bootstrap.* New York: Chapman & Hall.

Erkut, E. (1998). How to "Excel" in teaching MS. *ORMS Today,* Vol. 25, No. 5, 40–43.

Evans, J. R., and Olson, D. L. (1999). *Statistics, data analysis and decision modeling.* New Jersey: Prentice-Hall.

Fishman, G. S. (1996). *Monte Carlo: Concepts, algorithms, and applications.* New York: Springer-Verlag.

Law, A. M., and Kelton, W. D. (1982). *Simulation modeling and analysis.* New York: McGraw-Hill.

Markland, R. E., Vickery, S. K., and Davis, R. A. (1998). *Operations management: Concepts in manufacturing and services,* 2nd Edition. Mason, Ohio: South-Western College Publishing.

Moore, D. S., and McCabe, G. P. (1993). *Introduction to the practice of statistics.* New York: W.H. Freeman and Company.

Pritsker, A. B. (1995). *Introduction of simulation and SLAM II.* New York: John Wiley & Sons.

Williams, G. J., Hill, W. J., Hoerl, R. W., and Zinkgraf, A. (1999). The impact of six sigma improvement—A glimpse into the future of statistics. *American Statistician,* Vol. 53, No. 3, 208–215.

Multimedia

J. P. Shim
Mississippi State University

GLOSSARY

compression Reducing the amount of data needed to carry information. It codes data to minimize the size of its representation.

hypermedia Much more inclusive than a mere extension of hypertext; therefore, hypermedia is classified as linked and synchronized media. It incorporates other media such as audio, video, illustrations, diagrams, animation, and computer graphics.

hypertext Interactive approach for presenting text and graphic information by allowing users to jump from a particular subject to related subject. It is often referred to as "nonsequential" text and "linked" text.

multimedia Like hypermedia, it is any kind of computer-based system that uses text with one or more other media types; it is known as synchronized media. Distinctions between hypermedia and multimedia have become blurred with the exception of "time constraints."

synchronized multimedia integration language (SMIL) Streaming audio, streaming video, images, and text. It allows the user to view both the streaming video and animated PowerPoint slides with certain time intervals.

videostreaming A constant stream of video that is displayed in a window on the user's desktop.

I. INTRODUCTION

Over the past decade, hypermedia technology and multimedia technology have received considerable attention from academics and practitioners. Recently, distinctions between the terms *multimedia* and *hypermedia* have become blurred except for "time constraints." Multimedia means any kind of computer-based system that uses text with one or more other media types—graphics, images, animation, sound, or motion pictures. Hypermedia is defined as a navigational tool that allows users to explore interrelated materials from large databases of multimedia information. For example, hypermedia systems can be used to store, retrieve, and represent information in varied data formats such as text, graphics, images, animation, sound, and motion pictures. Multimedia used as the systems for decision making are indeed proving to be valuable for professionals in academia, government, and industry.

Multimedia systems are useful tools for corporations because such applications can provide an endless range of information available that is easy to comprehend. Examples include corporate organization structures, employee-related data files, and portfolios of assets. Many multimedia applications employ hypertext as a means of relating textual components with associated data in a variety of media forms. The major functions that multimedia products can provide to users are (1) browsing, (2) training and education, (3) briefing/illustration, (4) learning and analysis, (5) help, and (6) referencing and on-line documentation. With these types of functions, multimedia systems can play a major role in industry by accessing information and providing users with control through navigation of data. This article is a review of multimedia technology. An overview of multimedia is provided

along with its history, components, compression techniques, and its industry. This will be followed by a review of multimedia applications. Then, the latest trends in multimedia such as SMIL will be discussed.

II. OVERVIEW OF MULTIMEDIA

A. History of Hypertext, Hypermedia, and Multimedia

Originally introduced by Vannevar Bush, and then extended and refined by Nelson, Engelbart, and others, hypertext systems were initially defined as a valuable interactive approach for presenting text and graphic information by allowing users to jump from a given subject to related ideas. It is interesting to note that Ted Nelson, who is sometimes considered to be the originator of the term, includes all media in the term "hypertext." In the industry, however, hypertext generally refers to text-only-based systems. In general, we can consider hypertext to be a sophisticated approach for interactive use of text-only databases. Figure 1 depicts the evolution of hypertext, hypermedia, and multimedia beginning in 1935 through 2000. Since the early 1990s, multimedia has gone through a technological breakthrough such as Internet/Web based multimedia, and web-based videostreaming and synchronized multimedia.

Hypertext is often referred to as "nonsequential" or "linked" text. The structure of a hypertext database (Fig. 2) illustrates how users can quickly follow documents without losing their original context. Note that hypertext software creates links among the documents/nodes using four types of links: (1) hierarchical links, (2) keyword links, (3) referential links, and (4) cluster links (Fig. 3). As shown in Fig. 3, hierarchical links branch from top down; keyword links serve as an index to provide terminology; referential links jump from one subject to another subject to retrieve or "refer" information; and cluster links merge together information.

On the other hand, hypermedia is really much more inclusive than a mere extension of hypertext; therefore, hypermedia is classified as linked and synchronized media. It incorporates other mediums such as

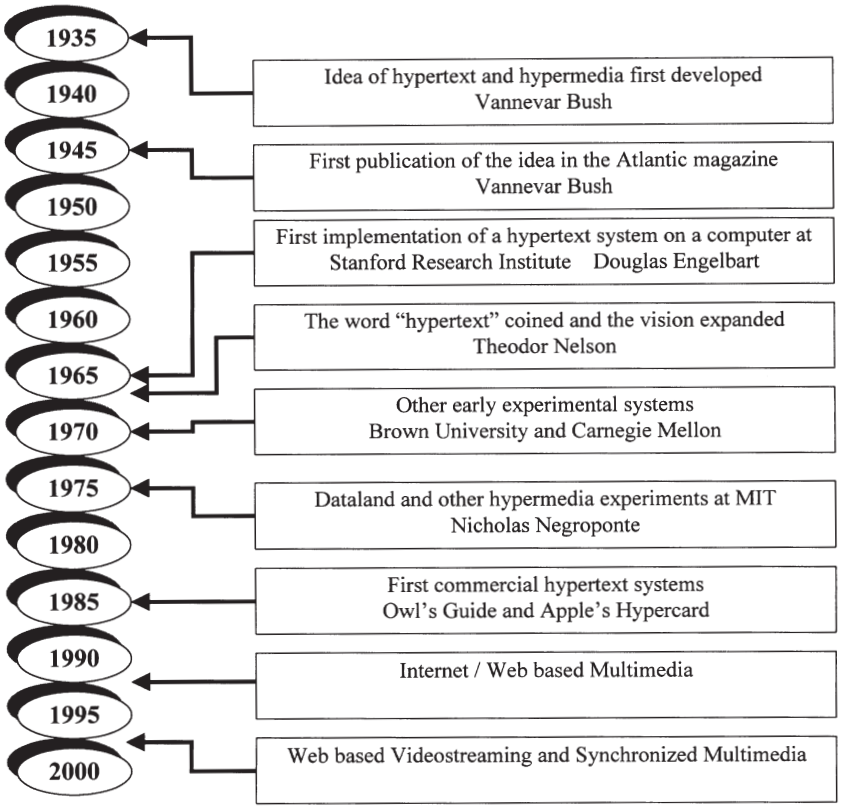


Figure 1 A history of hypertext, hypermedia and multimedia. [Adapted from R. E. Horn, *Mapping Hypertext,* The Lexington Institute, 1989; and from J. P. Shim, "SMIL and Videostreaming for Teaching Business Telecommunications and e-Commerce," *Decision Line,* July 2000.]

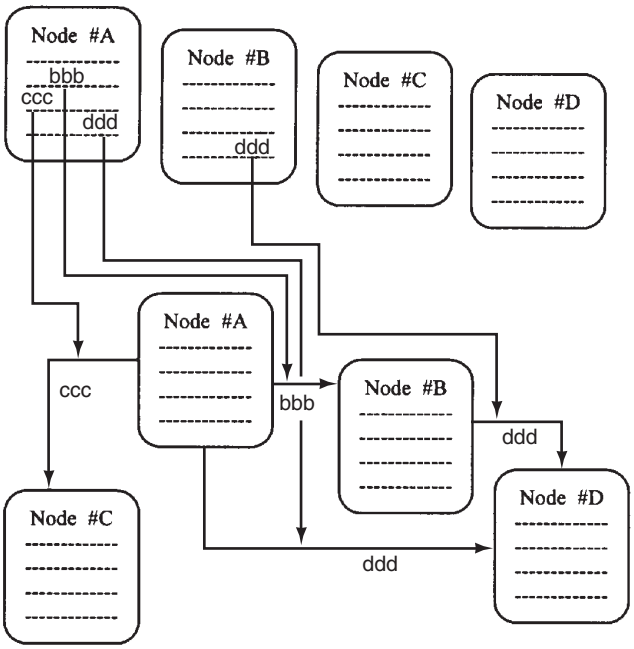


Figure 2 Structure of a hypertext database.

video, illustrations, diagrams, animation, and computer graphics. Five concrete benefits have accounted for the surge and popularity of hypermedia. Hypermedia can:

1. Save time by allowing users to browse through information
2. Offer nonlinear access to information
3. Promote a collaborative work environment by allowing to link individual pieces of information
4. Aid in the discovery of new and relevant information by indicating links to existing information
5. Present the same information in multiple media formats

For example, users (e.g., students) can read John F. Kennedy's speech in order to analyze rhetoric; they can also see and hear the speech to grasp its emotional content.

B. Multimedia Components and Compression Techniques

1. Components of Multimedia

Like hypermedia, multimedia systems are any kind of computer-based system that uses text with one or more other media types. Multimedia services are composed of the following media:

Medium	Definition
Text	Letters, numbers, punctuation, special characters, and controls, created with a text editor or word processor
Graphics	Lines, circles, boxes, shading, fill colors, created with a draw program
Images	Still pictures, expressed as the colors of many small individual picture elements (pixels), either captured and digitized from nature or painted using a paint program
Audio	Sound, including voice, music, and special effects, either captured from nature or synthesized
Video	Successive pictures presented sufficiently fast, frequently to give the appearance of smooth motion, either captured from nature or synthesized, such as by using an animation program

From Agnew, P. W., and Kellerman, A. S. (1996). *Distributed Multimedia,* Reading, MA: Addison Wesley.

Multimedia services are made possible by a number of underlying technologies. These include the processing of audio, image, video signals, hand-written data, as well as the high-quality transmission of audiovisual messages and data information. As Fig. 4 illustrates, multimedia systems add interactivity to the combination of text, graphics, images, and audio, and video (see Fig. 4).

2. Compression Techniques

The class of data signals differs from audiovisual signals in that no "real-time" transmission constraints exist. Data signals can tolerate transmission delays of hundreds of milliseconds or more without degrading their value or ultimate utility, but need accurate signals. Audiovisual signals, on the other hand, are useful sources of information and entertainment, even when the fidelity of the signal is less than perfect. Coding and synthesis are two key areas of interest in audiovisual signal processing. The purpose of coding is to achieve a compact digital representation of the signal in transmission or storage. Synthesis focuses on creating spoken or pictorial information starting from text, rather than from human speech or a real image. The four fundamental parameters of coding are bit rate, signal quality, processing delay, and complexity of implementation. The goal of coding is to decrease the bit rate while maintaining a specified level of signal quality; this is achieved through signal compression. Coding standards promote digital communication by providing interoperability of coders-decoders (codecs) from different manufacturers.

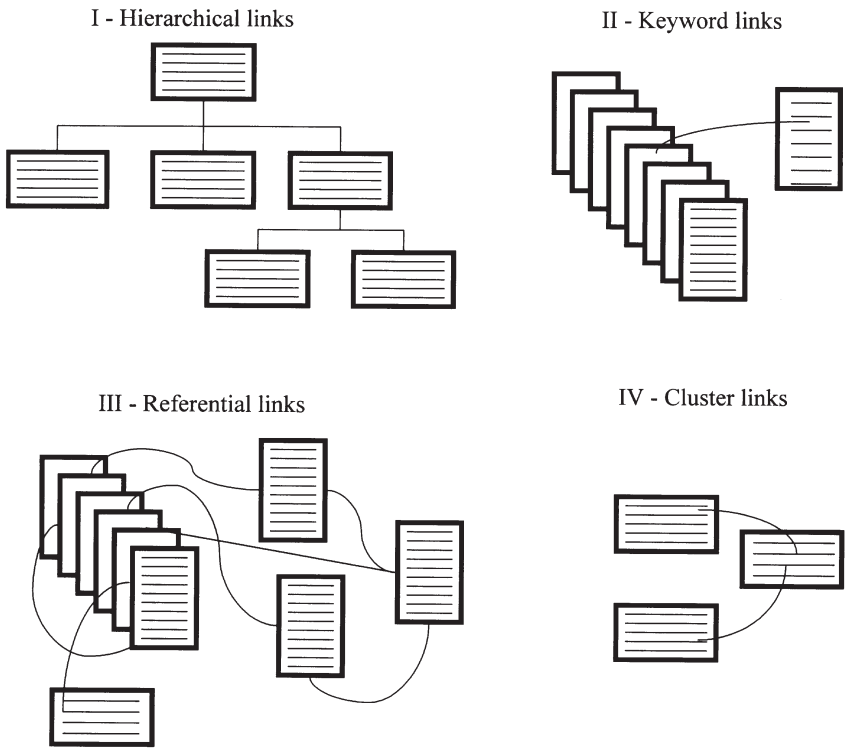


Figure 3 Four types of links.

Compression is defined as coding of data to minimize the file size of its representation. It reduces the storage requirement, increases the communication rate, and reduces redundancy prior to encryption. There are two types of compression techniques: lossy and lossless. Lossy compression assumes that some of the data is unnecessary and the human cannot notice the difference between the original object and the decompressed object. On the other hand, lossless compression assumes every bit of information is important and the original object can be perfectly recovered. The trade-off among compression ratio, data rate, and video quality is the most important type of decision in the field of multimedia. Higher compression ratios and higher data rates increase costs for users and producers. Also, a high data rate reduces

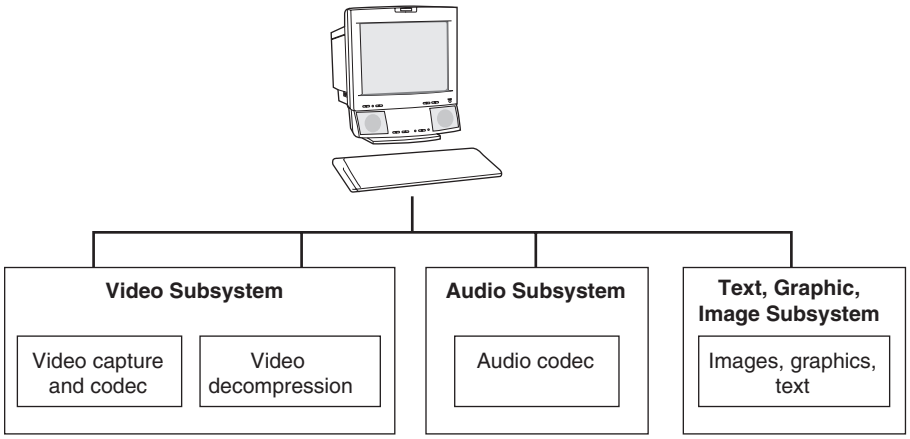


Figure 4 Components of a multimedia.

Table I Examples of Video Compression

Parameters case	Pixels per line	Lines per frame	Bits per pixel	Frames per sec	Compression ratio	Bytes per sec
JPEG	720	480	24	30	30	1 M
MPEG-1[a]	352	240	15	15	16	150 K
MPEG-2	720	480	24	30	100	300 K
Indeo	320	240	24	15	23	150 K
Indeo	640	480	24	30	184	150 K
Cinepak	240	180	24	10	8.6	150 K
Fractal	320	200	15	15	45	40 K
H.261 or P × 64	176	144	8	15	24	2 × 8 K = 16 K

[a]Note that the ratio for MPEG-1 is generally 30 for high quality and 50 for lower quality.
From Agnew, P. W., and Kellerman, A. S. (1996). Distributed Multimedia, Addison Wesley.

the number of minutes of video that a given type of disk can store. Table I illustrates some cases (on video compression) of this trade-off.

C. Multimedia Industry

The multimedia industry is a creative industry that is enabled, rather than driven by technology. A brief look at the emerging computing environment shows why such great advances have been made in multimedia applications. Table II shows the advances made in several different relevant areas. A recent study presents an interesting research on multimedia market penetration. As shown in Table III, a great number of respondents were tested to see how many different multimedia web formats they could view with their current web browser. The most highly viewed formats are, taken from the study, in descending order: Animated GIF, Flash, Java, and Shockwave.

Convergence has never been used more frequently than in today's multimedia world. Tremendous efforts have been made toward the convergence of communications (e.g, TV broadcasting), powerful computers, hypermedia (e.g., Internet), and consumer industries (e.g., digital video disks [DVD]). Prominent examples of such convergence include personal computers and TV. The hybrid is an appliance called digital TV set-top box that connects a TV set to the Internet as well as the cable system. The digital video disks is a bigger and faster compact disk that can hold video, audio, and computer data. The DVD has been

Table II Multimedia Computing Environment

Processing power	MPEG-1 decoding = 166 MHz MPEG-2 decoding = 400 MHz
Storage	23 GB/drive 10,000 RPM 5–30 Mbps transfer rate
Network infrastructure	Asynchronous transfer mode Direct satellite broadcast Cable modem Multichannel multipoint distribution systems
Digital TV	Offered in increasing number of U.S. cities 19.2 Mbps channel Much greater resolution MPEG-2 video, Dolby AC-3 audio
Video	MPEG-1 storage and retrieval MPEG-2 digital TV MPEG-4 multimedia production MPEG-7 multimedia material

Table III Multimedia Market Penetration

Animated images	
Types of format	**Percentage[a]**
Animated GIF	99.3
Flash[b]	76.8
Java	61.9
Shockwave[b]	52.0

[a]Percentage of respondents able to view specified formats.
[b]Flash and Shockwave are both commercial software that may or may not exist in the future.
From Walsh, J. (1999). "Web Multimedia Goes Mainstream." *InfoWorld* **21**(15).

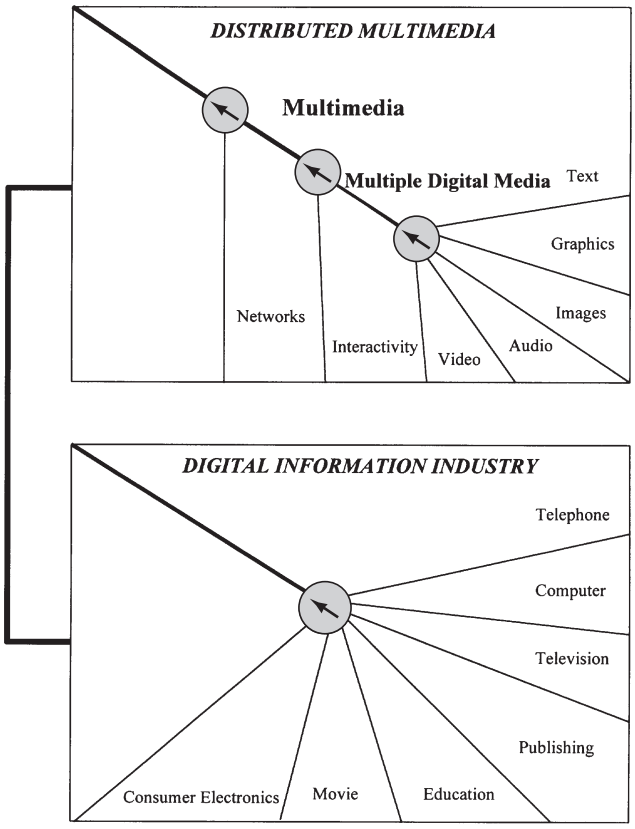


Figure 5 Integrated information and multimedia. [From P. W. Agnew and A. S. Kellerman, *Distributed Multimedia,* Reading, MA: Addison Wesley, 1996.]

widely accepted and strongly supported by major electronics and movie companies. As shown in Fig. 5, companies within many diverse industries form subdivisions of the multimedia industry; therefore, in the future, multimedia industry should be viewed as an integrated industry.

III. MULTIMEDIA APPLICATIONS

Multimedia systems have become useful tools for users because multimedia applications can make an endless range of easy-to-comprehend information available to them. Various multimedia applications have been developed and used in wide areas. The major functions multimedia systems provided to users include areas such as:

- Education and training
- Distance learning
- Information analysis and management
- Electronic games playing and entertainment
- Browsing and publishing
- Referencing and on-line documentation
- Briefing and illustration
- Video conferencing.

Over the past several decades, multimedia technology has evolved into the technologies like CD-ROM and the World Wide Web. Currently, cutting-edge multimedia consist of high-quality digital visuals and audio in an interactive package. This can be delivered on CD-ROM or over the Internet/Web, even though high quality delivery over the Web is still in its infancy.

A. Windows or CD-ROM-Based Multimedia Applications

A United States Department of Defense study found that multimedia training was roughly 40% more effective than traditional training, resulting in a retention rate that was 30% greater, and a learning curve that was 30% shorter. Trainees enthusiastically participated in learning where instructors could easily transmit and get trainees to remember important concepts and principles. According to Wilder, training time for a point-of-sale cashier dropped from an average of 8 hours to between 2 and 4 hours. Other benefits included decreased errors at the point of sale, better retention of skills, and standardization of training across the company.

In order to investigate the current and future use of multimedia, IBM Europe held a meeting of 49 participants and 20 presenters from 17 countries. Almost half of the presentations made at the meeting focused on successful multimedia applications in higher education. Multimedia applications were also demonstrated in the medical, scientific, and music fields. Multimedia can be used in the field of education and can complement it in the following three ways: (1) by enhancing communication channels between tutors and students to allow more effective open and distance learning, (2) by delivering knowledge, and (3) by enriching reference and resource documents used for education and research by adding all forms of audiovisual media.

Multimedia technology has been used as a direct mail advertising tool to reach target customers with greater brand awareness, because this technology can be used as a persuasion tool to reflect the changing nature of lifestyles of customers, time pressures, demographics, and attitudes. For example, Mattel has sold more Barbie fashion accessories and toys than ever before through its database-driven direct mail clubs for

Barbie Doll collectors who are 3–12 years old.

In many industries, growing numbers of retailers have installed interactive kiosks to place catalog orders, to inquire about service and credit, to check on the status of products being serviced, and to communicate with store managers. For example, Sears, Roebuck & Co. has spent $7 million on 6000 mini-kiosks allowing customers to place catalog orders and to inquire about services and credit. Also, K-Mart, together with IBM, has developed a multimedia kiosk pilot program to make shopping easier and more enjoyable for customers.

Multimedia systems have been used in the banking industry in dealing with high volumes of check images, even though the magnetic ink character recognition (MICR) technology has traditionally been widely used. Agency Management Services in the insurance industry have utilized multimedia technology to provide insurance agents with the functions of electronic image management, an electronic camera that produces images for computer display, fax transmission, and full access to reference manuals.

Besides service industries, manufacturing industries have developed multimedia applications because they require better tools to retrieve and represent information on the basis of huge investments made in the development of various types of information systems. Also, multimedia has been used in the transportation industry where customers have access to multimedia-based cargo booking and tracking systems that handle large amounts of shipping transactions. Airline computer reservation systems have also started to use multimedia; American Airlines has added multimedia features to the text-only SABRE computer reservation systems to improve travel agent productivity and to increase the number of hotel bookings by providing on-line hotel information in the form of text, color, photographs, and maps.

Multimedia applications are being used as a strategic tool in the areas of training, marketing, sales, decision making, and public relations in the service and manufacturing industries. The applications of multimedia have demonstrated the strategic opportunities to support decision making, training and learning, marketing and sales, business presentations, and public relations.

B. Internet/Web-Based Applications

With the advent of the World Wide Web and the release of Mosaic, the first graphical web browser in 1993, the last decade of the twentieth century has witnessed great advances in multimedia technology and application. The boom relies heavily on the radical technical breakthrough in the fields of video data compression, high-quality, real-time encoding and decoding, networks, and the Internet. Today's industries are seeking the following applications in order to increase revenues and profits:

- Desktop videoconferences with collaboration
- Multimedia storage and mail forwarding
- Consumer edutainment, infotainment, and sociotainment
- Shopping and advertising
- Digital libraries
- Video on demand
- Educational and health applications
- Hybrid applications

Very recently, a project entitled THRO at the University of Cincinnati used case studies to simulate real-world human rights problems and present ethical decision-making opportunities to students on-line via web-based multimedia.

As the new millennium approaches, the use of audio clips and video manipulation enable inexpensive and creative expression using multimedia technology in applications readily available to any users with a computer and the desire to express themselves visually. Streaming multimedia is starting to find its way into mainstream industry. Hewlett-Packard uses streaming video to broadcast live events and promotional videos of product launches. Recently, Jacobs Engineering Group Inc. has used streaming media technology for group collaboration on complex, three-dimensional diagrams of chemical, petroleum, and pharmaceutical semiconductor facilities. Benefits expected are reduced cycle time of projects and improved client input.

IV. SYNCHRONIZED MULTIMEDIA INTEGRATION LANGUAGE (SMIL)

As mentioned earlier, multimedia systems have the ability to incorporate images, audio, video, animation, and charts with text and data. Multimedia information systems make it possible for users to store, retrieve, manage, and analyze all information projected through the use of multimedia. The latest trends seem to be in the areas of live video and audiostreaming on the Internet; "streamed" is defined as streaming media (such as video and/or audio) that is displayed as it is being sent. Recently, multimedia systems have

incorporated the Internet in the use of synchronized presentations and videostreaming. The Internet has opened up (through the use of multimedia) as an exciting adventure into cyberspace.

A. Synchronized Multimedia Integration Language with Streaming Video

Synchronized multimedia integration language is the synchronized presentation of multiple media. The World Wide Web Consortium (W3C) recommended SMIL as an integration of a set of independent multimedia objects into a synchronized multimedia presentation on June 1998. With language similar to HTML, SMIL includes streaming audio, streaming video, images, and text. Streaming multimedia have many advantages over non-streaming multimedia: (1) playback can be started very quickly and (2) storage space is not needed on the client. The Real Networks had made beta SMIL implementation (G2) available in July 1998. In the Real Player, SMIL is tailored to Real Audio, Real Video, Real Pix (graphics), Real Text, and Real Flash. These media types are integrated through SMIL and are streamed for playback in the Real Player. A person can put additional graphics on the screen with the video with SMIL. Using traditional video, a person can only view the video, whereas the SMIL file allows the placement of different images on different portions of the screen, such as a PowerPoint file. This allows the user to view both the video and the PowerPoint presentation at the same time.

Videostreaming technology provides a constant stream of video that is displayed in a window on the user's desktop. This differs from the file transfer protocol (FTP) method in that the video is viewed as it is sent rather than being downloaded to the user's PC and then viewed later. Streaming video effectively brings audio and video together into one affordable technology. Because of its affordability, companies are using this technology to make their web sites more interactive and, consequently, more appealing. Streaming media allows companies to better fulfill the needs of their employees, customers, and business partners. Some examples of the way companies are using video streamlining to promote and enhance e-commerce are: (1) live video broadcasting, (2) live multimedia conferencing, (3) live distance education and training, (4) live monitoring and security, and (5) live customer and sales support. Figure 6 illustrates presentation of streaming technology, the encoding process, Real Producer, and a sample of SMIL.

Figure 6 Streaming, encoding process, encoder, SMIL.

B. Videostreaming-Related Wares

Creating a streaming video can be done with a digital video camcorder, a PC, and software such as Real Producer or Microsoft Media Producer. The software captures the video from the source via Firewire connections, encodes the video in streaming format, and saves the file in a resolution with the specified bit rate (e.g., 12,000). There are several compression techniques being used in videostreaming today. Two of the most widely known are Real Networks' RealSystem (G2, 7.0, or 8.0) and Windows Media. Windows Media uses a format within MPEG 4 codec; this technology is dynamically scalable for varying bandwidths and can accurately estimate motion changes between frames. RealSystem (G2, 7.0, or 8.0) uses discrete cosine transform (DCT), known as a video compression standard. The DCT transforms data on the spatial arrangement of color in an image into data based on the frequency of occurrence of a given color arrangement. Once the media are ready for use, they can be uploaded to a media server such as RealServer or Microsoft Media Server. The advantages of RealServer are its versatility and compatibility with many operating systems including Unix (i.e., Linux) and Windows NT. Table IV shows a list of hardware (video capture card) and software (RealProducer and RealServer).

V. FUTURE OF MULTIMEDIA

The future of multimedia rests in the continued development of a technological infrastructure. Specifically, the slow server response times will continue to plague those businesses that rely on rich media types until increased bandwidth is available to a larger percentage of the population. Additionally, the future lies in four main areas: (1) computers and technology (e.g., faster and higher storage capacity), (2) telecommunications (e.g., cable modems, wireless telecommunications), (3) human interfaces (e.g., 3-D virtual reality, improved voice recognition), and (4) video (e.g., digital video, HDTV). It is not easy to predict what will occur in the rapidly changing world of high technology. However, cutting-edge multimedia technologies, such as "multimedia over Internet," "wire-

Table IV Video Capture Cards and RealProducer/Server

Hardware: Video Capture Card			
Producer	**Model**	**Features**	**System Requirements**
ViewCast	Osprey–100	NTSC and PAL format, up to 30 fps uncompressed, real-time encoding for live broadcast, no audio input	Win 9x, NT4.0 or Win2000, Pentium processor (200 Mhz and up), 16 MB RAM, PCI slot, sound card
ViewCast	Osprey–200	NTSC and PAL format, up to 30 fps uncompressed, real-time encoding for live broadcast, with audio input	Win 9x, NT4.0 or Win2000, Pentium processor (200 Mhz and up), 16 MB RAM, PCI slot, 16 bit depth graphics card supporting Direct Draw
Winnov	Videum VO PCI	NTSC and PAL format, up to 30 fps 640 × 480 pixels, TWAIN drivers, support all videoconferencing protocols	Win 3.x, Win 9x and NT4.0, Pentium processor (120 Mhz and up), 16 MB RAM, PCI slot, sound card, CD-ROM drive
Winnov	Videum AV PCI	NTSC and PAL format, up to 30 fps 640 × 480 pixels, TWAIN drivers, with audio input	Win 3.x, Win 9x and NT4.0, Pentium processor (120 Mhz and up), 16 MB RAM, PCI slot, CD-ROM drive
Software: RealProducer 7.0, RealServer 7.0			
Operating platform: RealProducer Plus (Windows 9x, NT, 2000, Macintosh, Linux, Compaq True 64/DEC Alpha) RealServer Plus (Windows NT Server, Linux, Solaris)			

less multimedia," "satellite multimedia communication," and "distributed multimedia," will continue to expand in the future.

SEE ALSO THE FOLLOWING ARTICLES

Computer-Aided Design • Desktop Publishing • Geographic Information Systems • Hyper-Media Databases • Internet, Overview • Java • Knowledge Management • Marketing • Virtual Reality

BIBLIOGRAPHY

Agnew, P. W., and Kellerman, A. S. (1996). *Distributed multimedia.* Reading, MA: Addison-Wesley.

Anderson, R. E., Sallis, P. J., and Yeap, W. K. (1991). Enhancing a hypertext application using NLP techniques. *Journal of Information Science Principles & Practice* **17**(1), 49–56.

Conklin, J. (1987). Hypertext: An introduction and survey. *IEEE Computer* **20**(9), 17–41.

Fisher, C. (1991). No amazing feet: Dropouts mar athletic-shoe expectations. *Advertising Age* **62**(6), 31–32.

Flynn, R. J., and Tetzlaff, W. H. (March 1998). Multimedia—An introduction. *IBM Journal of Research and Development,* 165–174.

Gray, S. (2000). Multimedia across the disciplines. *Syllabus* **13**(9), 14–18; 61.

Horn, R. E. (1989). *Mapping Hypertext.* Lexington, MA: The Lexington Institute.

Hoschka, P. (1997). Towards synchronized multimedia on the web. *World Wide Web Journal,* Vol. II, Issue 2. http://w3journal.com/6/s2.hschka.html.

Jones, B. (1992). Multimedia with IBM. *Management Services* **36**(3), 28.

Magrath, A. J. (1992). The death of advertising has been greatly exaggerated. *Sales & Marketing Management* **144**(2), 23–24.

Richman, M. (2000). 1394 Trends. *Electronics News* **46**(9), 48.

Riggs, B. (1999). Streaming: Media goes mainstream. *InformationWeek Online,* Dec. 6. http://informationweek.com/764/stream.htm.

Senna, J. (Feb. 11, 2000). "Streaming media for the enterprise. *InfoWorld* 63–66.

Shim, J. P. (2000). SMIL and videostreaming for teaching business telecommunications and e-commerce. *Decision Line* **31**(4).

Shim, J. P., and Chun, S. (1994). The strategic use of hypermedia and multimedia by the Fortune1000. *Multimedia Computing: Preparing for the 21st Century* (S. Reisman, ed.), pp. 449–471, Harrisburg, PA: Idea Group Publishing Co.

Shulman, R. E. (1992). Multimedia . . . A high-tech solution to the industry's training malaise. *Supermarket Business.* **47**(4), 23–24; 77.

Tolley, H. (2000). Using multimedia to teach human rights online. *Syllabus* **13**(10), 59.

Walsh, J. (1999). Web multimedia goes mainstream. *InfoWorld* **21**(15), 65.

Wilder, C. (1992). Multimedia training: For good sports. *Computer World* **26**(18), 35.

Multiplexing

Curt M. White
DePaul University

GLOSSARY

code division multiplexing A multiplexing technique in which unique digital codes, rather than separate radio frequencies or channels, are used to differentiate one user from another (also know as code division multiple access).

dense wavelength division multiplexing Used with fiber optic systems and involves the transfer of multiple streams of data over a single optical fiber using multicolored laser transmitters.

frequency division multiplexing Involves assigning nonoverlapping frequency ranges to different signals.

ISDN multiplexing The ability of ISDN to support multiple digital channels of information. Currently there are two basic forms of ISDN multiplexing: primary rate interface and basic rate interface.

lambda The wavelength of each different colored laser.

multiplexing Transmitting multiple signals on one medium at the same moment in time.

multiplexor The device that combines (multiplexes) multiple input signals for transmission over a single medium and then demultiplexes the composite signal back into multiple signals.

statistical time division multiplexing A form of time division multiplexing in which the multiplexor creates a data packet of only those devices that have something to transmit.

synchronous optical network A high speed time division multiplexed system that employs fiber optic transmission.

synchronous time division multiplexing A multiplexing technique that gives each incoming source a turn to transmit, proceeding through the sources in round-robin fashion.

time division multiplexing A multiplexing technique in which the sharing of a signal is accomplished by dividing available transmission time on a medium among users.

T-1 multiplexing A multiplexing technique that creates a T-1 output stream that is divided into 24 separate digitized voice/data channels of 64 kbps each.

I. INTRODUCTION

Under the simplest conditions, a medium can carry only one signal at any moment in time. For example, the cable that connects a keyboard to a personal computer carries a single digital signal at any moment in time. Likewise, the twisted pair wire that connects a personal computer to a local area network (LAN) carries only one digital signal at a time. Many times, however, we want a medium to carry multiple signals at the same time. When watching television, we want to receive multiple television channels in case we don't like the program on the channel we are currently watching. We have the same expectations of broadcast radio. When you walk or drive around town and see many people talking on cellular telephones, something allows this simultaneous transmission of multiple signals to happen. This technique of transmitting multiple signals over a single medium is *multiplexing*.

Multiplexing is a technique performed at the physical layer of the open systems interconnect (OSI) model or the interface layer of the Internet model.

For multiple signals to share one medium, the medium must somehow be divided, giving each signal a portion of the total bandwidth. Presently there are four basic ways to divide a medium: frequency division multiplexing, time division multiplexing, dense wavelength division multiplexing, and code division multiplexing.

Frequency division multiplexing involves assigning nonoverlapping frequency ranges to different signals. For example, cellular telephone systems that use advanced mobile phone services (AMPS) technology divide the available bandwidth into multiple channels. Thus, the telephone connection of one user is assigned one set of frequencies for transmission, while the telephone connection of a second user is assigned a second set of frequencies.

In *time division multiplexing,* the available transmission time on a medium is divided among users. As a simple example, suppose two users A and B wish to transmit data over a shared medium to a distant computer. We can create a rather simple time division multiplexing scheme by allowing User A to transmit during the first second, then user B during the following second, followed again by user A during the third second, and so on. *Statistical time division multiplexing* is a form of time division multiplexing in which the multiplexor creates a data packet of only those devices that have something to transmit.

Although frequency division and time division are the two most common multiplexing techniques, a third technique, dense wavelength division multiplexing, has emerged in the last couple years and promises to be a very powerful multiplexing technique. *Dense wavelength division multiplexing,* or simply wavelength division multiplexing, is used with fiber-optic systems and involves the transfer of multiple streams of data over a single optical fiber using multicolored laser transmitters. This technique is very similar to frequency division multiplexing since each color laser is essentially a different frequency signal.

Code division multiplexing is also a relatively new technology. Whereas other multiplexing techniques differentiate one user from another by either assigning frequency ranges or interleaving bit sequences in time, code division multiplexing allows multiple users to share a common set of frequencies by assigning unique digital codes to each user. While this technique has been used for many years by the military, it has only recently been used by the commercial sector. Let's examine each of these multiplexing techniques in detail.

II. FREQUENCY DIVISION MULTIPLEXING

Frequency division multiplexing, as stated earlier, is the assignment of nonoverlapping frequency ranges to each "user" of a medium. A user might be the cellular telephone you are talking on, or it could be a computer terminal sending data to a mainframe computer. A user can also be a television station that has been assigned a set of frequencies on which to transmit its television channel into homes and businesses. So that multiple users can share a single medium, each user is assigned a channel. A channel is an assigned set of frequencies that is used to transmit the user's signal. There are many examples of frequency division multiplexing in business and everyday life. One of the earliest computer-based examples of frequency division multiplexing was the connection of multiple computer terminals to a mainframe computer using a pair of multiplexors. In this particular example, the *multiplexor* is the device that accepts input from multiple users, converts the digital data streams to analog signals using assigned frequencies, and transmits the combined analog signals over a medium that is fast enough to support the total of all the assigned frequencies. A second multiplexor, or demultiplexor, is attached to the receiving end of the medium and splits off each signal, delivering it to the appropriate receiver. Figure 1 shows a simplified diagram of frequency division multiplexing.

Frequency division multiplexing is used with analog signaling. The multiplexor assigns a different range of frequencies to each source and transmits the multiple sources on analog signals using the assigned frequency ranges. To keep one signal from interfering with another signal, a set of unused frequencies called a guard band is usually inserted between the two signals to provide a form of insulation. The demultiplexor on the receiving end receives the combined set of signals and, using electronic filtering, separates the signals back into the individual sources. The medium that transfers the combined set of signals from sender to receiver must be capable of carrying a range of frequencies that can support the sum of all the individual frequency ranges. Most frequency division multiplexors that are used to connect computer terminals to a mainframe computer use a static assignment of frequencies. If a computer terminal has no data to transmit, a range of frequencies is still assigned to that input source. If multiple devices are not transmitting, the need to assign a range of frequencies to all input sources can lead to a waste of frequencies. This form of static frequency assignment for terminal connections is decreasing in use and is being replaced with time division multiplexing techniques.

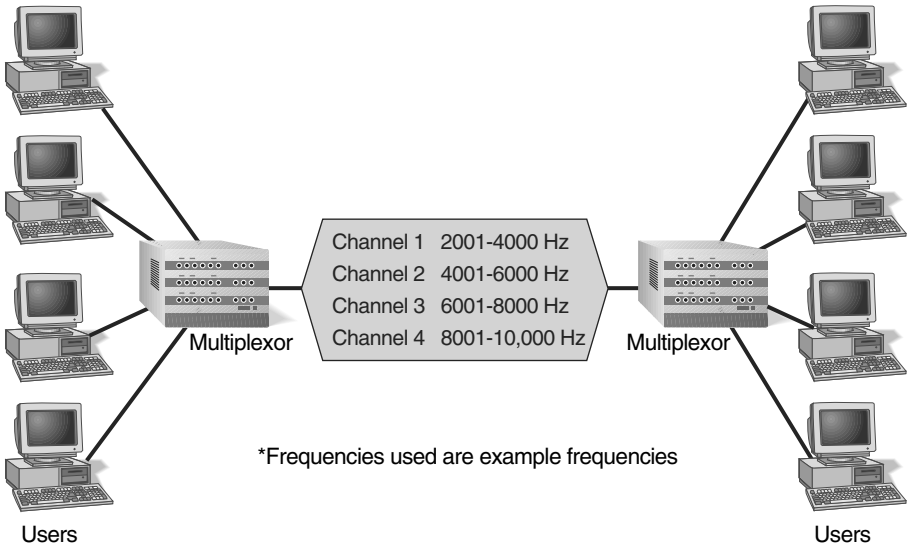


Figure 1 Simplified example of frequency division multiplexing. © Course Technology. Used with permission.

One of the most commonly found applications of frequency division multiplexing is cable television. Each cable television channel is assigned a unique range of frequencies, as shown in Table I. The television set, cable television box, or a video cassette recorder contains a tuner, or channel selector, which is the demultiplexor. The tuner separates one channel from the next and presents each as an individual data stream to you, the viewer. Note that each channel is assigned a range of frequencies by the Federal Communications Commission (FCC) and that the frequencies from one channel to another do not overlap. For example, channel 2's frequencies might end at approximately 60.2 MHz, while channel 3's frequencies might begin around 60.4 MHz.

A few companies still use frequency division multiplexing with broadband coaxial cable to deliver multiple audio and video channels to computer workstations. Videoconferencing is a common application in which two or more users transmit frequency multiplexed signals, often over long distances.

You can find another very common example of frequency division multiplexing in the cellular telephone system that employs AMPS technology. Cellular telephone systems allocate channels using frequency ranges within the 800–900 MHz spectrum. To be more precise, the 824–849 MHz range is used for receiving signals from cellular telephones (the uplink), while the 869–894 MHz range is used for transmitting to cellular telephones (the downlink). To carry on a two-way conversation, two channels must be assigned to each telephone connection. The signals coming into the mobile telephone come in on one 30-kHz band, while the signals leaving the mobile telephone go out on a different 30-kHz band. Cellular telephones are an example of dynamically assigned channels. When a user enters a telephone number and presses the send button, the cellular network assigns a range of frequencies based on current network availability. Dynamic assignment of frequencies is less wasteful than the static assigned frequencies found in terminal-to-computer multiplexed systems.

Frequency division multiplexing suffers from two major disadvantages. The first disadvantage is found in computer-based systems that multiplex multiple channels over a single medium. Since the frequencies are statically assigned, devices that do not have anything to transmit are still assigned frequencies and thus bandwidth is wasted.

The second disadvantage of frequency division multiplexing occurs because the technique uses analog signals exclusively, and analog signals are more disrupted by noise than digital signals. Nonetheless, many different types of applications use frequency division multiplexing, and it will more than likely be with us for a long period of time.

III. TIME DIVISION MULTIPLEXING

Signal division using frequency assignments is not terribly efficient, and frequency division multiplexing cannot be used with digital signaling techniques unless the digital signals are first converted to analog signals. In

Table I Assignment of Frequencies for Cable Television Channels

	Channel	Frequency in MHz
Low-band VHF and cable	2	54–60
	3	60–66
	4	66–72
	5	76–82
	6	82–88
Mid-band cable	95	90–96
	96	96–102
	97	102–108
	98	108–114
	99	114–120
	14	120–126
	15	126–132
	16	132–138
	17	138–144
	18	144–150
	19	150–156
	20	156–162
	21	162–168
	22	168–174
High-band VHF and cable	7	174–180
	8	180–186
	9	186–192
	10	192–198
	11	198–204
	12	204–210
	13	210–216

contrast, time division multiplexing (TDM) directly supports digital signals. In TDM, sharing of the signal is accomplished by dividing available transmission time on a medium among users.

How does time division multiplexing work? A simple example illustrates the technique. Suppose an instructor in a classroom poses a controversial question to students. In response, a number of hands shoot up and the instructor calls on each student, one at a time. It is the instructor's responsibility to maintain that only one student talks at any given moment so that each individual's conversation is understandable. In a relatively crude way the instructor is a TDM, giving each user (student) a moment in time to transmit data (express an opinion to the rest of the class). In the same way, a TDM calls on one input device after another, giving each device a turn at transmitting its data over a high-speed line. Since TDM was introduced in 1960s, it has split into two roughly parallel but separate technologies: *synchronous* and *statistical*.

A. Synchronous TDM

Synchronous TDM, the original technology, gives each incoming source a turn to transmit, proceeding through the sources in round-robin fashion. Given n inputs, a synchronous TDM accepts one piece of data, such as a byte, from the first device, transmits it over a high-speed link, accepts one byte from the second device, transmits it over the high-speed link, and continues this process until a byte is accepted from the nth device. After the nth device's first byte is transmitted, the multiplexor returns to the first device and continues in round-robin fashion. The resulting output stream from the multiplexor is demonstrated in Fig. 2.

Note that the demultiplexor on the receiving end of the high-speed link must disassemble the incoming bit stream and deliver each byte to the appropriate destination. Since the high-speed output data stream generated by the multiplexor does not contain addressing information for individual bits, a precise order must be maintained so that the demultiplexor can disassemble and deliver the bytes to the respective owners in the same sequence as what was input.

The synchronous TDM, under normal circumstances, maintains a simple round-robin sampling order of the input devices, as shown in Fig. 2. What would happen if one input device is sending data at a much faster rate than any of the others? You could provide an extensive buffer to hold the data from the faster device, but this buffer would provide only a temporary solution to the problem. A better solution is to sample the faster source multiple times during one round-robin pass. Figure 3 demonstrates how the input from device A is sampled twice for every one sample of the other input devices. As long as the demultiplexor understands this arrangement and this arrangement doesn't change dynamically, there should be no problems. This technique will only work, however, if the faster device is two, three, or four times—an integer multiple—faster than the other devices. If device A is two and a half times faster than the other devices, this technique will not work. Device A's input stream will have to be padded with additional "unusable" bits to make its input stream three times faster than the other devices.

What happens if a device has nothing to transmit? If a device has nothing to transmit, the multiplexor must still allocate a slot for that device in the high-speed output stream. In essence, that time slot is empty. Since each time slot is statically fixed, the multiplexor cannot take advantage of the empty slot and reassign busy devices to it. If, for example, only one

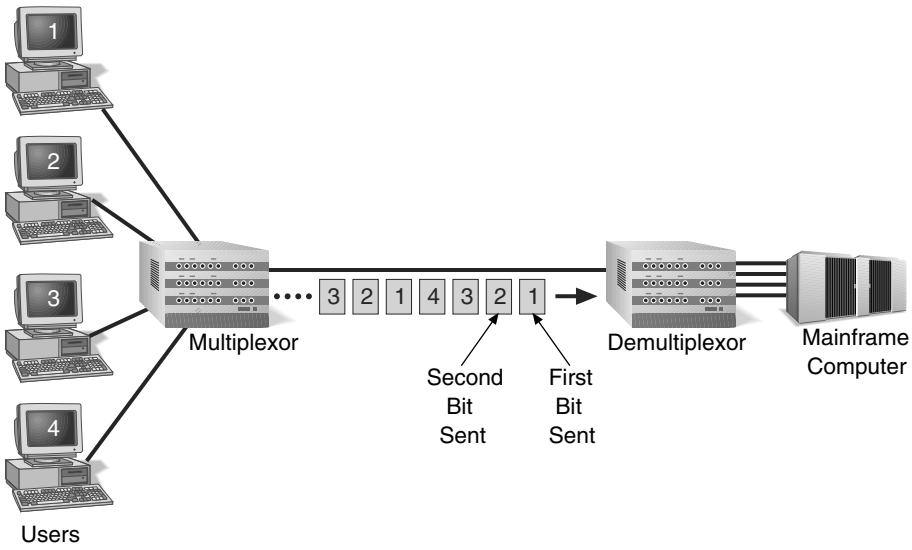


Figure 2 Sample output stream generated by a synchronous TDM. © Course Technology. Used with permission.

device is transmitting, the multiplexor still transmits a byte from each input device (Fig. 4). In addition, the high-speed link that connects the two multiplexors must always be capable of carrying the total of all possible incoming signals, even if all input sources are not transmitting data.

As with a simple connection between one sending device and one receiving device, maintaining synchronization across a multiplexed link is important. To maintain synchronization between sending multiplexor and receiving demultiplexor, the data from the input sources are often packed into a simple frame with synchronization bits added somewhere within the frame (Fig. 5). Depending on the TDM technology used, anywhere from one bit to several bits can be added to a frame to provide synchronization. The synchronization bits act in a fashion similar to Differential Manchester's constantly changing signal—they provide a constantly reappearing bit sequence onto which the receiver can anticipate and lock.

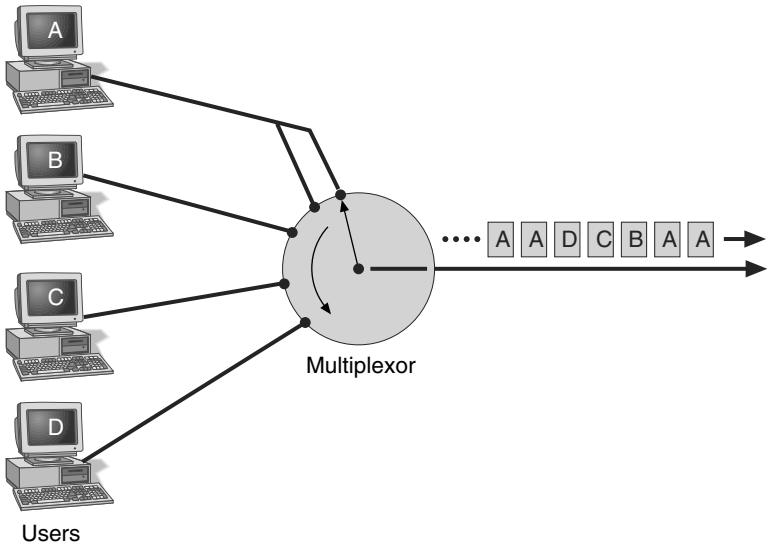


Figure 3 A synchronous TDM system that samples device A twice as fast as the other devices. © Course Technology. Used with permission.

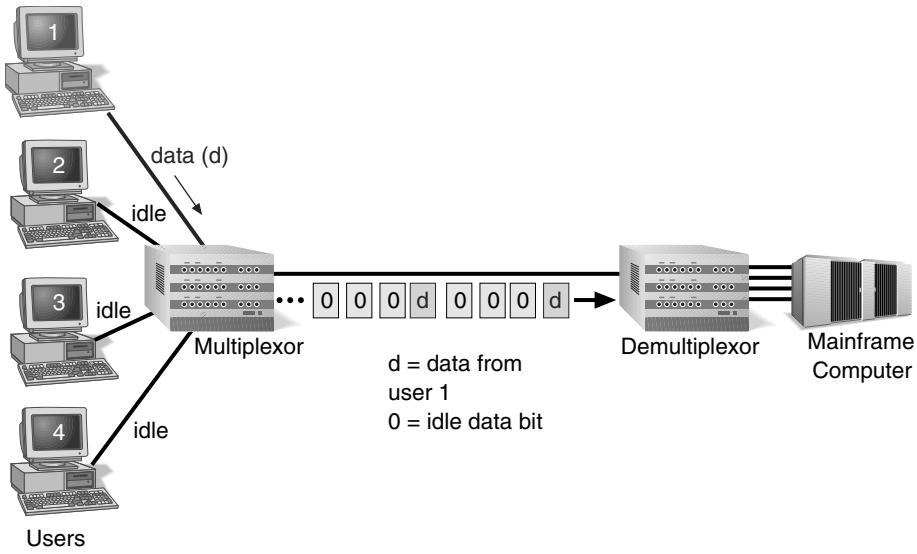


Figure 4 Multiplexor transmission stream with only one input device transmitting data. © Course Technology. Used with permission.

Three types of synchronous TDM that are popular today are T-1 multiplexing, ISDN multiplexing, and SONET. Although the details of T-1, ISDN, and SONET are very technical, a brief examination will show how multiple channels of information are multiplexed together into a single stream of data.

1. T-1 Multiplexing

In 1984, AT&T presented a service that multiplexed digital data and digitized voice onto a high-speed T-1 line with a data rate of 1.544 megabits per second. The T-1 line's original purpose was to provide a high-speed connection between AT&T's switching centers. When businesses learned of this high-speed service, they began to request the service to connect their computer and voice communication systems into the telephone network.

The T-1 multiplexor's output stream is divided into 24 separate digitized voice/data channels of 64 kbps each (Fig. 6). Users who wish to use all 24 channels are using a full T-1, while other users who need to use only part of the 24 channels may request a fractional T-1. The T-1 multiplexed stream is a continuous repetition of frames. Each frame consists of one byte from each of the 24 channels (users) plus one synchronization bit. Thus, data from the first user are followed by the data from the second user, and so on, until data from the 24th user are once again followed by data from the first user. If 1 of the 24 input sources has no data to transmit, the space within the frame is still allocated to that input source.

Although newer technologies such as frame relay and asynchronous transfer mode have grown in popularity, the T-1 system remains a classic application of synchronous TDM. The input data from a maximum of 24 devices are assigned to fixed intervals. Each device can only transmit during that fixed interval. If a device has no significant data to transmit, the time slot is still assigned to that device and data such as blanks or zeros are transmitted.

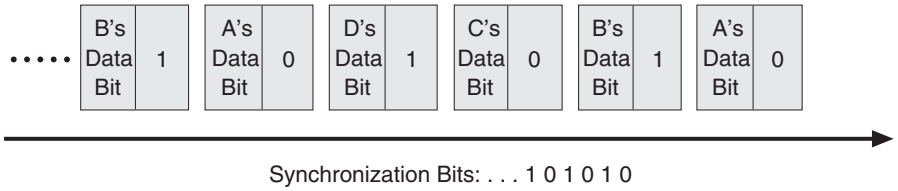


Figure 5 Transmitted frame with added synchronization bits. © Course Technology. Used with permission.

2. ISDN Multiplexing

Integrated Services Digital Network (ISDN) is a digital telephone service that provides voice and data transfer services over standard twisted pair into a home or small business. Although ISDN was designed to provide a number of services in addition to voice and data, data is the more popular use of most ISDN installations.

The ISDN service comes in two basic forms: primary rate interface (PRI) and basic rate interface (BRI). Primary rate interface was designed for business applications and, similar to T-1, multiplexes 24 input channels together onto one high-speed telephone line. Basic rate interface is the interface more often used by consumers to connect their home and small business computers to the Internet. The BRI multiplexes only three separate channels onto a single medium speed telephone line. Two of the three channels—the B channels—carry either data or voice, while the third channel—the D channel—carries the signaling information that controls the two data/voice channels. Since most consumers already have a standard telephone line, it is very common to use both of the B channels for data.

Figure 7 shows how the data from the two B channels (B1 and B2) plus the signaling information from the D channel are multiplexed together into a single frame. Note that 8 bits of data from the first B channel are followed by signaling control information, which is then followed by 8 bits of data from the second B channel and more signaling control information. These four groups of information repeat again to form a single frame.

3. SONET/SDH

Synchronous Optical Network (SONET) and synchronous digital hierarchy (SDH) are very powerful standards for multiplexing data streams over a single medium. SONET, developed in the United States by ANSI, and SDH, developed in Europe by ITU-T, are two almost identical standards for the high bandwidth transmission of a wide range of data types over fiber-optic cable. SONET and SDH have two features that are of particular interest in the context of multiplexing. First, they are both synchronous multiplexing techniques. A single clock controls the timing of all transmission and equipment across an entire SONET/SDH network. Using only a single clock to time all data transmissions yields a higher level of synchronization, since the system does not have to deal with two or more clocks having slightly different times. This high level of synchronization is necessary to achieve the high level of precision required when transmitting data at hundreds and thousands of megabits per second.

Second, SONET and SDH are able to multiplex varying speed streams of data onto one fiber connection. SONET defines a hierarchy of signaling levels, or data transmission rates, called synchronous transport signals (STS). Each STS level supports a particular data rate, as shown in Table II. Each of the STS signaling levels is supported by a physical specification called the optical carriers (OC). Note that the data rate of OC-3 is exactly 3 times the rate of OC-1; this relationship carries throughout the entire table of values. SONET is designed with this data rate relationship so that it is relatively straightforward to multiplex signals. For example, it is relatively simple to multiplex three STS-1 signals into one STS-3 signal. Likewise, four STS-12 signals can be multiplexed into one STS-48 signal. The STS multiplexor in a SONET network can accept electrical signals from copper-based media, convert those electrical signals into light pulses, and then multiplex the various sources onto one high-speed stream.

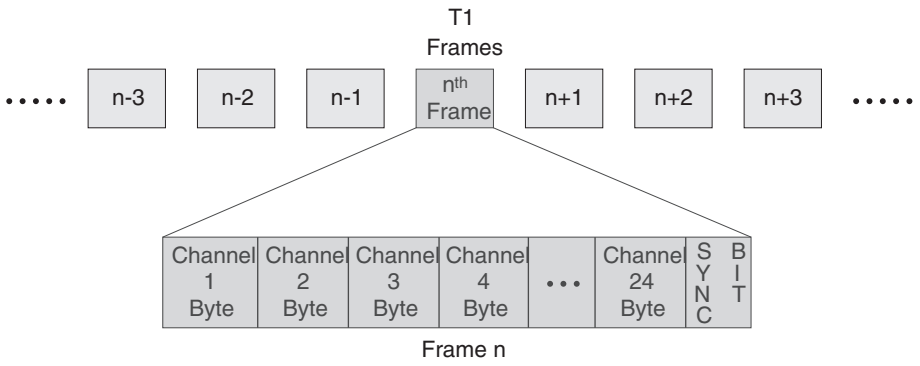


Figure 6 T-1 multiplexed data stream. © Course Technology. Used with permission.

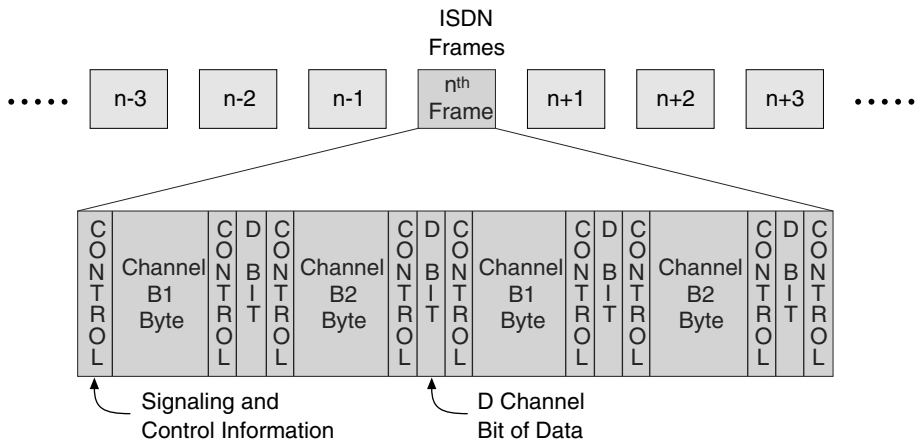


Figure 7 ISDN frame layout showing B channel bits and signaling control information bits. © Course Technology. Used with permission.

Each SONET frame contains the data that is being transmitted plus a number of control bits, which are scattered throughout the frame. Figure 8 shows the frame layout for the STS-1 signaling level. The STS-1 signaling level supports 8000 frames per second, and each frame contains 810 bytes (6480 bits) and 8000 frames per second times 6480 bits per frame yields 51,840,000 bits per second, which is the OC-1 data rate. The other STS signaling levels are similar except for the layout of data and the placement and quantity of control bits.

SONET and SDH are used in numerous applications in which very high transfer rates of data over fiber-optic lines are necessary. For example, two common users of SONET are the telephone company and companies that provide an Internet backbone service. Both telephone companies and Internet backbone providers have very high speed transmission lines, which span parts of the country. Since these transmission lines must transmit hundreds and thousands of millions of bits per second over long distances, fiber-optic lines supporting SONET transmission technology is a good solution.

Table II **STS Signaling Levels, Corresponding OC Levels, and Data Rates**

STS level	OC specification	Data rate (in Mbps)
STS-1	OC-1	51.84
STS-3	OC-3	155.52
STS-9	OC-9	466.56
STS-12	OC-12	622.08
STS-18	OC-18	933.12
STS-24	OC-24	1244.16
STS-36	OC-36	1866.23
STS-48	OC-48	2488.32
STS-96	OC-96	4976.64
STS-192	OC-192	9953.28

B. Statistical TDM

Both frequency division multiplexing and synchronous TDM can waste unused transmission space. One solution to this problem is statistical TDM. *Statistical TDM* (sometimes called asynchronous TDM) transmits data only from active users and does not transmit empty time slots. To transmit data only from active users, a more complex frame is created that contains data only from those input sources that have something to send. For example, consider the following simplified scenario: If four stations A, B, C, and D are connected to a statistical multiplexor but only stations A and C are currently transmitting, the statistical multiplexor transmits only the data from stations A and C, as shown in Fig. 9. Note that at any moment, the number of stations transmitting can change from two to zero, one, three, or four. If that happens, the statistical multiplexor creates a new frame containing data from the currently transmitting stations.

Since only two of the four stations are transmitting, how does the demultiplexor on the receiving end recognize the correct recipients of the data? Some type of address must be included with each byte of data to identify who sent the data and for whom it is intended (Fig. 10). The address can be as simple as a binary number that uniquely identifies the station that is

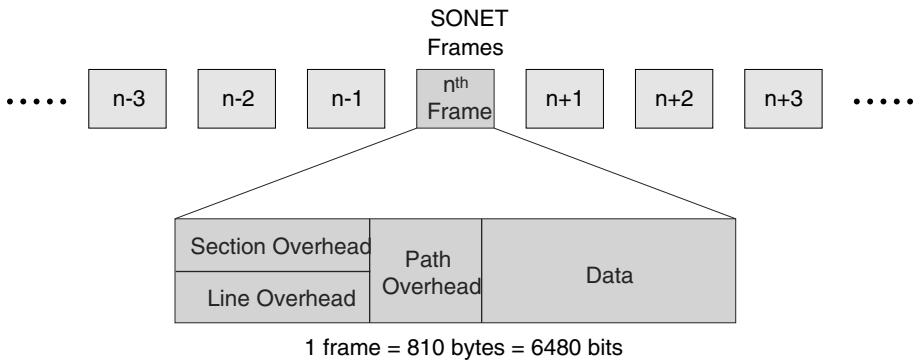


Figure 8 SONET STS-1 frame layout. © Course Technology. Used with permission.

transmitting. For example, if the multiplexor is connected to four stations, then the addresses can simply be 0, 1, 2, and 3 for stations A, B, C, and D. In binary, the values would be 00, 01, 10, and 11.

If the multiplexor transmits more than one byte of data at a time from each source, then an alternative form of address and data are required. To transmit a variable sized piece of data, a length field defining the length of the data block is included along with the address and data. This packet of address/length/data/address/length/data . . . is shown in Fig. 11.

Finally, the sequence of address/length/data/address/length/data . . . is packaged into a larger unit by the statistical multiplexor. This larger unit, shown in Fig. 12, is a more realistic example and looks much like the frame that is transmitted using a synchronous data link connection. The Flag at the beginning and end delimits the beginning and end of the frame. The Control field provides information that is used by the pair of multiplexors to control the flow of data between sending multiplexor and receiving multiplexor. Finally, the frame check sequence (FCS) provides information that the receiving multiplexor can use to detect transmission errors within the frame.

Statistical multiplexors have one very good advantage over synchronous TDMs. Although both synchronous TDM and statistical TDM can transmit data over a high-speed link, statistical TDM does not require as high speed a line as synchronous TDM does. Statistical TDM assumes that all devices do not transmit at the same time; therefore, it does not require a high-speed link that is the total of *all* the incoming data streams. Since it is assumed that only a percentage of input sources transmit at one time, the output

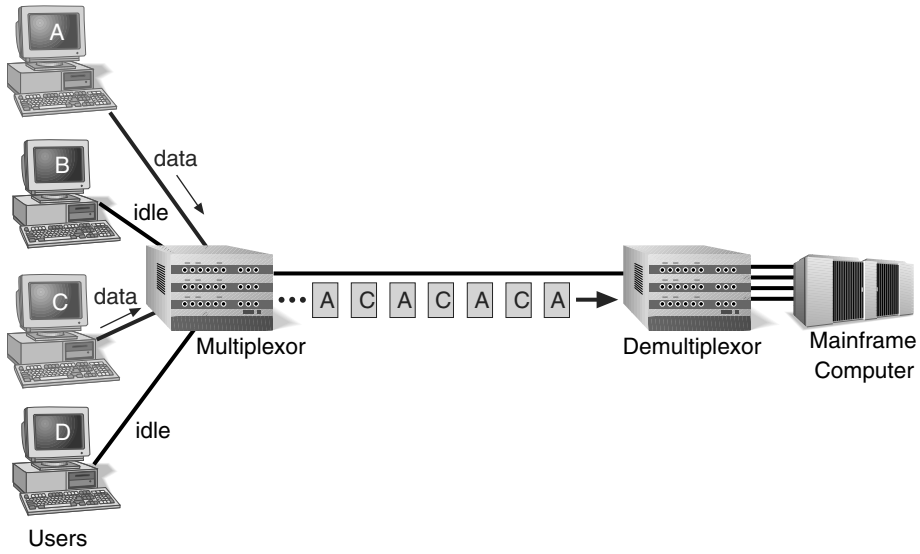


Figure 9 Two stations out of four transmitting via a statistical multiplexor. © Course Technology. Used with permission.

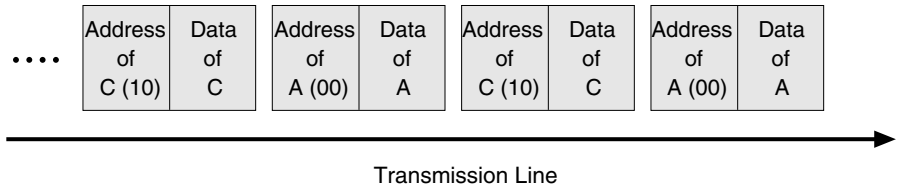


Figure 10 Sample address and data in a statistical multiplexor output stream. © Course Technology. Used with permission.

line capacity coming from the statistical multiplexor could be less than the output line capacity from the synchronous multiplexor, thus allowing for a slower speed link between multiplexors. This slower speed link usually translates into less cost.

One small disadvantage of statistical multiplexors is their increased level of complexity. Synchronous TDM simply accepts the data from each attached device and transmits that data in a nonending cycle. The statistical multiplexor must collect and buffer data from active attached devices and, after creating a frame with necessary control information, transmit that frame to the receiving multiplexor. Although this slightly higher level of complexity translates into higher initial costs, those costs are usually offset by the ability to use a smaller capacity interconnecting line.

Statistical TDM is also a good choice for connecting a number of lower speed devices that do not transmit data on a continuous basis to a remote computer system. Examples of these systems include data entry systems, point of sale systems, and many other commercial applications in which users enter data at computer terminals.

IV. DENSE WAVELENGTH DIVISION MULTIPLEXING

Dense wavelength division multiplexing (DWDM), or simply wave division multiplexing, is a fairly new technology that multiplexes multiple data streams onto a single fiber-optic line. Unlike TDM which divides input sources by time, DWDM is similar to frequency division multiplexing in which input sources are assigned different sets of frequencies. More precisely, DWDM uses multiple uniquely colored lasers (or frequencies of light) to transmit multiple signals. The wavelength of each different colored laser is called the *lambda*. Thus, DWDM supports multiple lambdas. The originating multiplexor combines the multiple optical signals of the input sources so that they can be amplified as a group and transported over a single fiber. Interestingly, because of the properties of the signals, light, and glass fiber, each signal carried on the fiber can be transmitted at a different rate from the other signals. This means that a single fiber-optic line can support simultaneous transmission speeds such as 51.84 Mbps, 155.52 Mbps, 622.08 Mbps, and 2.488 Gbps. These speeds are the speeds defined by OC-1, OC-3, OC-12, and OC-48—the optical carrier specifications for high-speed, fiber-optic lines. In addition, a single fiber-optic line can support different transmission formats such as SONET, asynchronous transfer mode (ATM), and numerous other data transmission formats, in various combinations.

Dense wavelength division multiplexing is scalable. As the demands on a system and its applications grow, it is possible to add additional wavelengths or lambdas onto the fiber, thus further multiplying the overall capacity of the original fiber-optic system. Current technology allows for as many as 64 lambdas per fiber, with the possibility for hundreds of lambdas within a few years. This power does not come without a price tag, however. Dense wavelength division multiplexing is an expensive way to transmit signals from multiple devices. Many technology experts predict, despite its cost, DWDM will continue to grow in popularity and may someday even replace technologies such as SONET and SDH.

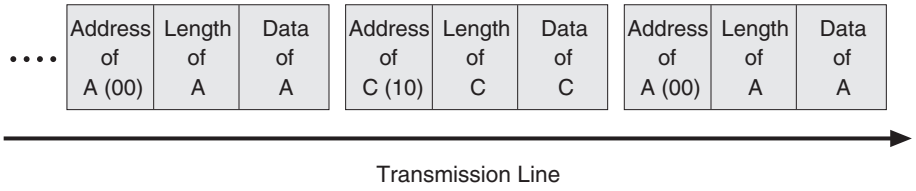


Figure 11 Packets of address and data fields in a statistical multiplexor output stream. © Course Technology. Used with permission.

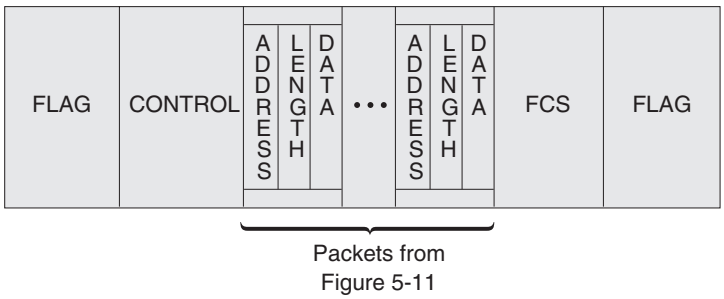


Figure 12 Frame layout for the information packet transferred between statistical multiplexors. © Course Technology. Used with permission.

V. CODE DIVISION MULTIPLEXING

Code division multiplexing (also known as code division multiple access) is a relatively new technology and has been used extensively by both the military and by cellular telephone companies. Whereas other multiplexing techniques differentiate one user from another by either assigning frequency ranges or interleaving bit sequences in time, code division multiplexing allows multiple users to share a common set of frequencies by assigning unique digital codes to each user.

More precisely, code division multiplexing is based upon spread spectrum technology. While spread spectrum systems fall into two categories—frequency hopping and direct sequence—code division multiplexing uses direct sequence spread spectrum technology. Direct sequence spread spectrum spreads the transmission of a signal over a wide range of frequencies using mathematical values. As the original data is input into a direct sequence modulator, it is mathematically combined with a pseudorandom bit stream. The result of the mathematical computation, which is now unique to the particular user, is transmitted to the intended receiver. When the data arrives at the intended receiver, the spread spectrum signal is again mathematically combined with the same pseudorandom bit stream that was used during the transmission of the signal. The result of this computation at the receiving end is the original data.

VI. COMPARISON OF MULTIPLEXING TECHNIQUES

The advantages and disadvantages of each multiplexing technique are summarized in Table III. Frequency

Table III Advantages and Disadvantages of Multiplexing Techniques

Multiplexing technique	Advantages	Disadvantages
Frequency division multiplexing	Simple Popular with radio, TV, cable TV Relatively inexpensive All the receivers, such as cellular telephones, do not need to be at the same location	Analog signals only Limited by frequency ranges
Synchronous TDM	Digital signals Relatively simple Commonly used with T-1 and ISDN	Wastes bandwidth
Statistical TDM	More efficient use of bandwidth Packets can be variable sized Frame can contain control and error information	More complex than synchronous TDM
DWDM	Very high capacities over fiber Scalable Signals can have varying speeds	Cost Complexity
Code division multiplexing	Large capacities Scalable	Complexity

division multiplexing relies on analog signaling and is the simplest and most noisy of all the multiplexing techniques. Synchronous time division multiplexing is also relatively straightforward, and like frequency division multiplexing, input devices that have nothing to transmit can waste transmission space. The big advantage of synchronous TDM is the lower noise during transmission. Statistical TDM is one step above synchronous TDM because it transmits data only from those input devices that have data to transmit. Thus, statistical TDM wastes less bandwidth on the transmission link. Dense wavelength division multiplexing is a very good, albeit expensive, technique for transmitting multiple concurrent signals over a fiber-optic line. Finally, code division multiplexing, while using a fairly wide bandwidth of frequencies, can produce system capacities that are eight to ten times that of frequency division multiplexing systems.

VII. SUMMARY

- For multiple signals to share a single medium, the medium must be divided into multiple channels.
- There are four basic techniques for dividing a medium into multiple channels: frequency division multiplexing, TDM, DWDM, and code division multiplexing.
- Frequency division multiplexing involves assigning nonoverlapping frequency ranges to different signals.
- Frequency division multiplexing uses analog signals while TDM uses digital signals.
- Time division multiplexing of a medium involves dividing the available transmission time on a medium among the users.
- Time division multiplexing has two basic forms: synchronous and statistical.
- Synchronous TDM accepts input from a fixed number of devices and transmits their data in a nonending repetitious pattern.
- T-1 and ISDN telephone systems are common examples of systems that use synchronous TDM.
- The static assignment of input devices to particular frequencies or time slots can be wasteful if the input devices are not transmitting data.
- Statistical TDM accepts input from a set of devices that have data to transmit, creates a frame with data and control information, and transmits that frame. Input devices that do not have data to send are not included in the frame.
- Dense wavelength division multiplexing involves fiber-optic systems and the transfer of multiple streams of data over a single fiber using multiple colored laser transmitters.
- Code division multiplexing employs direct sequence spread spectrum concepts and allows multiple users to share the same set of frequencies by assigning unique digital codes to each user.

SEE ALSO THE FOLLOWING ARTICLES

Integrated Services Digital Network (Broadband and Narrowband ISDN) • Network Environments, Managing • Standards and Protocols in Data Communications • Telecommunications Industry • Voice Communications

BIBLIOGRAPHY

Bates, R. J., and Gregory, D. W. (2000). *Voice & data communications handbook,* 3rd edition. New York: McGraw-Hill.

Flanagan, W. A. (1997). *T-1 networking: How to buy, install and use T-1 from desktop to DS-3,* 5th edition.

Forouzan, B. A. (2001). *Data communications and networking,* 2nd edition. New York: McGraw-Hill.

Shay, W. A. (1999). *Understanding data communications & networks,* 2nd edition. Boston: PWS.

Stallings, W. (2000). *Data and computer communications,* 6th edition. New Jersey: Prentice Hall.

National and Regional Economic Impacts of Silicon Valley

Daniel Felsenstein
Hebrew University of Jerusalem

GLOSSARY

agglomeration economies Economic advantages arising from the geographic concentration of production. These can be of two kinds: localization economies, which represent the economic benefits of being close to similar firms, and urbanization economies, which arise when disparate firms are clustered together in one large urban area.

business angels Wealthy individuals who invest small amounts of equity capital in start-up companies. Usually coming from a similar business background, they use their personal connections and experience in order to promote the company in its formative stage.

industrial clusters Concentrations of competing and interdependent firms that are grounded in physical proximity. Clusters create wealth in a region through exporting to other regions. Proximity is important for enabling the sharing of labor, suppliers, and networks and for reducing the costs of doing business.

local competitive advantage The mix of local conditions needed for achieving high and sustainable levels of productivity in a particular field. These include factor (input) conditions; the local context for competition such as the local tax system and local attitudes towards bankruptcy; demand conditions and the character of the local market; and local related and supporting industries (critical mass of local specialized suppliers).

local social capital A set of relationships, conventions, norms, and ties that exists within a given area and facilitates the exchange of ideas and experiences between individuals, firms, and institutions. In the case of Silicon Valley, the interactions between entrepreneurs, venture capitalists, universities, law firms, and government agencies give rise to a unique way of doing business grounded in performance, competition, and collaboration.

new economy Describes the notion of an economy pervaded and driven by information technology. This does not just imply an economy based on more high tech and R&D workers and firms. Rather it suggests that IT will affect all aspects of the economy: productivity, management, and organization.

venture capital Equity-based finance injected into new and high-risk firms. Venture capitalists are usually professional investors acting on behalf of clients. They often participate directly in the management of the firms they finance.

SILICON VALLEY is the popular name given to the strip of land roughly 70 miles long and 20 miles wide containing the world's largest concentration of high-technology industries. This area has captured worldwide attention due to its consistently high growth rates over a 40-year period. Countries and regions have sought to emulate the "Silicon Valley model" of economic growth, generally with limited success. The national and regional economic impacts generated by

Silicon Valley have become a source of inspiration and innovation in a variety of disparate fields ranging from technological development, through business finance, industrial organization, university–industry relations, work styles, and corporate culture. As such, "Silicon Valley" is more than just a geographic location or even a regional economy. Rather, it is a unique form of industrial organization that has evolved in a particular locale.

I. SILICON VALLEY AS AN ARCHETYPE

Few business locations have captured the public imagination to the extent of Silicon Valley. The attraction of the area transcends the fast economic growth and the allure of technological progress that characterize the region. At the cutting edge of scientific, economic, and social innovation, the influence of the valley lies in its inimitability. As an archetypal region, it represents a model of economic and social development that other regions strive to emulate, but never fully attain. Like a forbidden fruit, the visibility of the Silicon Valley model serves other budding locations as both a source of inspiration and as a constant reminder of what is, and is not, possible. This article outlines some of these parameters.

As an evolving and fast-growth region, the boundaries of Silicon Valley are constantly shifting outward. Historically, the concentration has come to be associated with the Santa Clara Valley that lies south of San Francisco. This area comprises an amalgam of communities and small towns with its epicenter lying between the towns of Palo Alto in the north and San Jose in the south. Between these two points, the valley incorporates other small localities such as Mountain View, Sunnyvale, Cupertino, and Santa Clara. In terms of administrative jurisdiction, the area has traditionally been associated with Santa Clara County. In recent years, however, the growth of the valley has seen the agglomeration spill over into adjacent counties such as Santa Cruz County, San Mateo County, and Alameda County. Today the area is thus bounded by the cities of Belmont to the north, Fremont in the east, and San Jose to the south (Fig. 1). While additional regional concentrations of high-tech activity exist at places such as San Francisco and Livermore, due to their noncontiguous nature they are not included here.

These geographic boundaries contain the largest single regional assembly of high technology activity and supporting institutions in the world. To appreciate the magnitude of this regional economy, note that Silicon Valley's gross product (GP) which measures the market value of all goods and services produced in the region, was estimated at over $100 billion in 1999. This is close to that of Portugal, which has six times the population, and greater than that of a string of national economies such as Malaysia and Venezuela, which have far larger populations. Fifty-eight percent of this GP is attributable to high-tech production, and no other singular metropolitan area in the United States comes close to this level of high-tech concentration in its regional economy.

In terms of regional productivity measured as value added per worker, the Silicon Valley average ($127,000) is more than twice that of the United States. The San Jose metropolitan statistical area (MSA), the closest statistical approximation of the Silicon Valley region, produced nearly 6% of U.S. high-tech real output in 1998, 11% of national high-tech manufacturing output, and San Jose computer firms contributed nearly 26% of the value of industry output in that sector. In all of these indices San Jose ranks first place nationwide and ahead of the metropolitan areas of Los Angeles, New York, Boston, Chicago, and Dallas. Other indicators of high-technology predominance, such as relative concentration of high-tech employment, output, and Internet presence, all put the San Jose metropolitan area ahead of other high-tech concentrations in the United States (Table I). The regional economy is also highly linked into global markets as evidenced by volume of export sales. In 1998, Silicon Valley exports were over $26 billion, putting the metropolitan area in fourth place nationwide behind Seattle, Detroit, and New York. A composite index combining locational concentration and share of national high-tech output gives the San Jose metro area a score 3.3 times higher that its nearest rival area (Dallas). This illustrates the magnitude and national dominance of the agglomeration of high-tech activity in the Silicon Valley area.

Silicon Valley houses close to 7000 high-technology companies. The main activities are clustered in the subfields of semiconductors, computers and communications, bioscience, defense/aerospace, software, manufacturing support services (such as computer maintenance, engineering, and research services), and business services (including corporate finance, legal services, management and public relations services and the like). These subsectors display their own microgeography within Silicon Valley. Thus, semiconductor companies are clustered in Santa Clara; Internet and software firms are in Mountain View, Palo Alto, and Sunnyvale; computer manufacturing facilities in San Jose and Milpitas; law firms in Palo Alto; venture capital companies in Menlo Park; and accountants in San Jose.

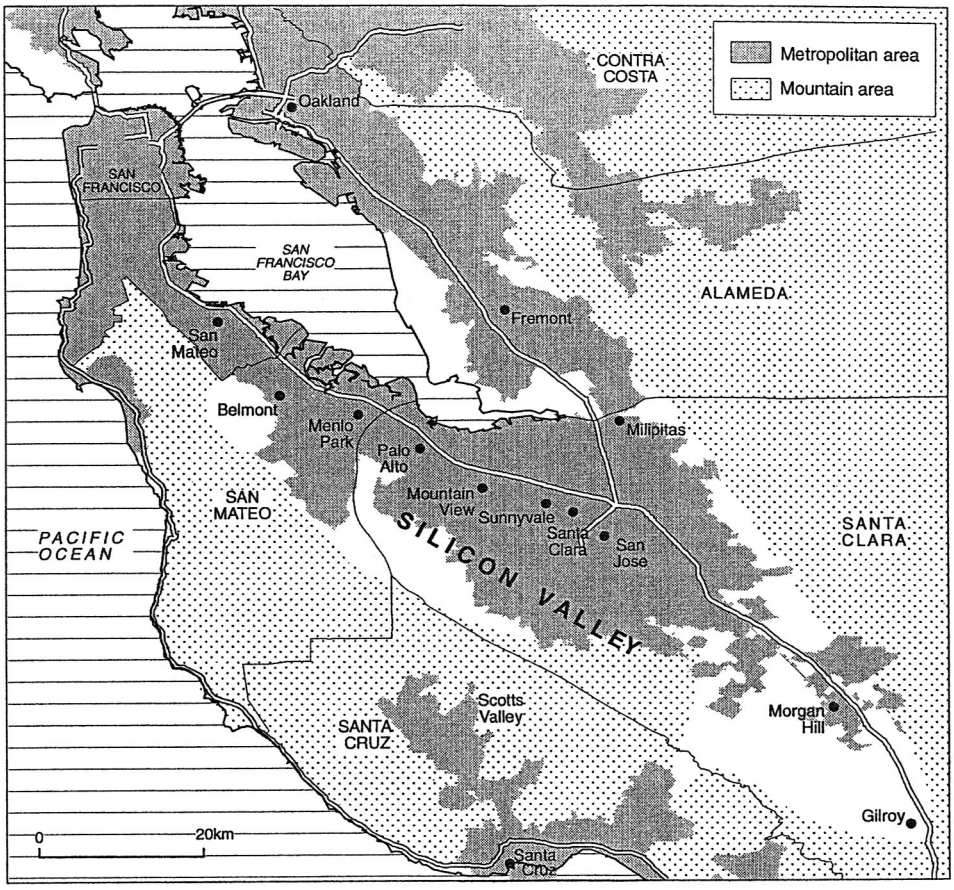


Figure 1 The Silicon Valley region.

Silicon Valley is home to roughly one-third of the world's leading electronics and software companies. Industry leaders such as Sun Microsystems, Cisco Systems, Silicon Graphics, Netscape, Applied Materials, Intel, 3Com, Oracle, and HP are intimately associated with the valley and its development. In total, employment in high-tech firms and allied services accounts for nearly 1.5 million jobs. Average wages in Silicon Valley firms ($66,400 in 2000) are nearly double the national average and real per capita incomes (which measure wealth creation) are 65% higher than the U.S. average.

These performance measures are further augmented by indices of regional innovation. In terms of gross number of patents issued and patents per capita, the San Jose MSA is ranked first in the nation, ahead of San Francisco, Boston, Chicago, and New York. A further indicator of regional dynamism relates to the volume of capital raised in the region. The estimated value of high-tech transactions for Silicon Valley is $18.6 billion for the year 2000, raised via nearly 1000 deals. This represents more than double the volume of capital and 30% more transactions than those conducted in the New York metro region, the second-ranked region.

In terms of Internet presence and connectivity, Silicon Valley is also a major center. While the San Jose metropolitan area is ranked seventh in the United States in terms of gross number of Internet domains (locations of Internet addresses), when the concentration of Internet addresses relative to volume of business activity is considered, then Silicon Valley again tops the list of U.S. metropolitan areas (Table I). This index, along with those of volume of high-tech production and innovative capacity, simply serves to reiterate the prominence of Silicon Valley as the world's major high-technology concentration.

The term *Silicon Valley,* coined by a local journalist writing a feature on the electronics industry for a Santa Clara trade newspaper in the early 1970s, has become synonymous with the physical concentration of high technology. Akin to Manchester of the Industrial Revolution and Detroit in the heyday of the automobile, Silicon Valley is the place most readily identified with

Table I Indicators for Silicon Valley and Other High-Tech Agglomerations

High-tech agglomeration[a]	High-tech employment[b] (thousands)	High-tech emp. as percent of labor force	High-tech output[c] ($ billions)	High-tech output as percent of gross metrop. product[c]	Percent U.S. domain names[d]	Domain name concentration[d]
San Jose	279	41	43.5	57.8	2.2	3.71
Los Angeles	402	9	43.3	13.1	7.3	2.21
Boston	329	21	44.4	20.6	2.5	1.77
Seattle	216	21	24.3	23.7	1.9	1.79
Washington, D.C.	265	20	39.2	20.2	3.8	2.06
Chicago	248	12	34.6	11.4	3.3	1.25
New York City	253	10	22.1	5.6	6.2	1.96
Atlanta	155	10	14.7	10.0	2.0	1.3
Dallas	210	16	27.4	19.1	—	—

[a]Based on MSA.
[b]*Source:* DeVol, R. C. (1999). *America's high tech economy.* Los Angeles: Milken Institute.
[c]*Source:* United States Conference of Mayors (2000). *U.S. metro economics, leading America's new economy.* Lexington, MA: McGraw-Hill.
[d]*Source:* Zook, M. (2000). Domain names in the U.S. Available at http://garnet:berkeley.edu/~zook/domain_names.

the microelectronics revolution, the development of the PC, and, more recently, the Internet. However, *Silicon Valley* conveys more than just a location for high-tech activity. Increasingly, the term has become associated with a way of doing business, a particular life and workstyle and even an attitude. Thus, while the considerable national and regional economic impacts of Silicon Valley should not be understated, the role of the location in social innovation should also not be ignored. Perhaps the single most alluring feature of Silicon Valley lies not in the story of fast economic growth and the replicability of this model of economic development. Rather, it lies in the ability to capture public imagination worldwide, offering a glimpse into the future and an opportunity to view a scenario of things to come from the perspective of the present.

II. THE FOUNDATIONS OF THE VALLEY

The Silicon Valley story is comprised of many elements: far-sighted individuals, key institutions, historical conditions, and chance occurrences. At different periods in the development of the Valley these elements have assumed varying levels of importance. In contrast to popular belief, the historical roots of the area go back almost 100 years with the formation of the Federal Telegraph Corporation (FTC) in Palo Alto in 1909. The vacuum tube was developed at the FTC laboratory by Lee de Forest in 1912 and led to a succession of experiments and innovations in the field of radio, television and military electronics. Interestingly enough, some key features of Silicon Valley development after World War II (local venture capital, university–industry collaboration, spin-off based growth, etc.) were already evident in the prototype model of Silicon Valley in the early 1900's. FTC was started by a Stanford graduate, Cyril Elwell, with the financial backing of the university president, David Starr Jordan. FTC subsequently spawned spin-offs such as Magnavox, Litton Industries, and Fisher Research Laboratories. These events also dovetailed with the development of the electronics industry in the San Francisco Bay Area in the 1920s, resulting in innovative breakthroughs such as the development of electronic transmissions of television images and shortwave radio systems for aircraft and airborne radio antenna. Thus prior to World War II a critical mass of electronics activity already existed in the wider region feeding off a climate of technological innovation and entrepreneurship. The transformation of the Santa Clara Valley from an agricultural area to a highly urbanized environment was not achieved overnight, nor can it be ascribed to unicausal explanations. Rather it was forged by the convergence of particular historical and geographic conditions that created the infrastructure on which the Silicon Valley model developed.

On this platform, the ingredients of modern-day Silicon Valley began to come together. The partner-

ship between William R. Hewlett and David Packard, who in 1938 began to develop an audio oscillator as a literal "garage operation," is often credited as the birth of Silicon Valley. As Stanford graduates they were animated, encouraged, and even financed by Fredrick Terman, then a professor of electrical engineering at the university. Using a Terman-inspired idea they produced and sold their first resistance-tuned oscillators to Walt Disney Studios, heralding a relationship between Silicon Valley and the Los Angeles entertainment industry that endures to this day. Subsequently, Hewlett-Packard developed into an electronics giant engaged in fields as diverse as electronic measuring devices, military products, computers, and peripherals. However, one of its more enduring contributions to Silicon Valley has been in the area of social innovation. The "HP way," which describes the benign mix of informal management with decentralized corporate structure and an emphasis on collaboration, responsibility sharing and entrepreneurship, has permeated much of the business culture of the Valley today.

Fredrick Terman, who has been dubbed the "Father of Silicon Valley," was an inspirational figure whose footprint can be discerned at critical junctures in the development of the valley in the post–World War II period. He spent the war years working on radar-jamming technology at Harvard and returned to Stanford in 1946 as dean of engineering. Wartime demand and defense contracts had generated vast new opportunities and knowledge in electronics and microwave technology, and Terman was determined to harness some of this accumulation of expertise in order to develop engineering at Stanford. In 1948, faculty members and research associates founded Varian Associates, which produced klystron and microwave tubes. This endeavor was literally incubated on the Stanford campus. Varian grew rapidly into a military electronics firms fueled by demand from the Korean War, and in 1951 became the first company to sign a lease for space in the Stanford Industrial Park.

This latter project was again a Terman initiative. He persuaded the university authorities to lease land on favorable terms for industrial use with an orientation to electronics and research. Aside from Varian Associates, early tenants included General Electric, Eastman Kodak, and Hewlett-Packard. By 1960 there were 25 companies at the (now renamed) Stanford Research Park. This number reached 70 in 1970 and grew to more than 100 firms in 2000 employing 28,000 employees on 900 acres of land adjacent to the university. In Terman's view, the Stanford Research Park furthered a series of goals simultaneously. It was the critical link in fostering the university–industry interaction in which he believed fervently. As a land-endowed university, leasing space at the park would generate revenues and enable the university to further attract promising faculty. The physical proximity between the park and the university would facilitate the creation of a "community of technical scholars" in which constant interchange between teacher, researcher, student, and product developer would be possible.

While the Stanford Research Park provided the nucleus around which the electronics industry was to agglomerate, a major technological breakthrough that would infuse the industry was still lacking. This appeared in1955 when William Shockley, co-inventor of the transistor at Bell Laboratories in New Jersey, decided to create a company in Mountain View in order to commercialize his research. As things transpired, this move by Shockley represented a significant technological import to the Valley. His company, Shockley Semiconductors Laboratory, started to work on developing double diffused silicon transistors. However. Shockley's personal temperament and obtrusive management style caused internal frictions within the company and eight of his initial star recruits left the company within a year. The "traitorous eight" as he bitterly called them started their own firm in order to promote volume production of silicon transistors. Leveraging capital from an East Coast camera company, they formed Fairchild Semiconductors in Palo Alto in 1957. Within a few years Fairchild had refined the integral process needed for the commercial production of integrated circuits. However, because of the opportunities that this emerging semiconductor industry harbored, Fairchild was finding it increasingly difficult to retain key employees. By 1965, 10 new semiconductor firms had been started by former Fairchild engineers. A decade later, about half of the leading 85 semiconductor companies in the United States could trace their roots to Fairchild. These included leading firms that defined the industry's products, processes, and manufacturing systems such as Intel, LSI Logic, National Semiconductor, and Advanced Micro Devices. Eventually the company that was spawned by the spin-off process and was so instrumental in its diffusion throughout the Silicon Valley economy fell victim to that same process.

Aside from the spin-off process, the presence of corporate research facilities in the San Jose area also did much to cement the development of Silicon Valley. Foremost among these was the IBM research laboratory established in 1952. This R&D center played a significant role in the development of magnetic data

storage and hard disk drives. In 1970, Xerox established the Palo Alto Research Center (PARC), which made a major contribution to seeding the technologies that promoted the networking advances of the 1990s such as graphic user interfaces, workstations, and Ethernet communications.

The Fairchild legacy was significant in the development of the Silicon Valley in a number of ways. First, technological innovation in semiconductor production called for a whole host of firms dedicated toward producing the specialized equipment and materials needed to actually produce semiconductors. The clustering of this allied industry in Silicon Valley further perpetuated the lead that the region had opened up through its technological dominance. Second, the financing demands of the semiconductor industry led to the development of the local venture capital industry. This was a cumulative process because many founders of Fairchild spin-offs had made huge capital gains that were further reinvested in venture capital firms or other spin-offs. A cyclical growth dynamic was thus set in motion in which wealthy local individuals could invest in new local firms without compromising their own personal financial status, thereby encouraging further rounds of new firm formation, investment, and capital gain. With all this occurring within a defined geographic area, the Silicon Valley model of regional development became a source of admiration and imitation the world over.

Note that defense contracts and programs underwrote much of the intense spin-off activity of the 1960s. NASA programs and aerospace projects provided much support for many of the riskier Silicon Valley cutting-edge technologies and thereby helped consolidate the process of agglomeration that was well under way. However, with the exception of Hewlett-Packard, many of the military electronics companies eschewed the civilian electronics market so that when the next wave of technological innovation hit Silicon Valley, they were ill-positioned to take full advantage. In 1974, the first PC was introduced by a small calculator company in Albuquerque, New Mexico. The effect was to immediately galvanize the many electronics hobbyists in the San Francisco Bay Area into producing rival products. For some of these young engineers and designers the motive for designing a personal computer was purely commercial. For others, business was mixed with visionary zeal; the PC was seen as an instrument of liberation, a source of personal power standing in stark contrast to the impersonal IBM mainframe-dominated world.

More than any other firm, Apple Computer epitomized this era in Silicon Valley development. The pioneering combination of Steve Jobs, the business visionary and Steve Wozniak, the technical expert, symbolized the individual and personal counterculture that permeated the Valley at this time. Cutting-edge lifestyles, thinking, and technology converged on Silicon Valley and became embodied in the PC and the hopes attached to it. Jobs and Wozniak were heralded as prophets of the future and were responsible for democratizing the use of computers and introducing them and their applications to the man in the street. While Apple did not survive the Silicon Valley (and Wall Street) downturn of 1987 and suffered competition from both IBM and IBM-type clones, its legacy for both Silicon Valley and the computer industry was considerable. It helped further entrench the electronics growth cluster and local competitive advantage in the Valley. In its wake other PC companies were spawned that both extended the stress on human-factor orientation and widened the computer platform in Silicon Valley to include the workstation market (Apollo, Sun Microsystems). While PC production quickly diffused out of Silicon Valley to Texas (Compaq, Dell) and South Dakota (Gateway) and offshore locations such as Taiwan and Malaysia, much of the value added remained in the region as Silicon Valley firms dominated niche markets in PC components. Intel and AMD provided microprocessors; Phoenix Technologies and AMI made BIOS chips; Seagate, Quantum and Conner Peripherals manufactured hard drives; and Cirrus Logic provided graphic chips. In addition, Apple was responsible for changing the composition of many of the professional jobs in the Valley's computer industry. Prior to Apple, engineers and technologists dominated computer manufacturing. Apple introduced the need for a more consumer-led approach. This necessitated the incorporation of MBAs and advertising experts as part of the high-technology production team and signaled the entry of the valley and its products into the world of mass consumerism.

The 1990s marked a further round of Silicon Valley renewal. The regional economy extricated itself from the combination of fierce competition from the Japanese and Southeast Asia computer and semiconductor manufacturers and Wall Street losses in the second half of the 1980s by riding the wave of consumer-oriented technology that had powered the PC revolution. The main instrument for this transformation was the Internet and also the ability of computer companies to reinvent themselves as communications and networking operations. Silicon Valley firms again took the lead in both these directions with Netscape opening up the browser market and firms like Cisco and 3Com leading the communications market. The impact of the Internet on the Silicon Valley regional

economy has been twofold. First, the expectations associated with the Internet have led to an enormous increase in both the amount of capital invested in Silicon Valley firms and the type of risk they present. An estimated 1000 high-tech transactions valued at $18.6 billion were conducted in Silicon Valley in 2000. This includes venture capital fund investment, stock market activity via initial public offerings (IPOs), mergers and acquisitions (M&A), and so on. The type of risk has also changed in that technological risk has now been superceded by market and sales risk. While the technological proficiency of the Internet hardly poses any risk, the main locus of uncertainty to Internet investors now lies in the market and sales prospects of the many products and services promoted by the Internet. The stock market shakeout of 2000–2001 certainly underscored this new source of risk.

Second, the development of the Internet has pulled Silicon Valley into the realm of national politics. Traditionally Silicon Valley interest in politics was limited to standard contributions to party campaigns and mobilization of Valley interests around issues affecting the economic future of the region such as local school standards and entry visas for foreign specialist workers. Recently, however, the Internet has been having a more sublime effect, challenging the role of government and its intervention in the economy. For example, the Internet calls into question governments' ability to tax and regulate, moving business activity into the ephemeral realm of cyberspace. The Internet also affords greater information to the individual who is consequently less dependent on government and its offices. Finally, the Internet also undermines traditional ideas about community. While government attempts to promote the virtues of community, the Internet illustrates that virtual communities of common interest can tie disparate people together as closely as communities of geographic proximity. As champions of the Internet, Silicon Valley representatives are being pulled into debates on societal wider issues of governance and community.

From the Silicon Valley story a few salient points emerge. The first is that the development process of Santa Clara Valley evolved over a period of nearly 100 years. There was little immediate jump-starting of the regional economy, despite the popular conception. Rather the process was one of constantly changing economic growth momentum. Second, many of the initial competences needed to set the growth process in motion were imported from the East Coast of the United States. Many of the faculty in the technical departments at Stanford were recruited from the East, much of the early technology was developed at places remote from the Silicon Valley area, initial capital sources came from the East Coast, and even some of the early new firm formations were transfers from the other side of the continent. Finally, the resilience of Silicon Valley needs to be noted. The region has managed to weather the boom and slump cycles that are a hallmark of technological development that constantly reinvents itself and changes its main product base over time: from a defense platform in the 1950s and 1960s (HP, Varian Associates) through integrated circuits (Intel, AMD, and National Semiconductors) and PCs (Apple, Sun) in the 1970s and 1980s through to the networking and Internet waves of the 1990s and 2000s (Netscape, Cisco Systems, 3Com, Yahoo, Bay Networks).

III. PROCESSES AT WORK

In trying to understand the rapid development of the area, observers of Silicon Valley have presented different perspectives on the dynamics of this growth. While the ingredients of the story are very similar (constant technological innovation, risk-taking entrepreneurs, availability of capital, university–industry interaction, physical agglomeration and so on), the emphases placed by the different explanations vary greatly. The basic platform underlying the different accounts is that of geographic proximity. All versions of the Silicon Valley story are predicated on the fact that the density and magnitude of engineers, financiers, specialized business services, scientific infrastructure, productive capacity and technological know-how within the confines of a small geographic area acted as a prerequisite for rapid economic growth. However, despite the common geographic framework, accounts diverge greatly emphasizing very different aspects of growth. We can suggest three main schools of explanation.

A. The Economic Growth Explanation

This sees the spatial agglomeration in Silicon Valley as a manifestation of increasing returns to scale and agglomeration economies. The physical mass of ideas, capital, and skills creates positive spillovers that are most readily available to those within the agglomeration. Some of these spillovers are in the form of formal codified knowledge (patents, production rights, etc) while others are available through less formal channels (learning, chance exchanges, copying, etc.) As these spillovers are released randomly in both space and time, physical concentration is necessary in order to ensure maximum exposure to the wealth of ideas

and know-how circulating the Valley. With growth, the dense network of suppliers and the vast labor pool that accumulates generate a further cycle of self-entrenching growth. While the many individual Silicon Valley firms are small scale, taken together the whole they create is clearly greater than the sum of its parts. The scale economies that arise are therefore external to the firm.

Others see the mass of activity located in the Silicon Valley as a result of scale economies that are internal to the firm. The interaction between fixed production costs and transport costs has caused Silicon Valley firms to cluster together close to a large market. In this way they can keep transport costs down and offer a wide variety of products, services, and specializations from a single location. Simultaneously, there are, of course, economic forces working against further agglomeration. Congestion, rising land prices, labor costs, and environmental problems all make for deconcentration away from Silicon Valley. However, if the forces for agglomeration are stronger than the centrifugal forces of dispersion, concentration will continue in a self-generating fashion.

This view of Silicon Valley development explains the generic economic forces at work. It does not offer an explanation of why all this has coalesced in the Santa Clara Valley in such a powerful form. To complement this explanation, Silicon Valley analysts mobilize the idea of historical accident in determining the future growth patterns of the Valley. Seen in this light, the individual chance decisions of Starr, FTA, Shockley, Terman et al. were a series of fortuitous choices that crystallized together in Silicon Valley. If another region had been fortunate in having such a combination of accidents and luck, there is no real reason why it would not have developed in a similar fashion. However, once Silicon Valley took off and opened up a lead against other competing regions, the forces of increasing returns and agglomeration economies ensured that this growth was cumulative, thereby locking in the gap between Silicon Valley and other locations who were effectively locked out.

B. The Regional Culture Explanation

This sees Silicon Valley as an industrial system made up of local interfirm relationships, local institutions, and a distinct intrafirm organizational culture. In this view, the main dynamos of Silicon Valley are the myriad small and medium sized firms. They operate on the basis of flexible, constantly evolving interfirm relations alternating between competition and collaboration—all in a highly localized context. Together these ingredients of cooperation and innovation in a given place combine to produce fast-growth regions. The glue sticking these pieces together is local "social capital"—a much vaunted but rather vague idea that conveys a particular way of doing business based on trust, cooperation, reciprocity, and social networks. While the mobility of the top-level workers in Silicon Valley is famous and may be considered detrimental to fostering trust and cooperation, the regional culture account sees this movement as a virtue. With key workers flitting from one company to another, career paths often overlap. Networks of information and experiences are formed on this personal basis and together contribute to regional innovative capacity.

This particular perception tends to ignore the competitive and sometimes cutthroat nature of economic development in the valley. While there may be a regional culture, it is often animated by the desire to realize innovative and commercial opportunities and reap fast market returns. Although networks might exist, they are instruments for creating commercial advantage. This is a far cry from the cozy picture of information sharing around the bar. In fact, many Silicon Valley firms get to meet their neighbors through the mediation of attorneys or in the law courts. Where trust exists it is commercially oriented and geared toward performance. Even the physical layout of the business and residential environment of Santa Clara County mitigates the chances of a regional culture grounded in civic-type social capital. Low-density suburban residences punctuated by shopping centers and malls, communities with rapid population turnover that are suffering from traffic congestion hardly provide the necessary infrastructure for the development of intense social links and civic engagement.

In fact, regional culture could be considered an effect or outcome of regional economic growth rather than a cause. The desire for increasing commercial performance leads to the emergence of new forms of trust. The imperative of capital gains animates all the forces in Silicon Valley (firms, investors, engineers, and so on). Each may harness its own sources of social capital toward this end. Social capital is thus used to enhance performance rather than the other way round.

C. The Corporate Power Explanation

In contrast to the two previous accounts, the corporate power explanation challenges both the view of Silicon Valley as a product of inexorable economic forces and the perspective that sees the valley as a

repository of social capital. This story sees the large corporations and the defense establishment as the key agents in understanding Silicon Valley growth. The small firm sector is not seen as an independent symbol of Silicon Valley vitality but rather as a sector intimately and ultimately linked to the large corporations operating in the valley.

Despite defense spending cuts, this view sees the role of military industries as vital to understanding both the historic and contemporary success of the Santa Clara Valley. While historically federal contracts were important in the development of the semiconductor industry, some observers contend that military involvement in seeding other areas such as microwave technology, aerospace, and even the Internet have continued to have a pervasive influence over the Valley. In this respect, the U.S. Department of Defense can be considered the original venture capitalist that funded, and continues to nurture, the Valley's spectacular growth. The development path of Santa Clara County cannot be divorced from federal contracts, corporate strategy, and technological innovation shaped by the Cold War period. Even today, Silicon Valley defense-related firms continue to be prominent employers (e.g., the Lockheed-Martin production facility in Sunnyvale and the R&D center in Palo Alto). Silicon Valley institutions such as Stanford University and SRI continue to attract substantial military funds, and the San Jose metropolitan area is one of the major beneficiaries of prime contracts per worker, roughly four times the national average.

A further component in this explanation of Silicon Valley growth is the corporate sector. While the popular heroes of the valley are the small firm and the entrepreneur, large corporations continue to be the dominant source of all employment. In fact, some observers link growth in small firm formation and employment to periods of downturn in the Silicon Valley large-firm sector. Power relations between large and small firms are invariably dominated by the large. They can impose rigid and hierarchical organizational arrangements on the small firms that act as their subcontractors or suppliers. This is a far cry from the world of flexible networking relations built on trust and mutual cooperation. Finally, this variant of the Silicon Valley story highlights the globalizing tendencies of many of the large corporations in the Valley such as Intel, HP, Sun Microsystems, and Applied Materials. These corporations engage in global collaborations that have very little to do with proximity and much to do with performance and capabilities. As a result, many of the once local flows of capital and know-how that are an integral part of the previous explanations of Silicon Valley success have now become externalized in a pattern that is much more complex than the economic growth and regional culture accounts would have us believe.

In sum, each of the above perspectives contains a partial account of the evolution of Silicon Valley. The whole story probably lies at the intersection of these individual explanations. All three features of Silicon Valley can be readily identified and the question is probably one of emphasis and degree. While no single perspective is the "correct" one, there is a need to know exactly what makes the valley "tick" if only for the purpose of preserving its success and informing efforts that try to replicate the Santa Clara model in other locations across the globe.

IV. NATIONAL AND REGIONAL DIFFUSION OF SILICON VALLEY CULTURE

Silicon Valley is more than just a geographic concentration of technological innovation. During the last 30 years a specific Silicon Valley style has emerged that reflects the unique way in which work, life and business are conducted in the area. The diffusion nationally of this approach to life and work and its accompanying values creates a specific Silicon Valley culture. In this respect the physical agglomeration in the Valley acts as a key link in taking technological innovations and translating them into societal change. While many technological revolutions have been associated with the development of an accompanying culture, it would seem that the ability of Silicon Valley to sustain its growth and constantly redefine itself over a 50-year period must be explained by more than just technological dominance. As noted earlier, from the late 1960s the Santa Clara area became associated with cutting-edge developments in both technology and lifestyle. San Francisco-inspired social thinking fused with innovative technology and led to the birth of the PC. This invention was seen by its founding fathers as both a technological and socially liberating instrument.

Subsequently, the valley has become associated with a strong cultural specificity that, like the Silicon Valley economic growth model, has become an object of imitation and admiration. The main components of this culture are an untrammeled belief in individualism and entrepreneurialism, a demanding working life, visible affluence and conspicuous consumption, the development of a particular corporate subculture, advancement on the basis of merit and adherence to the principles of equal entry, tolerance, and openness to new ideas.

Much of the Gold Rush atmosphere of Silicon Valley is due to a profound faith in the ability of entrepreneurship to animate the regional economy. Despite the presence of many large companies in the Valley, the role model continues to be the start-up and the small entrepreneur. The Internet-driven stock market boom of 1999–2000 took this tendency to the extreme. It resulted in the formation of many small firms, some of which were created for the express purpose of being sold and realizing huge capital gains. The faith in the entrepreneur complements the frontier-type belief in individualism as furthering the common good. As a result, business failure and bankruptcy, job-hopping and excessive risk taking are considered virtues rather than vices. A vast variety of institutions and services are available to allow the individual to materialize the goal of creating a firm, involving dense networks of producers and business services such as lawyers, venture capitalists, headhunters, and management consultants. In this atmosphere, government is treated with a measure of distrust, because its traditional role of taxing and regulation represents a potential threat to individualism. Note, however, that much of the entrepreneurial context has in fact been fashioned by state government. A liberal attitude in California's state legislature towards bankruptcy, capital gains, or postemployment covenants has allowed the culture of individualism to flourish.

The drive to success, however, takes its toll. A Silicon Valley work culture has evolved based on long working hours, burnout, and "technostress." Technological progress has extended the workday, increased connectivity with other places, and increased the toll on the individual worker. This then percolates into home life with surveys reporting above average levels of social and psychological stress among high-level Valley employees. Job-related stress also finds expression in social ills such as alcohol and substance abuse, family disruption and divorce, and emotional disorders in children. The combination of intensive work culture and extreme individualism lead to a particular demographic structure. Surveys point to nearly 35% of the upper level employees in Silicon Valley as never married, 18% divorced, and only 32% over the age of 45. This in turn, has implications for the physical development of the Silicon Valley towns and communities. Traffic patterns, demand for leisure, housing provision and public services (such as schools and hospitals) are all heavily influenced by the structure of local demography.

Another cultural trait of Silicon Valley is the tendency toward visible affluence and conspicuous consumption. While the average, regionwide wage in 2000 was nearly double the national average, this figure serves to obscure both the great variation in wages across subsectors that exists in Silicon Valley and also the skewness of the regional income distribution. Average wages in leading Silicon Valley subsectors are double the regional average with software jobs averaging $125,000 per year and semiconductors $117,000 per year in 2000. However, wages and salaries may not even be the correct indicators of personal wealth due to the enormous capital gains from stock options that have accrued to Silicon Valley employees in recent years. The standard of living in Santa Clara County is therefore among the highest nationwide, driven by the consumption of a high-wage, equity-rich portion of the local population. Some observers interpret the culture of conspicuous consumption in Silicon Valley as a stress release mechanism rather than an expression of status.

The Silicon Valley corporate culture is expressed through a few flagship examples that have been the source of management inspiration for other firms both in the valley and beyond. Foremost among these is Hewlett-Packard, which operates a benevolent organizational style. This "HP way" emphasizes decentralized management and decision making, negotiated corporate goals, the encouragement of "intrapreneurship" within the company, collaboration and team work between employees, the virtual absence of any union presence, and the absence of symbols of hierarchy among the labor force. These principles have also served other leading Valley corporations such as Intel and Apple. While this management style and the pecuniary rewards that accompany it, such as stock options and equity participation, are aimed at cultivating employee loyalty and cutting down the high levels of job turnover prevalent in the Valley, recent evidence would seem to point to the opposite. A loose management style and volatile stock values have done little to cut down employee circulation across firms. In fact, declining stock values can often act as an incentive to increase interfirm mobility.

Promotion and advancement within or between firms in the Valley is strictly on the basis of merit. In this respect the Valley culture adheres strictly to the principles of equal entry and tolerance. "Glass ceilings" whether of a racial or gender variety hardly exist. The combination of labor mobility and equal entry means that that the region calls on skilled workers from an international labor market. Thirty-five percent of the Silicon Valley labor force in 2000 were foreign born and aspiring immigrant entrepreneurs and engineers converge on Silicon Valley from all over the world. By operating open labor markets, the region

can sustain its lead in innovation. Among the foreign nationalities represented in the valley, the Indians, Chinese, Israelis, Taiwanese, French, and Iranians form particularly distinctive groupings. Chinese and Indian entrepreneurs alone are estimated to have started close to 2,800 firms employing (in 1998) some 60,000 workers. Aside from creating businesses, immigrant entrepreneurs have also created dense webs of networking relationships between themselves and with their home countries, allowing for information exchange and job searches. In this way they add a further layer of networks to those existing in the Valley and generate an important asset that allows the region to perpetuate its position as the world's primary high-tech location.

V. THE OVERHEATING VALLEY

There is also a price to pay for success. The ability of Silicon Valley to absorb a constantly rising standard of living and ensure a desirable quality of life is finite. In the long run, excess pressures on the local environment will have ramifications on the economic growth of the region, causing the valley to "overheat" with its own success. The sustainability of this economic growth is nowhere more apparent than in the areas of land development, transportation, and air and groundwater quality. Rapid urban development has put enormous pressure on land resources, causing local house prices to rise rapidly. The average house price in Silicon Valley in 2000 was in excess of $600,000, representing an increase of nearly 90% over a 5-year period. With prices set at these levels it becomes increasingly difficult to attract labor to the region. Skilled workers can find equally attractive job offers at alternative locations where the cost of living is much lower. Low-skilled workers who comprise roughly half the labor force in the Valley area are either forced out or increasingly engage in doubling and tripling-up in single-family homes. It has been estimated that only 16% of the region's housing stock is affordable to households earning the median income, in contrast to the national average of 60%.

Rising prices are partially a result of the housing market not responding to the constant increase in the number of jobs created in the region. Between 1992 and 2000, Silicon Valley added 330,000 new jobs but only 60,00 new housing units. This jobs–housing imbalance is the result of changes in the legislative environment governing the collection of local taxes and the funding of local public services. This has made commercial and office development more attractive to city governments than residential building. Coupled with local community opposition to higher residential densities, this preference for nonresidential development explains why the housing market has lagged behind job growth. The result has been an increasing number of Silicon Valley employees living at ever-greater distances from their place of work. In 1990 the number of workers commuting into Santa Clara County from surrounding areas was 144,000. By 2000, this figure had increased 47% to 212,000 and Santa Clara's share of "imported" labor had grown from 16% in 1990 to 20% in 2000. This regional labor market can stretch in some directions up to 100 miles. In areas such as the San Joaquin Valley, communities such as Tracy, Manteca, and Modesto serve as bedroom communities for employees who work 80 to 100 miles away in Silicon Valley.

Increased commuting implies increased pressure on local transportation. Traffic congestion represents a waste of resources, increased air and water pollution, productivity loss, and a lower quality of life. Within the San Francisco Bay Area, 5 of the 10 most congested routes are those leading to and from Silicon Valley. Economic growth, job creation, and a sluggish housing response have simply served to exacerbate the problems of traffic congestion and commuter gridlock. Many solutions to this problem have been proffered. These include both incremental improvements to the existing infrastructure (adding freeway lanes, improving express bus and commuter rail services), changes in land use (such as transit-oriented development), and promoting alternative transit modes such as carpool/rideshare arrangements, light rail, and HOV (high occupancy vehicle) lanes.

Both transportation and land use pressure threaten the quality of the local environment. Increasing car dependence in Silicon Valley has caused increased emissions of volatile organic chemicals and nitrogen oxides. In recent years, Santa Clara Valley has exceeded state-mandated emission levels for an average 10 to 20 days a year and exceeds State standards for particulate matter for an estimated 20 to 30 days a year. In terms of groundwater quality, rapid urban growth means that household chemicals and fertilizers, metal particulates from cars, and MTBE (a chemical compound that reduces gasoline air pollution but contaminates groundwater) all percolate into the region's aquifiers. Water consumption has increased 30% since 1990 but water protection and recycling has not proceeded apace. The threat of toxic industrial water pollution, which posed an environmental threat in the 1980s, seems to have been contained. However, fast urban growth has left Santa Clara County

with 29 contaminated sites prohibited for development (Superfund sites)—more than any other county in the United States. Most of these "brownfield" sites were contaminated by the electronics industry.

City governments have been taking steps to ensure that land quality does not fall victim to fast economic growth in the Valley. Twenty-four percent of the Silicon Valley area is designated as permanently protected open space. Many cities have adopted urban growth boundaries (UGBs) to ensure that this land remains protected in the future. However, success in ensuring sustainable growth is contingent to a great extent on interjurisdictional cooperation. Despite its progressive reputation, California has no state-wide planning policy and government fragmentation and competition often inhibits a concerted region-wide response to the environmental threats posed by economic growth.

Aside from the environmental questions, the issue of social equity also casts a shadow on the desirability of further valley growth. With rapid growth, large income disparities have developed in the region. As noted earlier, huge capital gains have accrued to a small stratum of the local population. This economic gain, however, has not filtered down to the lower income households. In terms of inflation-adjusted incomes, the bottom 20th percentile of the Silicon valley population was in fact worse off in 1999 than it was it 1993, despite a 20% income rise nationally among this group during this period. At the other end of the income distribution, the 80th percentile experienced a 20% rise in incomes over the same period. The average income for this group was $149,000 in 1999 compared to $40,000 for the bottom 20% of households.

This skewed income distribution is not equally allocated across ethnic groups in Silicon Valley. While Hispanics account for one-quarter of the region's population, they comprise an estimated 50% of the local working poor. High school graduation rates have also declined over the period 1993–1999 and again this indicator has an ethnic dimension. Asian students achieve the highest graduation rates (97%) while Hispanic students register the lowest (59%). Combined with other indices of educational disparity by ethnic group (university entrants, algebra proficiency, etc.), these facts suggest the existence of a "digital divide" within Silicon Valley. With the demand for skill requirements constantly growing, the existence of such a divide implies a loss of human capital resources and the perpetuation of a bifurcated social structure. Ironically, the region that has done so much to diffuse the technological revolution worldwide may have created a technology gap on its own doorstep.

VI. THE SILICON VALLEY MODEL: SOME CONCLUSIONS

As both a place and a concept, Silicon Valley defies easy definition. Geographically, the valley is not a readily measurable administrative unit and straddles four counties. In terms of physical morphology, it is neither city nor suburb. Rather it displays an urban form comprised of a collection of bedroom communities, low-density commercial, office and industrial centers, and open space linked by a web of freeways. Defining exactly what Silicon Valley does and how successfully it does it is also fraught with difficulty. "High technology" is not a readily identifiable industry in national accounts. In addition it is hard to measure high-technology output, especially in the case of software and other intangible products. Similarly, high-technology employment defies specification. Internet employment for example, does not readily match any of the standard occupational classifications.

Silicon Valley has also traditionally been a rather insular place. The "digital divide" that has been created within the region itself is a mark of embarrassment to a region that has done so much to diffuse technology worldwide. Similarly, the dichotomized income distribution that exists within the region is testimony to the fact that (corporate philanthropy aside) little of the economic welfare generated in the valley has trickled down to the wider community. Some observers believe this is simply a temporary stage in Silicon Valley development. As the Valley's business heroes begin to age, their level of interest and involvement in the wider community is expected to intensify. In addition, the nature of business finance in Silicon Valley is based on the fact that many of the huge capital gains in high-tech activity are reinvested back into new local companies, thereby perpetuating the growth of the area. Thus funds for the community are not always readily available.

Silicon Valley has captured the world's imagination because it gives a glimpse into life in the New Economy. Thus it does not just refer to an economy with increasing numbers working in high-technology industries. More fundamentally, it alludes to the basic economy-wide change driven by information technology in all its facets. Silicon Valley is not just a story of the production of leading-edge products in a particular location. Rather, it is the story of the way new technology has permeated traditional sectors and services and transformed the way business is done starting with venture capital through stock options and on to business angels.

Even more remarkable is the resilience of this form of business model. It has constantly redefined itself over a span of 50 years in a market characterized by boom and slump periods. The Wall Street collapse in 2001 and subsequent crisis in Silicon Valley must be seen in this perspective. In previous recessions (mid-1970s, Wall Street crash in 1987, post–Cold War slowdown in 1992, and so on), Silicon Valley always managed to emerge riding on the wave of a new technology. However this does not mean that other places will be able to emulate this pattern. The historical landscape is filled with places that rose to prominence on the back of a particular product or process technology but never managed to reemerge following its demise.

It is necessary to stress that Silicon Valley represents a form of development conditioned by very particular historical, place-specific, and environmental conditions. Two important features should be noted. The first relates to the role of government in spawning the valley economy. For all its glorification of the entrepreneur and belief in market forces, the pervading influence of government in seeding the valley cannot be denied. Defense contracts for semiconductors in the formulative years of 1958–1974 were worth a total of $40 billion. The Internet basically started as a government and defense project. State government was also influential. The liberal tax and capital gains structure, the legislative attitude toward business failure and bankruptcy, intellectual property rights, and labor mobility all helped create the context for local competitive advantage. Second, the Silicon Valley model cannot be adopted as a "quick fix" solution by lagging regional economies. As has been shown, the valley had a particularly lengthy incubation period and only began to mature into a fast-growth region in the last quarter of the 20th century. Silicon Valley therefore took the best part of a century to evolve.

The ingredients of Silicon Valley success would seem readily identifiable: strong university research base, entreprenueurial corporate culture, developed social and interfirm networks, industrial clusters, venture capital markets, supportive government, and high quality of life. However, the list of failed replicas that spans many different countries and landscape forms (Silicon Mountains, Glens, Prairies, Beaches, Forests, Alleys, etc.) attests to the fact that while the individual conditions for success can be isolated, the system as a whole is not easily reproduced. Even Fredrick Terman failed when trying to recreate the Stanford Research Park model in other locations such as Dallas and New Jersey. Other places that have been associated with major new technologies, such as Urbana–Champaign, the birthplace of the Internet browser, have not been able to generate their own regional economic growth dynamic around these technological leads. It would seem that aspiring regions have to develop their own adaptations of the system and generate their own particular regional equilibria in that way. They need, however, to be aware that some of the undesirable consequences of the Silicon Valley experience, such as social divisions, inequitable income distribution and environmental overheating, may emerge before any of the regional economic benefits begin to materialize.

SEE ALSO THE FOLLOWING ARTICLES

Digital Divide, The • Economic Impacts of Information Technology • Future of Information Systems • Globalization • Internet, Overview • People, Information Systems Impact on • Sociology

BIBLIOGRAPHY

Castells, M., and Hall, P. (1994). *Technopoles of the world: The making of 21st century industrial complexes.* London: Routledge.

Cohen, S. S., and Fields, G. (2000). Social capital and capital gains: An examination of social capital in Silicon Valley. *Understanding Silicon Valley: The anatomy of an entrepreneurial region.* Kenny M. (Ed.). Stanford, CA: Stanford University Press, 190–217.

DeVol, R. C. (1999). *America's high tech economy: Growth, development and risks for metropolitan areas.* Los Angeles: Milken Institute.

Economist (March 29, 1997). Future perfect? A survey of Silicon Valley.

Florida, R., and Kenney, M. (1990). Silicon Valley and Route 128 won't save us. *California Management Review,* Vol. 33, No. 1, 69–88.

Gray, M., Golob, E., Markusen, A. R., and Park, S. O. (1999). The four faces of Silicon Valley. *Second tier cities: Rapid growth beyond the metropolis.* A. R. Markusen, Y.-S. Lee, and S. DiGiovanna (Eds.). Minneapolis: University of Minnesota Press, 291–310.

Harrison, B. (1994). *Lean and mean: The changing landscape of corporate power in the age of flexibility.* New York: Basic Books.

Kenney, M. (Ed.) (2000). *Understanding Silicon Valley: The anatomy of an entrepreneurial region.* Stanford, CA: Stanford University Press.

Lee, C.-M., Miller, W. F., Hancock, M. G., and Rowen, H. S. (Eds.) (2000). *The Silicon Valley edge.* Stanford, CA: Stanford University Press.

Rogers, E. M., and Larsen, J. K. (1984). *Silicon Valley fever: Growth of high technology culture.* New York: Basic Books.

Saxenian, A. (1994). *Regional advantage: Culture and competition in Silicon Valley and Route 128.* Cambridge, MA: Harvard University Press.

Saxenian, A. (1999). *Silicon Valley's new immigrant entrepreneurs.* San Francisco, CA: Public Policy Institute of California.

Natural Resource Management

H. Michael Rauscher and Keith M. Reynolds
USDA Forest Service

GLOSSARY

bayesian belief network A causal network describing the dependency between system states in terms of judgments and probabilities.
decision support systems A software technology to help managers make decisions in situations where human judgment is an important contributor to the problem-solving process, but where limitations in human information processing impede decision making.
full service ecosystem management decision support systems Decision support systems designed to operate across the entire ecosystem management domain.
functional service modules Software service modules that provide specialized functional capabilities to full service decision support systems.
fuzzy logic A branch of applied mathematics that extends classical set theory by quantifying an observation's degree of membership in a set.
hypermedia or hypertext A highly nonlinear and interactive mixture of electronic text, graphics, images, video, and audio organized into conceptual chunks connected by meaningful links.
hypertext chunk or page An organized collection of information on a single topic which is internally self-contained and independently understandable.
hypertext link An electronic cross reference used to connect logically related chunks.
knowledge application The information systems or tools we create to assist us in using knowledge for some human purpose. Focus is on using knowledge.
knowledge-based systems A software technology for capturing qualitative, experience-based, inexact knowledge and then reasoning with it to solve problems.
knowledge management The methods we use to create, evaluate, organize, and distribute knowledge. Focus is on accessing knowledge.
knowledge repesentation The methods of structuring, encoding, and storing symbols so that the attributed meaning is clear and manipulating these symbols is possible.
mathematical simulation models A software technology for formulating problems in mathematical terms and then using numerical analysis methods to solve them.
truth value In fuzzy logic, a metric that expresses the degree to which a proposition is supported by its premises.

I. INTRODUCTION

Information systems are used to support both knowledge management and application in natural resource management. Developing the theory and practice of how to use information technology to support effective knowledge management and application is one of the most important challenges facing systems science today. The purpose of this chapter is to review how information technology is being used to support knowledge management and application in *natural resource management.* To accomplish this we will first introduce and define knowledge, its symbolic representation, and

the three major types of knowledge systems: *descriptive, predictive,* and *prescriptive*. Next, we will review and compare existing natural resource management knowledge management and application systems. Finally, we will present two particular natural resource management decision support systems, the Ecosystem Management Decision Support System (EMDS) and NED—a decision support system for natural resource management, and provide an application example for each.

Access to knowledge *(knowledge management)* and the ability to use it wisely *(knowledge application)* have always been the hallmark of successful individuals, companies, and nations. This is no less true for agriculture and natural resource management than it is for all other knowledge intensive activities. Before the mid-1920s, gains in agricultural and natural resource management productivity resulted largely from improvements in the application of human and animal power. Between the 1920s and 1950, mechanical power largely replaced human and animal power in the United States, resulting in enormous productivity increases. Since 1950, increases in agricultural and natural resource management productivity have resulted from improvements in knowledge management and application, which led to the development of genetic engineering and improved hybrids, fertilizer, and pesticides.

The methods we use to create, evaluate, organize, and distribute knowledge define the field of knowledge management. Until recently, knowledge management had changed little since the invention of the printing press in the middle of the fifteenth century. Our repositories of knowledge still largely consist of static, linear print media. Recent technological advances, however, introduced new tools for improving knowledge management methods. These include (1) the electronic computer (ca. 1942 to present), (2) knowledge-based systems (ca. 1956 to present), (3) hypertext concepts and software (ca. 1960 to present), and (4) the Internet telecommunications system (ca. 1986 to present). In combination, these four information technological developments are revolutionizing the way people think and the way the world works just as profoundly as the earlier invention of the printing press. The better we understand how to create, organize, manage, and deliver knowledge, the more efficient we will be as producers, distributors, and consumers of knowledge.

Decision-support capabilities (knowledge application) in natural resource management have been challenged to satisfy an ever-growing demand for useful knowledge tailored to individual problems and needs. In other words, we need ways to get the right knowledge, to the right people, at the right time, in the right form. This is particularly important as natural resource management moves from a difficult multiple-resource management paradigm to an even more difficult natural resource management paradigm. The need for better and more powerful knowledge application aids has become urgent.

II. KNOWLEDGE REPRESENTATION AND MANAGEMENT

Knowledge can reside in diverse forms and media. For our purposes, we will concentrate on knowledge encoded in language. Language-based knowledge resides in shared systems for symbolic communication. There are *natural* languages, such as English, which are robust, but ambiguous (Table I) and *artificial* languages such as FORTRAN, calculus, and symbolic logic which are used to describe a purposefully limited set of phenomena precisely. Artificial languages use strictly controlled vocabulary and grammar with simple structures designed to eliminate ambiguity. The cost of this precision in communication is lack of generality and increased brittleness—small errors in grammar cause comprehension to fail.

Languages, both natural and artificial, can be used to represent knowledge. The concept of knowledge representation refers to the methods of structuring, encoding, and storing symbols so that the attributed meaning is clear. All language-based knowledge representation methods and the electronic software that have been developed to manipulate them can be organized by three attributes (Table II): (1) generality, the size of the set of problems that the method is capable of addressing; (2) ambiguity, the precision with which meaning can be attributed to the symbols used; and (3) power, the effectiveness of the method in solving specific problems.

A number of different tools exist for storing and manipulating language-based knowledge—the predominant format for natural resource management knowledge (Table II). Word processors manipulate natural language documents, which may include graphic images, and help us manage print technology, based on a linear, sequential presentation logic. Given our continued reliance on print technology, word processors are the current *de facto* tool for knowledge management. Because natural language has great generality, it can be used to represent almost any knowledge, but in doing so sacrifices clarity and precision. In most natural language text documents, the structure of the subject is more or less hidden, camouflaged by the sequential nature of the medium and the need to gracefully and carefully transition from one idea to the next so the reader can under-

Table I Comparison of Natural and Artificial Language Characteristics

Natural languages (e.g., English, Spanish)	Artificial languages (e.g., BASIC, Calculus)
1. Not controllable	1. Formally defined and controlled
2. All valid words and structures cannot be enumerated	2. All valid words and structures can be enumerated
3. Ambiguous	3. Precise
4. Multiple definitions	4. Single definitions
5. Robust	5. Brittle
6. Implicit meanings allowed	6. Implicit meanings not allowed

stand the author's meaning. With some notable exceptions (such as encyclopedias), structure in natural language text documents is subservient to the content of the material being presented.

Improvements in information technology have led to the development, testing, and gradual application of three important classes of knowledge management tools: hypermedia systems for *descriptive* knowledge, quantitative and qualitative simulation models for *predictive* knowledge, and decision support systems for *prescriptive* knowledge. The following sections briefly review these relatively new knowledge management tools and discuss how these tools have been used in natural resource management.

A. Hypermedia Systems for Managing Descriptive Knowledge

Anyone who has accessed the World Wide Web has been exposed to hypermedia documents—a highly nonlinear and interactive mixture of text, graphics, images, video, and audio. (Note the terms "hypermedia" and "hypertext" are used interchangeably throughout this document.) Abstractly, a hypermedia document consists of a network of chunks connected by links (Fig. 1). A *chunk* (or page) is an organized collection of information on a single topic. Chunks are internally self-contained and independently understandable. A *link* is an electronic cross reference used to connect logically related chunks. The act of linking chunks creates the hypermedia document and at the same time creates *one* set of navigational jumps for readers to follow. Links simulate the mental association between chunks in the mind of the author. The *structure* of a hypermedia document refers to those elements that deal only with the organization of chunks. Structure commonly takes the form of tables of contents, outlines of chunks, graphical diagrams of chunk relationships (e.g., Fig. 1), indices, link organization, etc. The *content* of a hypermedia document refers to the domain-specific material that makes up the subject matter.

In contrast to natural language text documents, hypertext forces the author to explicitly highlight the structure (outline or concept map) first and foremost for the user. Only secondarily, is the user exposed to the content matter. It is the structure that guides the

Table II Language-Based Knowledge Representation Methods Organized According to Problem-Solving Power and Generality

		Artificial language systems		
			Mathematics	
Natural language systems		Logic		
NLT[a] documents	Hypertext documents	KBS[b] qualitative simulations	Numerical equations quantitative simulations	Analytical equations quantitative simulations
High — Generality — Low				
High — Ambiguity — Low				
Low — Problem-solving power — High				

[a]Natural language text.
[b]Knowledge-based systems (KBS).

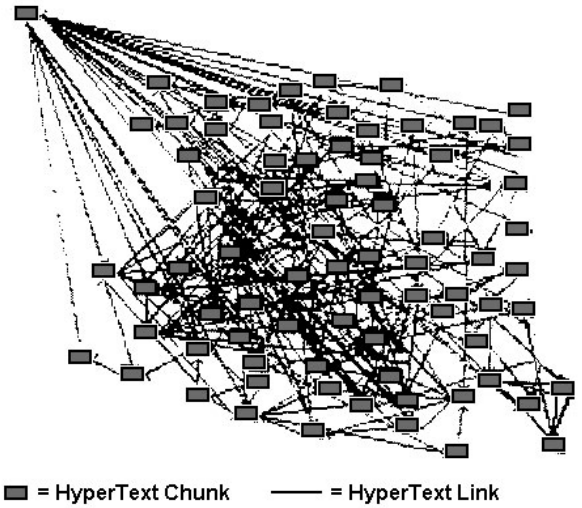


Figure 1 A conceptual representation of a three-dimensional network of chunks and links that make up a hyperdocument.

user, time and again, to try different paths in the hypermedia document. There are many possible sets of navigational jumps—other than the single one envisioned by the hypermedia author—for example, dynamic ones based on the user's responses to questions or on the user's navigational patterns. The author can no longer rely on sequential reading to present material. A reader can arrive at a particular location in the hypermedia document from many different starting points. Consequently, each page (or chunk) of the hypermedia document must be independently understandable, much as we demand that journal figures and tables be self-contained. Hypermedia authoring methods sacrifice some generality to increase the power of text to communicate meaning more clearly and to increase its problem-solving power by reducing ambiguity (Table II). Several examples of hypermedia knowledge management systems have been developed for the natural resource management domain. Rauscher developed the *Encyclopedia of AI Applications to Forest Science* in 1991, the *Ecology and Management of Aspen* in 1995, the *Northeast Decision Model Design Document* in 1995, and *Oak Regeneration: A Knowledge Synthesis* in 1997. Reynolds and Rauscher developed a Hypermedia Reference System to the Forest Ecosystem Management Assessment Team Report in 1995.

The emergence of the Internet and widely available hypermedia authoring systems has led to a paradoxical situation. As early as 1945, Vannevar Bush, science advisor to President Truman, pointed out the need to develop information technology to synthesize and thereby compact the huge, unwieldy, fragmented nature of human knowledge. Bush's prescience provided the motivation for scientific research into hypertext systems. Despite the current availability of a mature theory and practice of how to author powerful hypermedia systems, the typical HTML document is essentially a paper document in electronic form. A simple experiment will convince the reader of this reality. Use any capable Internet search engine to query on the term "ecosystem management." We obtained an overwhelming 5003 "hits." A large number of these sites are self-promotional in nature, i.e., advertisements for books or for projects on which a particular university or government agency is working. Some of the better sites simply provide paper documents in electronic form (i.e., linear, sequential text and illustrations that look, read, and convey information just like a paper document and which are, in fact, meant to be printed, rather than consumed in electronic form). For example, the Ecological Society of America provides a nice report on "The Scientific Basis for Ecosystem Management" (www.sdsc.edu/ecmtext.htm) that is 89 screen pages long. The Cato Policy Institute has available a very nicely written article warning of the dangers of ecosystem management (www.cato.org/pubs/pas/pa-217.html) that is 59 screen pages long. In contrast, good hypermedia authoring practice is that no single chunk or "Internet Page" of content be longer than 3–5 screen pages and the shorter the better. Compare the above documents with those produced by Reynolds and Rauscher and the reader will immediately see a major improvement in knowledge synthesis, understandability, and ease of navigation. The truth is that most current hypermedia systems on the Internet do not help diminish knowledge fragmentation nor do they advance knowledge synthesis or knowledge management.

B. Knowledge-Based and Simulation Models for Managing Predictive Knowledge

Along with hypertext systems, knowledge-based systems (KBSs) are key knowledge management tools (Table II). They grew out of the science of symbolic logic and were adapted for practical problem solving by scientists in the field of artificial intelligence (AI). Researchers in AI realized that much of what human experts know cannot be easily formulated into a system of mathematical equations. In structure, a KBS may consist of many components (Fig. 2). At the heart of any KBS is its knowledge base. Knowledge from experts, or other expert sources (e.g., literature), is codified into logical statements (the knowl-

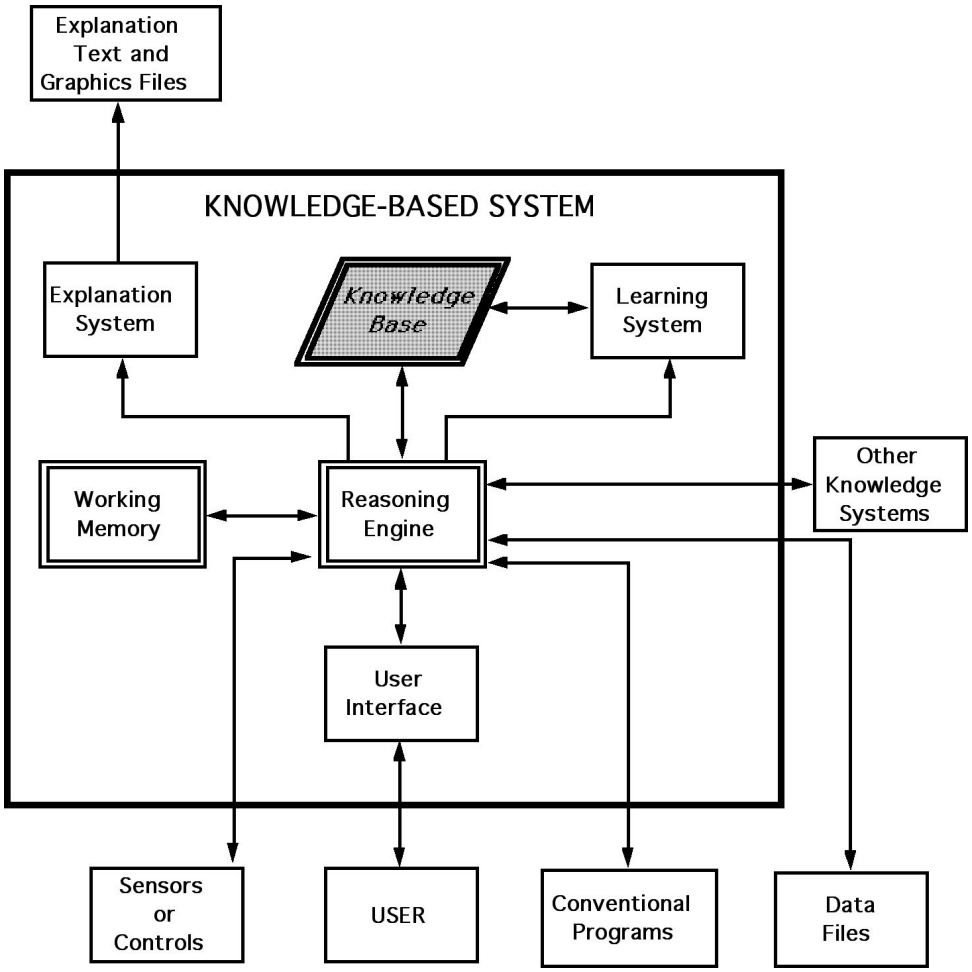


Figure 2 A KBS potentially contains many components. Note that the explanation subsystem of many KBS is implemented as a hyperdocument.

edge base) which can be manipulated for problem-solving purposes. Knowledge-based systems sacrifice even more generality than hypertext systems. They are more costly to develop than hyperdocuments and they cannot capture the breadth of knowledge that hyperdocuments can. They, however, are more precise (less ambiguous) and more compactly represent knowledge than hyperdocuments. Furthermore, KBS increase problem-solving power by supporting automated reasoning about the domain of interest.

Despite our most strenuous efforts to quantify important ecological processes to support a theory in simulation model form, by far the larger body of what we know can only be expressed qualitatively, comparatively, and inexactly. Most often this qualitative knowledge has been organized over long years of professional practice by human experts. While KBSs allow us to capture some of this qualitative, experience-based expertise, it is still not possible to capture the full range and flexibility of knowledge and reasoning ability of human experts in knowledge-based software. We have learned, however, in many cases how to capture and use that portion of expertise that the human expert considers routine. Many KBSs have been developed for the agriculture and forestry domains and have been published in the scientific journals *AI Applications, Computers and Electronics in Agriculture,* and others. An extensive presentation of KBS and how to develop them specifically for natural resource management purposes can be found in a 1996 book by Schmoldt and Rauscher.

Simulation models are the information technology of choice when knowledge can be expressed mathematically (Table II). In the last 20 years, an impressive amount of mathematical simulation software

has been developed for all aspects of natural resource management. Schuster et al. (1993) conducted a comprehensive inventory of simulation models available to support forest planning and natural resource management. They identified and briefly described 250 software tools. Jorgensen and colleagues produced another compendium of ecological models that incorporate an impressive amount of ecosystem theory and data. There is no way to even begin to cover all the available simulation software useful for natural resource management.

Both knowledge-based and simulation models for natural resource management are typically developed independently of one another. As a result, they are large, monolithic, stand-alone systems that function like "little islands of automation" unable to easily communicate with each other. Although collectively, the existing knowledge-based and mathematical simulation models could potentially address many of the predictive forecasting needs of managers, this potential is likely to go unrealized until effective software communication standards are introduced and followed. Natural resource managers need integrated suites of software tools to provide comprehensive coverage of the range of biological predictions needed. Although such interoperability standards have received considerable attention outside the realm of natural resource management, they have, until recently, been largely ignored in natural resource management software development.

C. Decision Support Systems for Prescriptive Knowledge Management

Decision support systems (DSS) help managers make decisions in situations where human judgment is an important contributor to the problem-solving process, but where limitations in human information processing impede decision making. The goal of a DSS is to amplify the power of the decision makers without usurping their right to use human judgment and make choices. They attempt to bring together the intellectual flexibility and imagination of humans with the speed, accuracy, and tirelessness of the computer.

Decision support systems may contain a number of subsystems, each with a specific task (Fig. 3). The first, and most important, is the subsystem composed of the decision maker(s). Decision makers are consciously diagrammed as part of the DSS because without their guidance, there is no DSS. The group negotiation management subsystem helps decision makers organize their ideas, formulate relationships surrounding issues and arguments, and refine their understanding of the problem and their own value systems. Examples of group negotiation tools include: the active response Geographic Information System (AR/GIS) and the issue-based information system (IBIS). Group negotiation tools are used to construct issue-based argument structures to clarify the values and preferences of group members in the attempt to reach group consensus. For example, IBIS uses for-

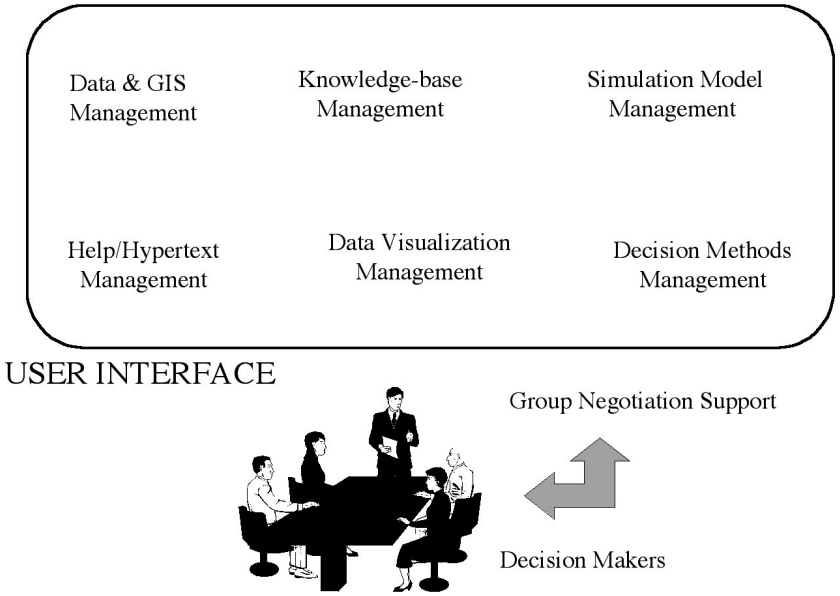


Figure 3 The major components of a generic DSS.

mal argument logic (the logic of questions and answers) as a way to diagram and elucidate argumentative thinking. By asking and answering crucial questions, you can begin to better understand the problem and its solution set. Decision support systems should specifically employ mechanisms by which the biological realities guide and, if appropriate, constrain the desires of the stakeholders. For example, compromise is not acceptable for some issues. If the productive capacity of an ecosystem is fixed yet key stakeholders all want to extract a product from that ecosystem at a higher level, a compromise midway between the levels will be unsustainable.

The next major subsystem, spatial and non-spatial data management, organizes the available descriptions of the ecological and management components of natural resource management. Data must be available to support choices among alternative management scenarios and to forecast consequences of management activities on the landscape. There is a trade-off between the increasing number of goals that decision makers and stakeholders value and the high cost of obtaining data and understanding relationships that support these choices. Monitoring both natural and anthropogenic disturbance activities and disturbance-free dynamics of managed forest ecosystems are also extremely important if a DSS is to accurately portray the decision choices and their consequences. Barring blind luck, the quality of the decision cannot be better than the quality of the knowledge behind it. Poor data can lead to poor decisions. It is difficult to conceive of prudent natural resource management without an adequate biophysical description of the land base in question.

The next three subsystems—hypertext, knowledgebase, and simulation model management—deal with effectively managing knowledge in the many diverse forms in which it is stored, represented, or coded. These systems have been covered in more detail in the preceding sections. The simulation model management subsystem of the DSS is designed to provide a consistent framework into which models of many different origins and styles can be placed so the decision makers can use them to analyze, forecast, and understand elements of the decision process. The knowledge management subsystem of the DSS is designed to organize all available knowledge-based models in a uniform framework to support the decision-making process. The software subsystems of a DSS described so far help decision makers organize the decision problem, formulate alternatives, and analyze their future consequences. The decision methods management subsystem (Fig. 3) provides tools and guidance for choosing among the alternatives, for performing sensitivity analysis to identify the power of specific variables to change the ranking of alternatives, and for recording the decisions made and their rationale.

There are many facets or dimensions that influence the decision-making process. The rational/technical dimension, which concerns itself with the mathematical formulation of the methods of choice and their uses, is the one most often encountered in the decision science literature. But there are others including the political/power dimension and the value/ethical dimension.

Decision makers might find themselves at any point along the political/power dimension bounded by a dictatorship (one person decides) on the one extreme and by anarchy (no one can decide) on the other. Intermediate positions are democracy (majority decides), republicanism (selected representatives decide), and technocracy/aristocracy (experts or members of a ruling class decide). Currently three approaches seem to be in use at multiple-societal temporal and spatial scales: management by experts (technocracy), management by legal prescription (republicanism), and management by collaboration (democracy). No one approach predominates. In fact, the sharing of power between these three approaches creates tensions which help make natural resource management a very difficult problem. In the context of natural resource management, the value/ethical dimension might be defined on the one extreme by the preservationist ethic (reduce consumption and let nature take its course) and on the other by the exploitation ethic (maximum yield now and let future generations take care of themselves). Various forms of the conservation ethic (use resources, but use them wisely) could be defined between these two extremes. The rational/technological dimension is defined by the normative/rational methods on the one hand and the expert/intuitive methods on the other. Numerous intermediate methods also have been described and used. The formal relationships between these dimensions affecting the decision process have not been worked out.

Informally, it is easy to observe decision-making situations where the political/power or value/ethical dimensions dominate the rational/technical dimension. Choosing an appropriate decision-making method is itself a formidable task that influences both the design of alternatives and the final choice. Many DSSs do not offer a decision methods subsystem due to the complexity and sensitivity of the subject matter. Unfortunately, providing no formal support in DSSs for choosing

among alternatives simply places all the burden on the users and may make them more vulnerable to challenges of their process and choice mechanisms.

III. A COMPARISON OF EXISTING NATURAL RESOURCE MANAGEMENT DSS

In 1997, Mowrer and colleagues published a survey of 24 of the leading natural resource management DSSs developed in the government, academic, and private sectors in the United States. Their report identified five general trends: (1) while at least one DSS fulfilled each criteria in the questionnaire used, no single system successfully addressed all important considerations; (2) ecological and management interactions across multiple scales were not comprehensively addressed by any of the systems evaluated; (3) the ability of the current generation DSSs to address social and economic issues lags far behind biophysical issues; (4) the ability to simultaneously consider social, economic, and biophysical issues is entirely missing from current systems; (5) group consensus-building support was missing from all but one system—a system that was highly dependent upon trained facilitation personnel. In addition, systems that offerered explicit support for choosing among alternatives provided decision makers with only one choice of methodology. The reviewers noted that little or no coordination had occurred between the 24 development teams, resulting in large, monolithic, stand-alone systems, each with a substantially different concept of the natural resource management process and how to support it.

Different DSSs appear to support different parts of the natural resource management process. Table III lists 33 DSSs, the 24 systems surveyed by Mowrer et al. plus nine DSSs not included in that study. Nineteen of the 33 are labeled full service DSSs at their scale of operation because they attempt to be comprehensive, offering or planning to offer support for a complete natural resource management process. These DSSs can be further classified by the scale of support which is their primary focus: regional assessments, forest planning, or project level planning. The remainder, labeled functional service modules, provide specialized support for one or a few phases of the entire natural resource management process. These service

Table III **A Representative Sample of Existing Ecosystem Management Decision Support Software for Forest Conditions of the United States Arranged by Operational Scale and Function**

Full service EM-DSS		Functional service modules	
Operational scale	**Models**	**Function**	**Models**
Regional assessments	EMDS	Group negotiations	AR/GIS
	LUCAS*		IBIS*
		Vegetation dynamics	FVS
Forest level planning	RELM		LANDIS
	SPECTRUM		CRBSUM
	WOODSTOCK		SIMPPLLE
	ARCFOREST		
	SARA	Disturbance simulations	FIREBGC
	TERRA VISION		GYPSES
	EZ-IMPACT*		UPEST
	DECISION PLUS*		
	DEFINITE*	Spatial visualization	UTOOLS/UVIEW
			SVS*
Project level planning	NED		SMARTFOREST*
	INFORMS		
	MAGIS	Interoperable system architecture	LOKI
	KLEMS		DCOM
	TEAMS		
	LMS*	Economic impact analysis	IMPLAN
		Activity scheduling	SNAP

*References for models can be found in Mowrer *et al.* (1997) and Rauscher (1999).

modules can be organized according to the type of functional support they provide: group negotiations, vegetation dynamics, disturbance simulation, spatial visualization, and interoperable system architecture.

A. Full Service Natural Resource Management DSSs

1. Regional Assessments

The EMDS system is designed to support ecological assessments, usually at regional, provincial, or watershed scales (Table III). It provides a general software environment for building knowledge bases that describe logical relations among ecosystem states and processes of interest in an assessment. Once these knowledge bases are constructed by users, the system provides tools for analyzing the logical structure and the importance of missing information. The EMDS provides a formal logic-based approach to assessment analysis that facilitates the integration of numerous diverse topics into a single set of analyses. It also provides robust methods for handling incomplete information. A variety of maps, tables, and graphs provide useful information about what data are missing, the influence of missing data, and how data are distributed in the landscape. It also provides support for exploring alternative future conditions.

The EMDS integrates a knowledge base engine into the ArcView (Environmental Systems Research Institute, Redlands, CA) geographic information system (GIS) to provide knowledge-based reasoning for landscape-level ecological analyses. Major components of EMDS include the NetWeaver knowledge base system, the EMDS ArcView application extension, and the Assessment system (Fig. 4).

Primary components of the EMDS Arcview application extension are the DataEngine and MapDisplay objects that customize the ArcView environment with methods and data structures required to integrate NetWeaver's knowledge-based reasoning schema into ArcView (Fig. 5). The Assessment system is a graphic user interface to the NetWeaver engine for end users of the EMDS application that controls setup and running of analyses, runtime editing of knowledge bases, and display of maps, tables, graphs, and evaluated knowledge base state related to analyses.

The NetWeaver knowledge base system is composed of a logic engine and a graphic user interface for

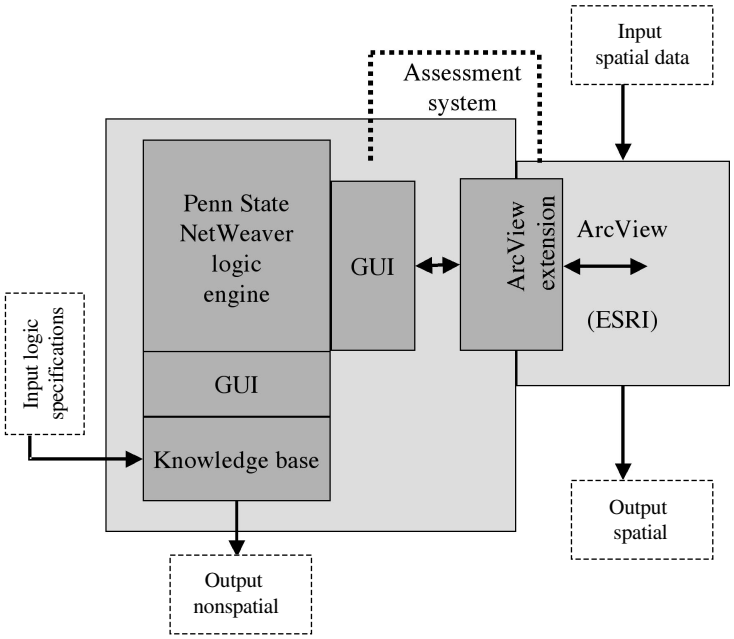


Figure 4 EMDS system component diagram. EMDS extends the functionality of the ArcView geographic information system (Environmental Systems Research Institute, ESRI) with logic modeling for landscape analysis. There are two graphical user interfaces (GUIs) to the NetWeaver logic engine used in EMDS: one GUI, external to the EMDS extension, is used in knowledge base development, while a second simpler one is integrated into the extension as an end-user interface.

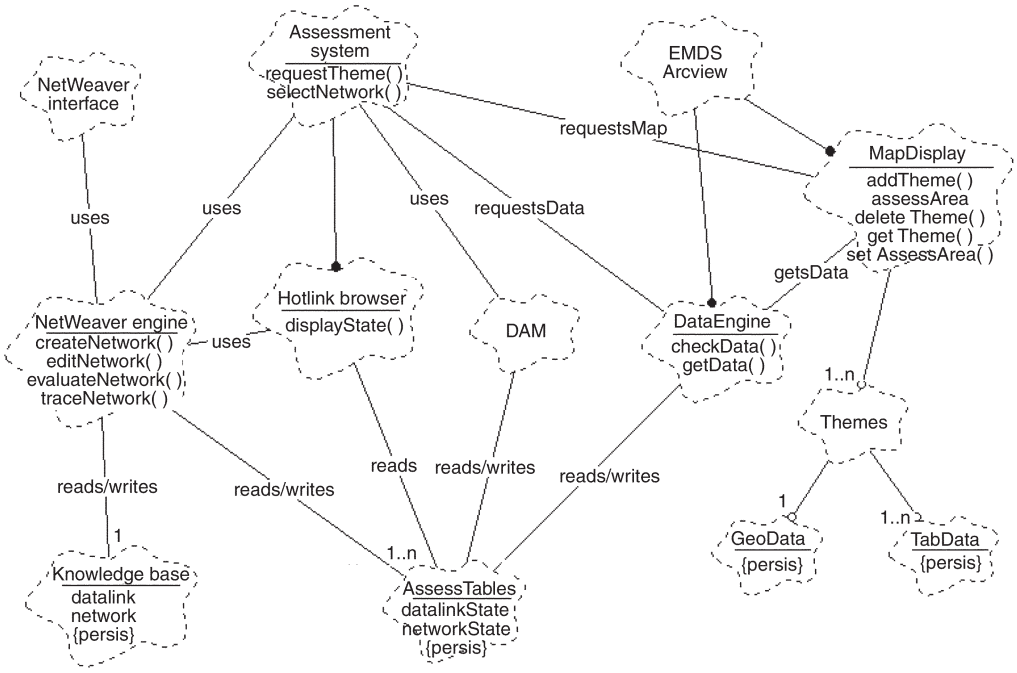


Figure 5 Object model diagram for the EMDS ArcView extension. Some objects are annotated with their key attributes and methods. Attributes are displayed as unqualified text, methods are indicated by "( )" after the method name. Objects with the annotation "[persis]" indicate persistent data structures stored in files. The Data Acquisition Manager is abbreviated as DAM.

knowledge-base developers that provide controls for designing, editing and interactively evaluating knowledge bases. NetWeaver initially was developed in 1988 by Dr. Michael Saunders of Pennsylvania State University. It has steadily evolved since its introduction, and now provides a substantially different form of knowledge representation that offers several critical advantages over production rule systems and make it highly suitable for landscape-level ecosystem analyses. Key features of the system include an intuitive graphical user interface, object-based logic networks of propositions, and fuzzy logic. Implementation of the user interface, together with NetWeaver's object-based representation, supports design of modular knowledge bases. Modularity in turn enables effective, incremental evolution of knowledge base structures from simple to complex forms. It has been asserted as an axiom of modern systems theory that incremental evolution is a requirement for design of complex systems.

The knowledge-based reasoning schema of NetWeaver uses an object- and fuzzy-logic-based propositional network architecture for knowledge representation. The system facilitates evaluation of complex, abstract topics such as water quality that depend on numerous, diverse subordinate conditions because NetWeaver is fundamentally logic based. The object-based architecture of NetWeaver knowledge bases is conducive to incremental, evolutionary design of complex knowledge representations, which has been recognized as crucial to successful design of complex systems. The architecture of a NetWeaver knowledge base allows both the ability to evaluate the influence of missing information and the ability to reason with incomplete information.

Use of fuzzy logic in NetWeaver affords significant practical advantages over Bayesian belief networks and classical rule-based knowledge representations that depend on bivalent (e.g., yes/no, true/false) or multivalent logic in the context of knowledge bases that are conceptually broad and that include a wide variety of topics. Bayesian belief networks work well on narrow, well-defined problems, and may be preferable to fuzzy logic networks when conditional probabilities of outcomes are known. However, Bayesian belief networks are difficult to apply to large, general problems because the number of conditional probabilities that must be specified can quickly become extremely large as the conceptual scope of a problem increases. In such situations, model design not only becomes very difficult to manage, but many of the probabilities will not be well characterized and will therefore need to be supplied by expert judgment,

thus negating much of the value to be gained by a statistically based approach to knowledge representation. Similarly, the number of rules required in a bivalent logic knowledge base increase to unmanageable levels as soon as the model designer attempts to account for shades of outcomes such as poor, fair, good, excellent, etc. These arguments should not be taken to infer that fuzzy logic networks are inherently superior to other forms of knowledge representation. On the contrary, the various methods just discussed may be highly complementary to one another. In particular, fuzzy logic networks are ideally suited as logical frameworks for integrating model results from a variety of analytical systems such as simulators, mathematical optimization, Bayesian belief networks, and rule bases.

Ecological and natural resource sciences have developed thousands of mathematical models to characterize specific relations among ecosystem states and processes. However, as either the breadth or depth of a problem domain increases, knowledge of interrelations among all relevant states and processes becomes progressively more qualitative. In part, this is a consequence of inevitable gaps in knowledge and in part this is a consequence of not having perfect knowledge of all possible dependencies among all relevant relations. Approximate, or fuzzy, reasoning, described in detail by Zadeh in 1992, significantly extends the ability to reason with the types of imprecise information typically found in natural resource science. Zadeh's early thoughts still are pertinent:

> . . . the ineffectiveness of computers in dealing with [biological] systems is a manifestation of what might be called the principle of incompatibility—a principle which asserts that high precision is incompatible with high complexity. Thus, it may well be the case that the conventional techniques of system analysis and computer simulation . . . are intrinsically incapable of coming to grips with the great complexity of human thought processes and decision-making. . . . Indeed, it is entirely possible that only through the use of [approximate reasoning] could computer simulation become truly effective as a tool for the analysis of systems which are too complex or too ill-defined for the application of conventional quantitative techniques.

Application of fuzzy logic to natural resource science and management is still relatively new. General areas of application include classification in remote sensing, environmental risk assessment, phytosociology, geography, ecosystem research, and environmental assessment. More specific applications include catchment modeling, cloud classification, evaluation of plant nutrient supply, soil interpretation, and land suitability for crop production.

2. Forest Level or Strategic Planning

Forest level planning corresponds to the strategic planning process of decision science as described by Holsapple and Whinston in 1996. Many federal agencies, including the USDA Forest Service, have relied on linear programming systems of various kinds as the primary strategic planning decision support methodology. In 1979, a linear programming, harvest scheduling model was turned into a forest level planning too, FORPLAN, and until 1996 all national forest supervisors were required to use it as the primary analytical tool for strategic forest planning. After 17 years of increasingly fierce criticism that the normative, rational, optimization approach to decision analysis implemented by FORPLAN and its successor, SPECTRUM, was not adequate, the Forest Service finally removed its formal requirement to use FORPLAN/SPECTRUM. The specifics of the arguments critical of FORPLAN/SPECTRUM as an analytical tool for forest planning are beyond the scope of this paper and can be readily found in Barber and Rodman and Hoekstra and colleagues.

Forest level planning may be more successfully performed using soft, qualitative decision analysis formalisms than the hard, quantitative methods employed in rational, linear or nonlinear optimization schemes. Many other decision analysis formalisms exist along with the tools that make them useful and practical (Table III). A number of these techniques may offer greater support for dealing with power struggles, imprecise goals, fuzzy equity questions, rapidly changing public preferences, and uneven quality and quantity of information. In particular, DECISION PLUS and DEFINITE are well-developed and tested analysis tools for forest planning that use judgment-based, ordinal, and cardinal data to help users characterize the system at hand and explore hidden interactions and emergent properties.

A forest plan should demonstrate a vision of desired future conditions. It should examine current existing conditions and highlight the changes needed to achieve the desired future conditions over the planning period. Finally, the forest plan should demonstrate that recommended alternatives actually lead toward desired future conditions by tracking progress annually for the life of the plan. The forest plan should be able to send accomplishable goals and objectives to the level of project implementation and receive progress reports that identify the changes in forest

conditions that management has achieved. Ideally, all competitors in this class of DSS should be objectively evaluated for their effectiveness in supporting these tasks, their ease of use in practice, and their ability to communicate their internal processes clearly and succinctly to both decision makers and stakeholders. Such an evaluation has not yet been conducted.

3. Project Level Implemention or Tactical Planning

Forest plans are programmatic in that they establish goals, objectives, standards, and guidelines that often are general. The public and USDA Forest Service personnel have flexibility in interpreting how forest plan decisions apply, or can best be achieved, at a particular location. Forest plans typically do not specify the precise timing, location, or other features of individual management actions. Decision support systems at the project level assist the user to identify and design site-specific actions that will promote the achievement of forest plan goals and objectives. For example, a strategic level forest plan might assign a particular landscape unit for management of bear, deer, and turkey with minimum timber harvesting and new road construction, and no clearcutting. The tactical, project implementation plan would identify specific acres within that management unit that would receive specific treatments in a specific year. Project level DSSs have been developed to support the tactical level planning process.

NED is an example of a project-level DSS that implements a goal-driven, multiobjective decision analysis process (Fig. 6). Because management is defined to be a goal-driven activity, goals must be defined before appropriate management actions can be determined. It cannot be overemphasized that without goals, management cannot be properly practiced. Goal-driven systems, such as NED, assist the user in creating an *explicitly defined goal hierarchy.*

A goal is an end state that people value and are willing to use allocated resources to achieve or sustain. Goals form a logical hierarchy with the ultimate, all-inclusive goal at the top, subgoals at the various intermediate levels, and a special goal, which may be called a *desired future condition* (DFC), at the bottom. A DFC is a goal statement containing a single variable measuring some observable state or flow of the system being managed. They are the lowest level of the goal hierarchy and are directly connected to the management alternatives being considered. Furthermore, DFCs precisely define the measurable variables that each alternative must contain. In other words, each

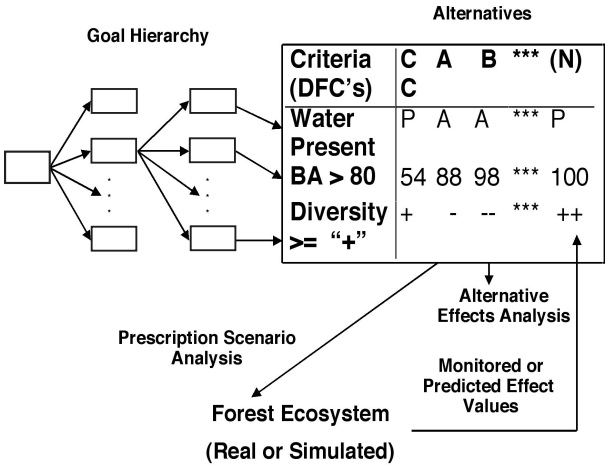


Figure 6 A diagrammatic description of the NED goal-driven decision analysis process.

DFC provides a measure of the degree to which any given ecosystem state, current or simulated future, meets the goal statement. For example:

```
Goal: Freshwater Fishing
   Opportunities Exist IF
  DFC(1): pH of all (>=(90%)
    freshwater lakes > 5 AND
  DFC(2): popular game fish are
    plentiful AND
  DFC(3): access to all (>=90%)
    freshwater lakes is adequate.
```

Given that "plentiful" and "adequate" are further defined so that they are measurable, the three DFCs above define three attributes, i.e., pH, popular game fish, and access to freshwater lakes, that each alternative under consideration must have. Otherwise, it will be impossible to determine whether the alternative can satisfy the goal "Freshwater Fishing Opportunities Exist." Desired future conditions should be objective in nature. That is, there should exist a commonly understood scale of measurement, and the application of that scale to the ecosystem should yield roughly the same results no matter who makes the measurement. DFC(1), above, is objective in this sense. On the other hand, DFC(2) and DFC(3) are using subjective, binary scales for "plentiful" and "adequate." Research results indicate that subjectively developed scales can be reliably used by qualified professionals. An operationally practical DFC should (1) provide the appropriate information on theoretical grounds and (2) also be obtainable. In other words, a DFC should accurately define the lowest level subgoal and be measurable in practice.

Unlike goals, which depend primarily on value judgments, defining DFCs requires primarily factual knowledge. A set of DFCs that define a lowest level goal is not generally unique. There are usually alternative ways to define the same lowest level goal, and no *a priori* tests exist to show that one way is better or worse than another. This disparity typically results from competing scientific theories or professional judgment. Consequently, it is important to document the justification for defining the lowest level goal in any particular way.

Constraints, like goals, have a standard that is either met or not. This standard is meant to screen unacceptable goals, objectives, alternatives, and management prescriptions from consideration. Requirements are equivalent to constraints, but are phrased differently. Permissions are the logical inverse of constraints. Permissions make it explicitly clear that certain goals, objectives, alternatives, and silvicultural prescriptions are allowed if they naturally surface in the management process. Permissions are not mandatory; if they were, they would be requirements.

Alternatives are the courses of action open to a decision maker for satisfying the goal hierarchy. In natural resource management, each alternative contains a set of ACTION-LOCATION-TIME triples that is intended to change the landscape so that goal satisfaction is improved. These triples, called prescriptions, embody the purposeful application and expenditure of monetary, human, material, and knowledge resources that define natural resource management.

The design of alternatives, like the design of the goal hierarchy, is largely an art that relies heavily on decision science expertise along with an expert-level understanding of natural resource management. It is also very much an iterative process. High-quality decisions require the design of a set of promising, distinct alternatives to evaluate. Given knowledge about (1) the current condition of the forest ecosystem, (2) the goals and DFCs, (3) the standards, guides, best management practices, and constraints in force at any given time, and (4) available management prescriptions, an experienced manager can craft alternatives that represent reasonable answers to the question: What state of organization do we want for this forest ecosystem so we can best meet the goals? The human mind is the sole source of alternatives.

B. Functional Service Modules

The full service EM-DSS relies on specialized software service modules to add a broad range of capabilities (Table III). Tools to support group negotiation in the decision process are both extremely important and generally unavailable and underutilized. AR/GIS is the most fully developed software available for this function. IBIS, another group negotiation tool, is an issue-based information system that implements argumentation logic (the logic of questions and answers) to help users formally state problems, understand them, clearly communicate them, and explore alternative solutions. Vegetation dynamics simulation models, both at the stand and at the landscape scale, provide EM-DSS with the ability to forecast the consequences of proposed management actions. Disturbance models simulate the effects of catastrophic events such as fire, insect defoliation, disease outbreaks, and wind damage. Models that simulate direct and indirect human disturbances on ecosystems are not widely available. Although, models that simulate timber harvesting activities exist, they provide little, if any, ecological impact analyses such as the effect of extraction on soil compaction, on damage to remaining trees, or on the growth response of the remaining tree and understory vegetation. Models that simulate the impact of foot traffic, mountain bikes, and horseback riding on high-use areas are largely missing. Models that simulate climate change, nutrient cycling processes, acid-deposition impacts, and other indirect responses to human disturbance exist but are rarely practical for extensive forest analyses. Stand and landscape level visualization tools have improved dramatically in the last few years. It is now possible, with relatively little effort, to link to and provide data for three-dimensional stand level models such as SVS as well as landscape level models such as UVIEW and SMARTFOREST.

The next two sections of this article illustrate how two very different DSSs, EMDS and NED, can be used to support natural resource management. Both EMDS and NED have been described in Section III, above. Section IV deals primarily with illustrating more clearly how they are used.

IV. ASSESSING WATERSHED CONDITION WITH EMDS

The United States Environmental Protection Agency (EPA) Office of Research and Development and the Forest Service (United States Department of Agriculture) have cooperatively developed a new analytical technique for watershed assessment, using knowledge-based processing of landscape databases that enable environmental managers to make better decisions.

Much of the knowledge-base design concerns assessment of stream characteristics to provide decision support for the Total Maximum Daily Load (TMDL) program of the United States EPA. Section 303(d) of the Clean Water Act, identifies sources of pollution remaining after end-of-pipe discharges are regulated and best available technology has been applied. Remaining sources of pollutants are termed nonpoint sources (NPS). Under requirements of the act, states develop lists of waters that do not meet state water quality standards, even after point sources of pollution have installed required levels of pollution control technology. States must establish priority rankings based on severity of pollution and beneficial uses of water bodies, such as recreation or fishing, and must develop TMDLs for waters on the lists. TMDLs specify amounts of pollutants that need to be reduced to meet state water quality standards and allocate pollution control responsibilities among pollution sources in a watershed.

The NPS TMDL parameters critical to aquatic ecosystems include streamflow, stream temperature, nutrients, stream sediment, and in-channel stream habitat. Unlike point source TMDLs, NPS parameters vary across watersheds due to physiographic and biogeoclimatic differences among watersheds. In particular, loadings for NPS TMDLs depend on cumulative changes in land use and land cover patterns, and subtle variations in climate and weather patterns. Vulnerability of water bodies to NPS impairment is thus a function of the physiographic characteristics of watershed and the extent and spatial pattern of landscape-scale stressors that alter these characteristics.

Conventional TMDL development is carried out on individual stream reaches by analyzing impaired conditions. Conventional methods for TMDL analysis cannot address the spatial and temporal scales required to: (1) establish adequate reference conditions for NPS parameters; (2) project likely scenarios of water quality change due to changes in land use, cover, or climate; (3) relate monitoring technologies and standards to defined ecoregional scales; and (4) establish schedules for TMDL development that are ecologically meaningful and compatible with federal agency responsibilities under the Endangered Species Act.

The objectives of this particular EMDS application were to design a knowledge base as a logical framework for assessment of 6th-code watershed (a hydrologic basin, roughly 10,000–20,000 ha in the western United States) condition and illustrate its application in landscape analysis. Given the objectives of the EPA water quality assessment program, the primary knowledge base topics included in design were watershed processes, watershed patterns, general effects of human influence, and specific effects of human influences on aquatic species. A key decision, made early in the design process, was that the method of assessment should be sufficiently general that the knowledge base could be applied in any geographic region of the United States. This design criterion was implemented by constructing all fuzzy membership functions as dynamically defined functions of data representing standards. That is, all fuzzy membership functions in the knowledge base are defined by standards input during analysis. All data are evaluated by comparison to standards for which we conceptually distinguish three basic types: reference conditions representing attributes of unmanaged watersheds, management standards set by resource management agencies such as the USDA Forest Service, and regulatory standards set by regulatory agencies such as the EPA.

A. Spatial and Temporal Contexts

Knowledge bases for landscape assessment frequently have implied spatial and temporal resolutions. Given the nature of the problem domain discussed above, the most practical landscape unit for analysis of watershed condition was considered to be the 6th-code watershed. The authors selected this level of spatial resolution primarily because watershed databases for this scale would typically contain detailed information about valley bottom characteristics relevant to TMDL assessment. The problem domain of a knowledge base also frequently implies a temporal context for analysis. Careful definition of temporal context is particularly important for knowledge bases that contain networks for evaluating effects of proposed management. Definition of temporal resolution was not needed in the current application because the problem domain of the knowledge base was only concerned with assessment of existing conditions.

B. Knowledge Elicitation

Design began with identification of the primary networks needed to evaluate watershed condition. Collectively, the set of primary logic networks can be construed as the logical dimensions along which watershed condition is evaluated. Knowledge base design generally proceeded in a top-down manner. For a given network, the authors first developed a concise statement of its proposition, and then enumerated the set of premises needed to evaluate the truth of the proposition for the network, and the logical construction of those premises.

The nature of each identified premise was discussed to make a preliminary determination of whether the premise was simple and representable by evaluation of data, or whether the nature of the premise was more complex, having its own set of premises, and therefore requiring representation by another logic network.

C. Knowledge-Base Structure

The primary logic networks for assessing watershed condition are watershed processes, watershed patterns, human influence, and aquatic species. Each logic network evaluates a specific proposition about the state of watershed condition (Table IV). A simplified hierarchy of the logic structure under the network for watershed processes illustrates the scope of the watershed processes topic (Table V). Topic structure has been simplified for brevity.

We trace part of the logic structure from watershed processes down to total yield as a typical example of knowledge base structure. The truth of the proposition that watershed processes are within suitable ranges of conditions depends on the degree to which the premises, or logical antecedents, of watershed processes are true. The logic structure of the network (not shown) makes the meaning of the proposition (Table V) explicit. The logic network for hydrologic processes similarly has logically antecedent conditions, represented by networks that determine the truth of its proposition. Determination of suitable hydrologic processes depends, among other things (Table V), on suitable stream flow. Evaluation of the stream flow network depends on evaluation of four other networks, but whereas the logic network for hydrologic processes is evaluated by a fuzzy logic operator (AND), stream flow is evaluated by calculating a sum of products. The difference in formulations is semantically significant. The computation of the stream flow sum calculation effectively asserts that networks contributing to evaluation of stream flow can compensate for one another to some extent. If, for example, the logic network for total yield evaluates to false (truth value = -1), but some other network, say peak flow, evaluates to partially true (a truth value between -1 and 1), then peak flow at least partially compensates for total yield.

D. Example EMDS Analysis

An example analysis of erosion processes in a small portion of the Columbia River Basin (northwestern United States) illustrates landscape application of the

Table IV Primary Networks in the EMDS Knowledge Base for Assessing Watershed Condition

Network name	Proposition evaluated by network
Watershed processes	Watershed processes are within acceptable ranges
Hydrologic processes	Hydrologic processes in the watershed are within acceptable ranges compared to reference conditions (Table V)
Erosion processes	Erosion processes in the watershed are within an acceptable range compared to reference conditions (Table V)
Fire processes	Fire processes in the watershed are in good condition compared to reference conditions (Table V)
Watershed patterns	Watershed patterns are within acceptable ranges
Upland patterns	Upland patterns in the watershed are within acceptable ranges compared to reference condition; evaluation includes vegetation composition and structure
Valley bottom patterns	Valley bottom processes in the watershed are within acceptable ranges compared to reference conditions; evaluation includes vegetation composition and structure, stream type composition, sinuosity, woody debris, pool frequency, bank stability, and sediment transport capacity
Channel patterns	Channel characteristics in the watershed within acceptable ranges compared to reference conditions; evaluation includes bankfull width to depth ratio, pool depth, floodplain width, and fines in riffles
Human influence	Aggregate effects of human influence are within acceptable ranges compared to management standards, evaluation includes effects of roads, dams, diversions, channelization, groundwater extraction, mines, grazing, and recreation
Aquatic species	Likelihood of long-term viability of aquatic species is good.
Fish habitat	Fish habitat potential in watershed is good, evaluation includes effects of baseflow, substrate, water temperature, and cover.

Table V Propositions Associated with EMDS Networks Antecedent to the Watershed Processes Network

Network name	Proposition	Source of comparison[a]
Hydrologic processes	Hydrologic processes are within suitable ranges	
Stream flow	Stream flow characteristics are within suitable ranges	
Total yield	Total water yield is within a suitable range	Reference
Peak flow	Peak flow is within a suitable range	Reference
Base flow	Base flow is within suitable range	Reference
Bankfull discharge	Bankfull discharge is within suitable range	Reference
Water quality	Attributes of water quality are within suitable ranges	
Sediment	Sediment attributes are within a suitable range	
Bedload	Bedload is within a suitable range	Reference
Dissolved solids	Concentration of dissolved solids is within a suitable range	Reference
Suspended solids	Concentration of suspended solids is within suitable range	Reference
Coliform	Coliform count is within a suitable range	Regulation
Dissolved O_2	Concentration of dissolved oxygen is within a suitable range	Regulation
Water temp	Water temperature characteristics are within suitable ranges	
Temp max	7-day running average for summer maximum water temperature is within within a suitable range	Regulation
Temp thresh.	Number of days that daily maximum water temperature exceeds threshold is within a suitable range	Regulation
Nutrients	Nutrient concentrations are within suitable ranges	
Nitrogen conc.	Nitrogen concentration is within a suitable range	Regulation
Phosphorous conc.	Phosphorous concentration is within a suitable range	Regulation
Metals	Metal concentrations are within suitable ranges of regulatory requirements	
Aluminum conc.	Aluminum concentration is within a suitable range	Regulation
Arsenic conc.	Arsenic concentration is within a suitable range	Regulation
Copper conc.	Copper concentration is within a suitable range	Regulation
Mercury conc.	Mercury concentration is within a suitable range	Regulation
Zinc conc.	Zinc concentration is within a suitable range	Regulation
Erosion processes	Erosion processes are within suitable ranges	
Surface erosion	Amount of surface erosion is within a suitable range	Reference
Mass wasting	Amount of mass wasting is within a suitable range	Reference
Debris avalanche	Amount of debris avalanche is within a suitable range	Reference
Sediment delivery	Amount of sediment delivery is within a suitable range	Reference
Fire processes	Fire processes are within suitable ranges	
Fire frequency	Fire frequency is within a suitable range	Reference
Fire hazard	Amount of expected fire damage is within a suitable range	Reference

[a]Observed data values are compared to fuzzy membership functions, representing either reference conditions or regulatory requirements, to determine if an observed value falls within a suitable range of values. Data defining fuzzy membership functions for reference conditions and regulatory requirements are read by the knowledge base to parameterize the fuzzy membership functions.

knowledge base for watershed condition in EMDS. The Assessment system was used to specifically select the erosion processes network for evaluation in this example. In general, the Assessment system can be used to select any combination of networks for analysis. Map output shows the computed truth of the proposition that erosion processes are within a suitable range of conditions for each 6th-code watershed in the assessment area selected for this example (Fig. 7).

Partial evaluations, based on currently available data, can be performed in EMDS. Truth values for erosion processes in the map output (Fig. 7) only reflect a partial evaluation of the network because data values for volumes of mass wasting and debris avalanche are missing in this example. Ecological assessments frequently are broad in conceptual scope, requiring evaluation of possibly numerous and diverse data. Consequently, several to many data elements needed for complete evaluation of a knowledge base or any of its components may be missing at the start of an assessment. In this example, complete evaluation of erosion processes requires data values

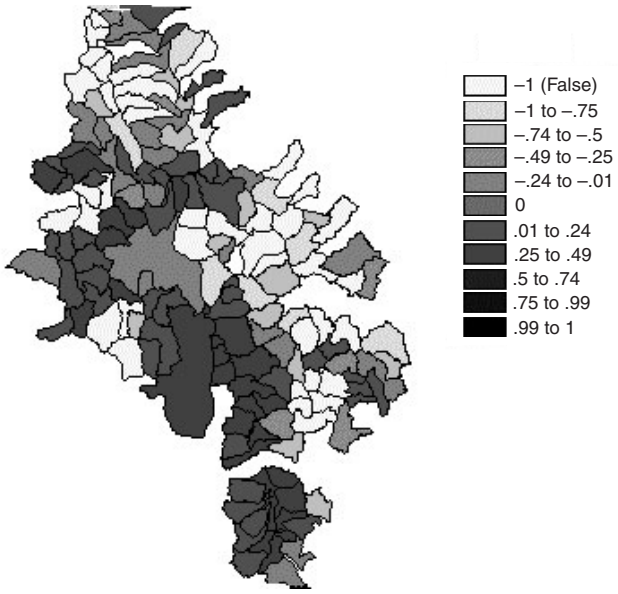


Figure 7 Map of truth values for evaluation of erosion processes logic network.

for volumes of surface erosion, sediment delivery to streams, mass wasting, and debris avalanche, but only the first two data elements were available at the time of analysis. However, given the set of knowledge-base objects and their logical organization within the knowledge base, the NetWeaver engine computes the relative influence of missing data.

Although data on volumes of mass wasting and debris avalanche were not available at the time of analysis, logic processing still allowed useful summarization of erosion conditions given available data. Seventy-three watersheds had truth values for erosion processes near 0.25, indicating some positive evidence favoring the conclusion of suitable erosion processes in these watersheds. Perhaps even more significantly, 38 watersheds evaluated as being in a completely unsuitable condition with respect to erosion processes, although the knowledge base only had partial data about these watersheds as well. Even though data on these 38 watersheds were incomplete, the data that were available were sufficient to reach a conclusion of complete unsuitability, given the values of those data.

Finally, the Hotlink browser (center of Fig. 5) provides a way to examine details underlying results of an analysis, allowing the user to view the evaluated state of the knowledge base for any landscape feature selected on a map (Fig. 8). The top pane of the browser window displays an expandable, hierarchical outline of the knowledge structure (or component thereof, if only a portion was selected for analysis as in this example). The logic specification of a network highlighted in the outline pane is displayed in the lower pane. Network objects in the specification pane are color coded with their state for fast interpretation of graphic displays, and actual values of objects are reported in the status bar at the bottom of the browser window when the mouse pointer is positioned over them.

E. EMDS Case Study Discussion

The knowledge base for evaluation of watershed condition was designed for general application. The architecture is such that the knowledge base should be applicable in any geographic region with no more than minor adaptation. Specification of standards as data to be read from a database is an important ingredient of this general applicability. Clearly, this approach to a general solution begs the question: Where do specifications for reference conditions come from? We suggest the following approach. For any geographic region or subregion, the vegetation potential of unmanaged watersheds is conditioned by geographic and

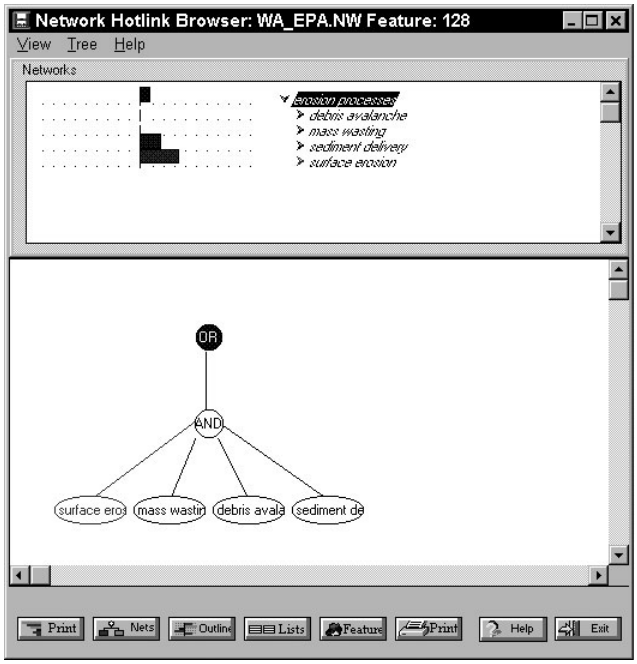


Figure 8 The EMDS Hotlink Browser is used to navigate the structure of an evaluated knowledge base. The upper panel of the browser displays the knowledge base structure in outline form. Logic specifications of components selected in the upper panel are displayed in the lower panel. Color coding and pop-up text windows (not illustrated in the figure) display the evaluated state of components.

climatic factors. Widely available synecological analysis tools provide a basis for arranging watersheds along geographic and climatic gradients, and identifying groupings indicative of reasonably separable vegetation potentials in the absence of management. Most resource management agencies have sufficiently detailed GIS coverages to identify watersheds that have experienced little or no management, and the attributes of such watersheds can be used as reference conditions.

The knowledge base for assessing watershed condition evaluates data in relation to standards by comparing current values to a set of four x values, Q1, Q2, Q3, and Q4 with fuzzy memberships of -1, 1, 1, and -1, respectively. Points Q1 and Q4 define extreme conditions for a standard that indicate unsuitable conditions when the data value is $\leq$Q1 or $\geq$Q4. Data values in the closed interval [Q2, Q3] indicate that the current condition is completely suitable, and the open intervals (Q1, Q2) and (Q3, Q4) indicate ranges in which the current data value has partial membership in the fuzzy set defined by the four points. Collectively, the set of four points define a trapezoidal membership function. The four-point method is flexible because Q1–Q4 may equally well define points on an absolute scale, a relative scale (e.g., 0–100), a ratio scale measuring departure from a ratio of 1 (current data value divided optimum reference value), or an ordinal scale based on subjective judgments such (e.g., 0 = completely unacceptable, 1 = slightly acceptable, etc.).

The EPA has established a five-step approach to setting NPS TMDLs: (1) identify waters requiring TMDLs; (2) priority ranking and targeting; (3) develop TMDLs; (4) implement control actions; and (5) assess control actions. Used in the mode of an initial landscape assessment, the knowledge base directly addresses decision needs in steps 1 and 2. Used in an evaluation mode, the knowledge base application is equally suitable for step 5. The example analysis is quite small, being limited to evaluation of only a small subset of topics in the complete knowledge-base. Nevertheless, analysis results suggest that a knowledge-based approach to landscape analysis for TMDL assessment generally is feasible because the conduct of a comprehensive analysis differs very little in principle from the small example given.

The complete knowledge base has large data requirements, but any combination of logic networks, representing subsets of the full knowledge base, may be selected for analysis. Key advantages of a landscape analysis based on fuzzy logic networks as implemented in NetWeaver and used in EMDS include the ability to reason with incomplete information, and the ability to evaluate the influence of missing information. Fuzzy-logic based landscape analysis may be most useful for construction of logical frameworks within which a wide variety of analytical results can be effectively integrated into a single, coherent analysis.

V. MANAGING THE BENT CREEK EXPERIMENTAL FOREST USING NED

The Bent Creek Experimental Forest is an approximately 2400-ha watershed which is part of Pisgah Ranger District of the Pisgah National Forest near Asheville, NC. The experimental Forest is a management unit containing 65 forest stands. The stand is the essential unit of forest management. It is a contiguous group of trees sufficiently uniform in species composition, arrangement of age classes, and growing condition that it can be distinguished as a management unit. Most of these stands, 67% by area, are in the oak (*Quercus* spp.) forest type with 19% in the yellow poplar *(Liriodendron tulipifera)* forest type. The small sawtimber size class represents 76% of the forest area. Large sawtimber represents 19% and the pole size class another 5%. None of the 65 stands are classed as regeneration. The average basal area is 27.3 square miles/ha (range: 10–54). The average number of trees per hectare is 410 (range: 69–2937) and the average quadratic mean diameter is 31.2 cm (range: 12.2–65.5 cm).

Five goals were selected for the Bent Creek Experimental Forest in order to illustrate the NED project-level natural resource management process. They represent five domains of interest: visual quality, ecology, timber production, water, and wildlife. For the sake of brevity and clarity, we selected only five goals of the many possible for this illustrative example. This small number of goals should not in any way be viewed as a limitation of the proposed natural resource management process.

The first goal, Large-scale Variety, is from the visual quality domain. It was one of a number of goals identified by a comprehensive literature review on measurable user preferences for visible aspects of forest environments. Large-scale variety within a forested area is obtained by creating a few medium- to large-sized openings that provide the desired variety when viewed from an overlook. These openings also provide variety over time and are perceived as incremental changes that occur as individual stands change in age and vegetation character. This goal would prevent the creation and maintenance of unbroken, relatively homogeneous forest areas.

```
Goal-1: Visual Quality Domain
Large Scale Variety is attained IF
  DFC-1: The Number of Stand Size
    Classes >= 3; AND
  DFC-2: Openings < 67% of the
    forest area; AND
  DFC-3: % of Area in each Stand
    Size Class <= 50%.
```

The second goal, Local Biological Diversity, represents the ecological domain. Interest in enhancing local biological diversity stems from a desire to manage the forest for plant and animal species richness; to preserve or establish plants, plant associations, and habitats that are unique to this local area. Direct measurements of species richness are possible, however, in practice such measurements are seldom available. An alternative way to define local biological diversity is in terms of the state of the forest landscape that would enhance biological diversity. A wide variety of vegetative conditions is likely to perpetuate the maximum number of plant species and provide habitat for the maximum number of animal species. Therefore, the local biological diversity goal is defined as follows:

```
Goal-2: Ecological Resource Domain
Local Biological Diversity is attained
IF
  DFC-1: % Stands in Large & Small
    Sawtimber > 20%; AND
  DFC-2: % Stands in Sapling & Pole
    Size Classes > 10%; AND
  DFC-3: % Stands in Regeneration Size
    Class > 10%; AND
  (DFC-4: Openings > 5%; OR
  DFC-5: Water is Present).
```

The third goal, Continuous Quality Sawtimber Production, represents the timber commodity domain. The tree species that make up the southern Appalachian hardwood forests are particularly well suited to the production of large, high-quality sawtimber. Unlike the previous two goals, the third goal must be defined both at the management unit level and at the individual stand level.

```
Goal-3: Timber Resource Domain
Continuous Quality Sawtimber
  Production is attained IF
  Management Unit Level:
  DFC-1: % Stands in Large Sawtimber
    >= 10% & <= 15%; AND
  DFC-2: % Stands in Small Sawtimber
    >= 25% & <= 35%; AND
  DFC-3: % Stands in Sapling & Pole
    Size Classes >= 35 & <= 45%; AND
  DFC-4: % Stands in Regeneration Size
    Class > 5% & <= 10%;

  Stand Level:
  DFC-5: Relative Density >= 60 and
    < 100; AND
  DFC-6: Basal Area of Acceptable
    Growing Stock >= 6.9 m2 /ha; AND
  DFC-7: Basal Area of High Value
    Species >= 6.9 m2/ha; AND
  DFC-8: Basal Area of Commercial
    Species >= 11.5 m2/ha
```

Desired future conditions for the management unit-level test for the existence of a balanced size class distribution throughout the forest in order to provide a continuous supply of sawtimber products. Desired future conditions for the stand-level test that stands themselves are well stocked with trees of the appropriate species and quality.

The fourth goal, Limit Peak Flows, represents the water management domain. The goal to limit peak flows is focused on reducing erosion, silting, and flooding in the watershed by concentrating on the sensitive riparian zone stands.

```
Goal-4: Water Resource Domain
Limit Peak Flows is achieved IF
  Management Unit Level:
  DFC-1: % Openings <25%; AND DFC-2:
    Riparian Stands must meet all
    stand level DFCs.

  Stand Level:
  DFC-3: Relative Density >= 70; AND
  DFC-4: % Basal Area of evergreen
    trees > 0 m2/ha; AND
  DFC-5: Canopy Closure > 25%.
```

Finally, the fifth goal, Black Bear, represents the wildlife management domain. This goal is designed to create and/or enhance habitat for black bear.

```
Goal-5: Wildlife Resource Domain
Habitat for black bear is achieved IF
  Management Unit Level:
  DFC-1: > 30% of Stands must meet
    stand level DFCs.

  Stand Level:
  DFC-2: Coarse woody debris > 3.5
    m3/ha; AND
  (DFC-3: Soft mast producing trees
    are present; OR
  DFC-4: Hard mast producing trees
    are present).
```

These five goals and their DFCs define a formal goal hierarchy for the project-level natural resource management process for Bent Creek Experimental Forest. This formal goal hierarchy explicitly and clearly defines a logical relationship between the top-level goal of successfully managing Bent Creek Experiment Forest and the five subgoals introduced above. In order to achieve or maintain any one subgoal, we need to satisfy each of its defining DFCs. The DFCs are defined in terms of variables that can be measured either in the real forest ecosystem as it currently exists or as it is forecast to exist in the future. Monitoring and evaluation can determine whether meeting defined DFCs does achieve a goal or if new DFCs are needed.

A. Current Condition Analysis

Given a goal hierarchy and a description of the current condition of the Bent Creek Study Area, usually supplied by the monitoring process, we can assess how well the current condition meets our desired condition. This current condition analysis allows us to understand how near or far the forest ecosystem currently is from achieving our goals. It also yields the knowledge we need to decide how to change the forest to better meet our goals. We seek this understanding in order to design a rich and interesting set of alternative courses of action. The NED Natural Resource Management Decision Support System, NED for short, has been specifically developed to perform this function.

It is convenient to treat the current condition of the forest ecosystem as if it were an alternative. The current condition "alternative" is defined by the DFC measurement component precisely like any other alternative. The current condition "alternative" is only different from other alternatives because it has no prescription component. The current condition is a very useful reference condition against which other alternatives can be compared.

It is important to notice that once the goal hierarchy has been defined, the current and future condition analysis assumes a "closed-world" situation. A *closed-world assumption* means that all knowledge about the goals and their DFCs are present in the database. The desirability or undesirability of any and all changes of states of the forest landscape can only be evaluated with reference to the goal hierarchy. The *means* used to implement these state changes can be debated and evaluated with reference to the current constraint network; but that is a separate issue. If it turns out that there are hidden goals that surface, then they need to be added to the goal hierarchy along with their defining DFCs. Such a change in the goal hierarchy simply creates a new closed world of goals against which current or future forest landscapes can be evaluated. This convention is a necessary condition of this process and is one of the powerful concepts that make this approach to natural resource management relatively simple to understand.

The current condition analysis of Bent Creek results in the finding that goal G-1, Large Scale Variety, is rated "not satisfied" (Table VI). The goal completion report in the NED DSS presents the following facts:

```
Goal-1: Visual Quality      Not Satisfied
  Domain
Large Scale Variety
  is satisfied IF
  DFC-1: The Number of      Minimally
    Stand Size Classes        Satisfied
    >= 3; AND                 (Value=3)
  DFC-2: Openings < 67%     Satisfied
    of the forest             (Value=0)
    area; AND
  DFC-3: % of Area in       Not Satisfied
    each Stand Size           (Value=76)
    Class <= 50%.
```

"Value = 3" for DFC-1 means that the actual number of stand size classes in the current condition equals three. Therefore, DFC-1 is minimally satisfied, i.e., it is just barely satisfied. From this analysis, it is evident what the problem is. Bent Creek Experimental Forest does not satisfy the goal "large scale variety" because more than 50% of the area, in fact 76%, is in one stand size class—the small sawtimber size class. Bent Creek is too homogeneous to satisfy this goal.

Goal G-2, Local Biological Diversity, is rated "not satisfied" (Table VI). The goal completion report in the NED DSS provides the following facts:

```
Goal-2: Ecological          Not Satisfied
  Resource Domain
Local Biological Diversity is
  satisfied IF
  DFC-1: % Stands           Satisfied
    in Large &                (Value=95)
    Small Sawtimber
    > 20%; AND
  DFC-2: % Stands           Not Satisfied
    in Sapling &              (Value=5)
    Pole Size
    Classes > 10%;
    AND
  DFC-3: % Stands in        Not Satisfied
    Regeneration              (Value=0)
```

```
    Size Class
    > 10%; AND
  (DFC-4: Openings
    > 5%; OR
  DFC-5: Water is          Satisfied
    Present).                (Value=
                             present)
```

It is easy to see that Local Biological Diversity is not satisfied because the size class distribution of the forest is skewed toward the large-sized forest stands.

Goal G-3, Continuous Quality Sawtimber Production, is rated "not satisfied" (Table VI). The goal completion report in the NED DSS provides the following facts:

```
Goal-3: Timber             Not Satisfied
  Resource Domain
Continuous Quality
  Sawtimber
  Production is
  attained IF
  Management Unit Level:
  DFC-1: % Stands          Not Satisfied
    in Large                 (Value=19)
    Sawtimber >= 10%
    & <= 15%; AND
  DFC-2: % Stands in       Not Satisfied
    Small Sawtimber          (Value=76)
    >= 25% & <= 35%;
    AND
  DFC-3: % Stands in       Not Satisfied
    Sapling & Pole           (Value=5)
    Size Classes
    >= 35 & <= 45%
    AND
  DFC-4: % Stands in       Not Satisfied
    Regeneration             (Value=0)
    Size Class
    > 5% & <= 10%;
```

At the management unit level, Bent Creek does not have a balanced size class distribution that would provide a sustainable supply of high quality sawtimber. Unlike the previous goals, this goal is also defined at the stand level.

```
Stand Level:
DFC-5: Relative Density >= 60 and
  < 100; AND
DFC-6: Basal Area of Acceptable
  Growing Stock >= 6.9 m2/ha; AND
DFC-7: Basal Area of High Value
  Species >= 6.9 m2/ha; AND
DFC-8: Basal Area of Commercial
  Species >= 11.5 m2/ha
```

Using the NED DSS to examine the goal report for each of the 65 stands in the management unit, it becomes clear that stands fail to satisfy the stand conditions for two major reasons: either they are below or above the relative density range in DFC-5 or they do not have enough high value species (DFC-7). In contrast, DFC-6 and DFC-8 rarely make a stand unsatisfactory at Bent Creek.

Goal G-4, Limit Peak Flows, is rated "not satisfied" (Table VI). The goal completion report in the NED DSS provides the following facts:

```
Goal-4: Water              Not Satisfied
  Resource Domain
Limit Peak Flows
  is achieved IF
  Management Unit
    Level:
  DFC-1: % Openings        Satisfied
    < 25%; AND                (Value=0)
```

Table VI **Current Condition NED DSS Goal Analysis for Bent Creek Experimental Forest**

Goal name	Management unit rating	Stand rating %				
		No. of stands	Fully satisfied	Minimally satisfied	Nearly satisfied	Not satisfied
Large scale variety	Not satisfied	0	N/A	N/A	N/A	N/A
Local biodiversity	Fully satisfied	0	N/A	N/A	N/A	N/A
Sawtimber production	Not satisfied	65	8	14	0	78
Limit peak flow	Not satisfied	8	0	0	0	100
Black bear	Minimally satisfied	65	25	0	0	75

Note: The column labeled "no. of stands" contains the number of stands for which the stand rating percentages apply. For example, there are only eight riparian stands in the Bent Creek Experimental Forest. Nonriparian stands are not considered in evaluating the goal "limit peak flow." N/A means not applicable.

```
DFC-2: % All
  Riparian Stands
  must meet stand
  level DFCs              Not Satisfied

Stand Level:
DFC-3: Relative
  Density >= 70; AND
DFC-4: % Basal Area
  of evergreen trees
  > 0 m2/ha; AND
DFC-5: Canopy
  Closure > 25%.
```

The Bent Creek management unit has no trouble meeting DFC-1 because there are no stand-sized permanent openings or regeneration size-class stands that would also qualify as an opening. There are only eight riparian stands on the Bent Creek Experimental Forest and none of them satisfy DFC-4. The problem is that none of the eight riparian stands have evergreen trees.

Goal G-5, Black Bear, is rated "not satisfied" (Table VI). The goal completion report in the NED DSS provides the following facts:

```
Goal-5: Wildlife          Not Satisfied
  Resource Domain
Habitat for black bear
  is achieved IF
Management Unit Level:
  DFC-1: > 30% of         Minimally
    Stands must meet        Satisfied
    stand level DFCs        (Value=25)

  Stand Level:
  DFC-2: Coarse woody
    debris > 3.5 m3/ha;
    AND
  (DFC-3: Soft mast
    producing trees
    are present; OR
  DFC-4: Hard mast
    producing trees are
    present).
```

Because we did not have direct coarse woody debris inventory data for Bent Creek, we assumed that both regenerating stands and large sawtimber stands had more than the 3.5 m3/ha of coarse woody debris per acre threshold value in DFC-2. All other size classes had less. Almost all stands in Bent Creek have either a soft or hard mast component and since the threshold is simply a presence/absence one, all stands meet DFC-3 or DFC-4. Only 25% of the stands currently satisfy DFC-2 which is within +/−10% of the required threshold value of 30% and therefore earns a "minimally satisfied" rating.

B. Alternative Design and Analysis—The Custodial Alternative

In the previous section, we went into great detail in order to illustrate how to compare a goal hierarchy to the current condition. The same method is used to compare the goal hierarchy to any other alternative. From this point forward, we will simply present and discuss the interesting results.

One popular alternative is to do nothing. This alternative may be labeled the Custodial Alternative. With no active human management activities allowed, the Custodial Alternative can be used to represent one end of the spectrum of choices. In order to evaluate the consequences of the Custodial Alternative we can forecast the natural dynamics of growth and death on the Bent Creek Experimental Forest to a common point in time in this case 40 years into the future. This forecasting simulation needs to provide us with an estimate of the effects of implementing the alternative under consideration on the landscape being managed. From this analysis we will gain an understanding of the consequences inherent in the proposed alternative. Forest vegetation simulation system (FVS), (Table III), a general vegetation dynamics simulation model, was used to generate the forecast for this example. The resultant simulated future forest ecosystem was then compared to the goal hierarchy using NED.

Under the Custodial Alternative (Table VII), Goal G-1, Large Scale Variety, has lost ground. Although the overall rating of "not satisfied" has not changed, 40 years from now 93% of the Bent Creek management unit will be in the Large Sawtimber size class. From the perspective of satisfying goal G-1, having 93% of the area in a single size class is definitely worse than having 76% in a single size class, which is the current situation. Goal G-2, Local Biological Diversity, has also moved further away from being satisfied. Now, the small and large sawtimber sizes together make up 99% of the forest compared to the current condition of 95%. Goal G-3, Sawtimber Production, has also declined. Under the Custodial Alternative 87% of the stands do not meet the stand DFCs, up from the current 78%. The proportion of stands that were rated "fully satisfied" dropped from 14–6% due to a reduction of the proportion of high-value species as a percentage of total stand basal area. In other words, high-value sawtimber species were losing the competition battle to more vigorous but lower value sawtimber species. A few stands improved their rating because their relative density increased. But overall, the Custodial Alternative moved the forest away from satisfying goal G-3. Goal G-4, Limit Peak Flow, remained un-

Table VII Custodial Alternative NED DSS Goal Analysis for Bent Creek Experimental Forest

Goal name	Management unit rating	Stand rating %				
		No. of stands	Fully satisfied	Minimally satisfied	Nearly satisfied	Not satisfied
Large scale variety	Not satisfied	0	N/A	N/A	N/A	N/A
Local biodiversity	Not satisfied	0	N/A	N/A	N/A	N/A
Sawtimber production	Not satisfied	65	6	5	2	87
Limit peak flow	Not satisfied	8	0	0	0	100
Black bear	Fully satisfied	65	78	0	0	22

Note: The column labeled "no. of stands" contains the number of stands for which the stand rating percentages apply. For example, there are only eight riparian stands in the Bent Creek Experimental Forest. Nonriparian stands are not considered in evaluating the goal "limit peak flow." N/A means not applicable.

changed because evergreen species were still missing from the riparian stands.

Finally, Goal G-5, Black Bear, showed a marked improvement under the Custodial Alternative (Table VII). The overall rating for the management unit improved from "minimally satisfied" to "fully satisfied." Due to numerous stands growing into the large sawtimber size class from the small sawtimber size class, the proportion of stands rated as "fully satisfied" improved from 25 to 78%. Remember, that we are assuming that all large sawtimber sized stands provide more than the required 3.5 m3/ha of coarse woody debris while small sawtimber sized stands do not. One could certainly argue that the desired conditions for black bear ought to be modified so that mast production is more than a presence/absence metric. One would expect a large sawtimber sized oak stand to produce significantly more acorns than a small sawtimber oak stand. But the "closed world assumption" forces us to ignore such considerations because they are not considered in the DFCs as currently defined. The goal hierarchy can be changed at any time by the decision makers, stakeholders, and domain experts to create a new closed world assumption if the present one needs improvement.

In summary, the homogeneous large sawtimber management unit created by the Custodial Alternative is less successful than the current condition in satisfying goals G-1, G-2, and G-3. It improves the rating of goal G-5 compared to the current condition and is neutral with regard to goal G-4.

Several important points should be noted. First, these analyses are clear and understandable. Each goal can readily be compared to the current conditions and any hypothesized alternative. Some goals such as goals G-1, G-2, and G-3 move in concert with each other. Some goals, such as G-5, move in the opposite direction. Some goals, such as G-4, are neutral with respect to some kinds of changes in the forest ecosystem. Furthermore, it is easy to recognize which goals have which tendencies. Those goals that tend to move in concert over a wide variety of ecosystem state changes can be thought of as generally compatible with each other. Those which frequently move in opposite directions can be thought of as generally incompatible with each other. One by-product of this process is therefore a clear and objective way to identify goal compatibility, neutrality, or conflict.

C. Alternative Design and Analysis—Maximum Sustainable Sawtimber Production

An interesting, and often used, contrast to the Custodial Alternative is the Maximum Sustainable Sawtimber Production Alternative. Under this alternative, Goal G-3 is favored and allowed to dominate all other goals. This means that goals that are compatible with goal G-3 will probably also improve their satisfaction rating. Those goals in conflict with goal G-3 will be negatively impacted.

We used a Southern Appalachian Hardwood Forest Regeneration Simulation Model to forecast stand composition and size following a regeneration harvest. The USDA Forest Service's FVS was used for growth and mortality prediction of existing stands. It was also used to simulate various types and intensities of thinnings over the 40-year projection period. The NED DSS was again used to compare the resulting simulated forest with the goal hierarchy. We assumed no financial constraints.

To design this alternative, we need to understand what we have to do in order to improve quality sawtimber production at Bent Creek. By reviewing the current condition analysis, we realize that we have to create a better distribution of size classes. This implies regeneration harvesting. Beyond that, we need to make sure each stand is well stocked, not understocked and not overstocked. Stocking control is usually achieved by thinnings of various kinds. We also need to favor the high-value species whenever possible. We can discover which species are high value by looking at the species encyclopedia in NED. Releasing high-value species from competition also implies thinning prescriptions.

After examining the current conditions and the custodial alternative, we identified several types of stands where active management will move Bent Creek closer to the goal of sustained sawtimber production. First, there are many stands in the small sawtimber size class that become too dense if left alone for 40 years to grow and where the proportion of basal area in high-value species declines in favor of less valuable commercial tree species. By thinning these stands from below we can keep the stand within the density range best for sawtimber production. By favoring the high-value species while we are thinning and removing their immediate competitors, we can increase their proportionate basal area. Next, there are stands in the large and small sawtimber size classes with lower than optimum stand density. These stands become prime candidates for regeneration.

The maximum sustainable sawtimber alternative appeared to improve goal accomplishment dramatically (Table VIII). Both goals, G-1 (Large Scale Variety) and G-2 (Local Biological Diversity), became fully satisfied as we aggressively achieved a more balanced size-class distribution of stands in the management unit. Goal G-3 (Sawtimber Production) improved to minimally satisfied because 40 years was not enough time to repopulate the small sawtimber size class from stands growing out of the sapling and pole size classes. At the stand level, we can see that the thinning prescriptions were effective in reducing the number of stands in the "not satisfied" category from 78% in the current condition to only 20% under this alternative. Goal G-4 (Limit Peak Flow) has still not been satisfied because there are still no evergreen trees in the riparian zone stands. As expected, Goal G-5 (Black Bear) changed from "minimally satisfied" to "not satisfied." Recall that DFC-1 for Black Bear required greater than 30% of the stands to meet stand level DFCs. Although regenerating stands also provide enough coarse woody debris to pass DFC-2, these stands grow into the sapling and pole size classes after only 10 years. The transient nature of the regenerating stands and the reduction in large sawtimber stands contribute to the degradation of Black Bear habitat under this scenario.

D. Alternative Design and Analysis—Equal Preference for All Five Goals

Another possible alternative might be to give all five goals equal preference. The equal preference alternative would, in our example, lead to a more moderate management intensity. To improve the status of three of the five goals, we still need to strive for a more balanced size distribution. But because adequate coarse woody debris, which is a desired condition for the Black Bear goal, is associated with either regeneration or large sawtimber size classes, we might

Table VIII **Maximum Sustained Sawtimber Production Alternative NED DSS Goal Analysis for Bent Creek Experimental Forest**

			Stand rating %			
Goal name	Management unit rating	No. of stands	Fully satisfied	Minimally satisfied	Nearly satisfied	Not satisfied
Large scale variety	Fully satisfied	0	N/A	N/A	N/A	N/A
Local biodiversity	Fully satisfied	0	N/A	N/A	N/A	N/A
Sawtimber production	Minimally satisfied	65	22	43	15	20
Limit peak flow	Not satisfied	8	0	0	0	100
Black bear	Not satisfied	65	13	0	0	87

Note: The column labeled "no. of stands" contains the number of stands for which the stand rating percentages apply. For example, there are only eight riparian stands in the Bent Creek Experimental Forest. Nonriparian stands are not considered in evaluating the goal "limit peak flow." N/A means not applicable.

want to overrepresent the large sawtimber size class. We can only keep a stand in the regenerating size class for 10 years but we can maintain a healthy large sawtimber size stand for a much longer time. We can also create more coarse woody debris by altering how we thin stands. If we prescribe that low-value trees, either due to low-value species, poor form, or size, be simply felled and left on the site during the commercial thinning operations, then we can artificially improve béar habitat at relatively low cost. Of course our assumption that small sawtimber size stands do not have adequate woody debris may not be accurate. Monitoring and evaluation of that assumption may alter it and thus produce a substantial change in the satisfaction rating for that DFC.

To implement this alternative, we thinned the same stands as under the maximum sustained sawtimber alternative. We also regenerated the stands in the large and small sawtimber size classes with lower than optimum stand density. The well-stocked, small sawtimber stands with deficient high-value species, however, were left to grow.

In addition, the equal preferences alternative provided the justification to spend resources to artificially introduce an evergreen component into the eight riparian stands in order to satisfy the "limit peak flow" goal. We simulated the underplanting of hemlock along with a light release thinning for each hemlock planted.

Implementing the equal preferences alternative just described resulted in satisfying all the goals to some degree (Table IX). As before, we used FVS, the Southern Appalachian Hardwood Regeneration Simulation Model, and NED to simulate implementing this alternative. Goal G-1 (Large Scale Variety) was rated as "minimally satisfied" because the Large Sawtimber size class represented 46% of the area of the stand, which is just barely below the threshold of 50% in DFC-3. Goal G-2 (Local Biological Diversity) was "fully satisfied." Goal G-3 (Sawtimber Production) was rated "nearly satisfied" primarily because the percentage of stands in the sapling and pole size class was just under the minimum 35% threshold value and the large sawtimber size class was just over the maximum 15% threshold value. Within the time frame of the example we were unable to harvest and regenerate a sufficient number of stands to fully satisfy the balanced size class requirements of Goal G-3. At the stand level, this alternative was not as good as the maximum sawtimber alternative (Table VIII) but was substantially better than the current condition or the custodial alternative (Tables VI and VII). Because we planted the riparian stands with hemlock, we were able to fully satisfy goal G-4 (Limit Peak Flow). Finally Black Bear was "fully satisfied" because many stands had enough coarse woody debris to satisfy the threshold in DFC2. This alternative fostered many large sawtimber sized stands, some regenerating stands, and, improved the amount of coarse woody debris in other size class stands by leaving coarse woody debris in thinned stands.

E. Alternative Selection and Authorization to Implement

At this point we have created the goal hierarchy and designed alternative courses of action. For each alternative, we simulated its implementation and forecast the resultant forest ecosystem at the end of a 40-year period. We then evaluated each alternative by determining whether the DFCs in the goal hierarchy were fully satisfied, minimally satisfied, nearly satisfied, or unsatisfied (TablesVII–IX). We found it convenient to treat

Table IX Equal Preferences Alternative NED DSS Goal Analysis for Bent Creek Experimental Forest

			Stand rating %			
Goal name	Management unit rating	No. of stands	Fully satisfied	Minimally satisfied	Nearly satisfied	Not satisfied
Large scale variety	Minimally satisfied	0	N/A	N/A	N/A	N/A
Local biodiversity	Fully satisfied	0	N/A	N/A	N/A	N/A
Sawtimber production	Nearly satisfied	65	12	18	42	28
Limit peak flow	Fully satisfied	8	100	0	0	0
Black bear	Fully satisfied	65	69	0	0	31

Note: The column labeled "no. of stands" contains the number of stands for which the stand rating percentages apply. For example, there are only eight riparian stands in the Bent Creek Experimental Forest. Nonriparian stands are not considered in evaluating the goal "limit peak flow." N/A means not applicable.

the current condition as if it were an alternative because it allows us to compare whether any of the alternatives improved on the current condition of the forest ecosystem. This body of information may then be used to objectively select an alternative for implementation.

Of the many possible choice methodologies available, we decided to use the Analytical Hierarchy Process (AHP) developed by Saaty (1992) as implemented by the commercial CRITERIUM DECISION PLUS software package (InfoHarvest, 1996). Two preparatory steps are required to apply the AHP choice methodology: (1) decide how important each goal of the same hierarchical level is in comparison with all others and (2) rate the performance of each DFC against each alternative. Recall that the design of our alternatives involved two separate positions on the relative importance of goals in the hierarchy. The one position was that all goals are equally important and the second position was that the goal G-3, Sawtimber Production, was more important than the others. We will examine the influence of this value-driven decision on the choice of alternatives.

If we assume that all goals in the goal hierarchy are equally important in the decision process and then rate the expected results of implementing each alternative against the DFCs in the goal hierarchy, we find that the "equal goal preferences" alternative is slightly better than the "maximum sustainable sawtimber" alternative (Table X). The rating value of 0.89 in Table X is a composite score across all goals in the goal hierarchy calculated by the software using the AHP algorithm. These composite values are meaningful only for ranking the alternatives on a common, relative scale. Both the "equal goal preferences" and "maximum sustainable sawtimber" alternatives are a considerable improvement over both the "current condition" and the "custodial—do nothing" alternatives. The effect of the "equal goal preferences" alternative is to provide a good balance of achievement across all the goals. When we alter the preference values to make the "maximum sustainable sawtimber" objective twice as important as the other goals, we observe the expected shift in the preferred alternative (Table VIII). The preferred alternative is influenced not only by the consequences of implementing that alternative on the forest ecosystem but also by the differences in the importance of the goals as reflected by the preferences used in the analysis. The ability to clearly understand and communicate both of these factors in alternative selection is an important attribute of the AHP methodology.

In this simple illustration, it is easy to review Table IX and determine that we have satisfied all the goals to some degree, whereas in Table VIII it is readily apparent that the "maximum sustainable sawtimber" alternative leaves two of the five goals unsatisfied. A more realistic example with over 100 goals would be more difficult to synthesize using output in the format of Tables VI–IX. For this larger problem situation, the composite scores generated by the AHP method would have clear advantages in alternative comparison and selection.

VI. SUMMARY AND CONCLUSIONS

We have come a long way in the application of information technology to natural resource management although much of this progress has occurred relatively recently. Widespread use of information systems

Table X Alternative Comparison of Results from Bent Creek using the Analytical Hierarchy Process (AHP) as the Ranking System

	Equal preference values for all goals	Double preference value for sawtimber production
Alternative name	AHP Scores[a]	AHP Scores[a]
Current condition	0.58	0.40
Custodial (do nothing)	0.60	0.41
Maximum sustainable sawtimber focus	0.82	0.81
Focus on achieving all goals	0.89	0.76

Note: The alternative name indicates the types of treatments that were selected to achieve a particular focus. The current condition is simply an analysis of how well the current forest ecosystem satisfies the goal hierarchy. Preference values are part of the AHP scoring system which is independent of how alternatives have been defined.

[a]Higher scores are better.

in natural resource management had to await the introduction of the microcomputer and the evolution of high-quality and user-friendly word-processing systems in the mid-1980s. At that time, word-processing systems became the first knowledge management tool widely used in natural resource management. Although quantitative simulation models for natural resource management applications had been developed as early as the 1960s for mainframe computers, these were primarily research tools and not widely used. Once again, it took the availability of inexpensive microcomputers to make forest growth and yield projection systems as well as other simulation models commonly used tools in natural resource management. The advent of the Internet as a generally available information technology is even more recent.

Despite these recent advances, many significant aspects of available information technology have not yet been adopted by natural resource managers. The recognition that natural resource management is a knowledge-intensive activity and that knowledge is *the* pivotal commodity is not widespread. Well-organized and maintained hypertext encyclopedias of knowledge are still a novelty. Knowledge-based advisory systems are a little more common, but still not widely used. Full service natural resource management DSSs are still in the development stage and have yet to prove their worth in practical natural resource management applications.

Over the next decade, we look forward to significant changes in the availability and popularity of the entire gamut of knowledge management and application tools. First, Internet-based hypertext encyclopedias of knowledge will be developed and tested. We expect a gradual shift in the delivery of scientific information from paper-based journals to fully synthesized, Web-based hypertext systems. Figuring out how to finance these ventures and make them profitable will be the key breakthrough. Second, full service DSSs for significantly larger, more complex problems will become commonplace. This development awaits the arrival of a simple and effective way to allow independently developed software to work together to solve a joint problem. Recent advances in such software communication standards look promising. We also need to design and test flexible and practical decision analysis processes that are easy to understand by the general public. Once again, recent research in testing such decision analysis processes in conjunction with decision support for natural resources management looks promising.

SEE ALSO THE FOLLOWING ARTICLES

Decision Support Systems • Expert Systems • Geographic Information Systems • Hybrid Systems • Knowledge Management • Knowledge Representation • Multimedia

BIBLIOGRAPHY

Barber, K. H., and Rodman. S. A. (1990). FORPLAN: the marvelous toy. *Journal of Forestry,* 88, 26–30.

Greber, B. J., and Johnson, K. N. (1991). What's all this debate about overcutting? *Journal of Forestry* 89, 25–30.

Hoekstra, T. W., Dyer, A. A., and LeMaster, D. C., eds. (1987). FORPLAN: An evaluation of a forest planning tool. USDA Forest Service, Gen. Tech. Rep. RM-140.

Holsapple, C. W., and Whinston, A. B. (1996). *Decision support systems: A knowledge-based approach.* Minneapolis/St. Paul:West Publishing Co.

InfoHarvest. (1996). *Criterium decision plus: user's guide.* Seattle, WA: InfoHarvest Inc.

Jorgensen, S. E., Halling-Sorensen, B., and Nielsen, S. N. (1996). *Handbook of environmental and ecological modeling.* Boca Raton, FL: Lewis Publishers.

Klein, M., and Methlie, L. B. (1990). *Expert systems: A decision support approach.* Reading, MA: Addison-Wesley.

Mowrer, H. T., Barber, K., Campbell, J., Crookston, N., Dahms, C., Day, J., Laacke, J., Merzenich, J., Mighton, S., Rauscher, M., Reynolds, K., Thompson, J., Trenchi, P., and Twery, M. (1997). Decision support systems for ecosystem management: An evaluation of existing systems. USDA Forest Service, Interregional Ecosystem Management Coordination Group, Decision Support System Task Team, Rocky Mountain Forest and Range Experiment Station, RM-GTR-296.

Rauscher, H. M. (1999). Ecosystem management decision support for federal forests in the United States: A review. *Forest Ecology and Management,* 114, 173–197.

Reynolds, K., Jensen, M., Andreasen, J., and Goodman, I. (2000). Knowledge-based assessment of watershed condition. *Computers and Electronics in Agriculture* (in press).

Saaty, T. L. (1992). *Multicriteria decision making—The analytical hierarchy process.* Pittsburgh, PA: RWS Publications.

Schmoldt, D. L. and Rauscher, H. M. (1996). *Building knowledge-based systems for natural resource management.* New York: Chapman & Hall.

Schuster, E. G., Leefers, L. A., and Thompson, J. E. (1993). A guide to computer-based analytical tools for implementing national forest plans. USDA Forest Service, Gen. Tech. Rep. INT-296.

Zadeh, L. A. (1975). The concept of a linguistic variable and its application to approximate reasoning. Part II. *Information Science* 8, 301–357.

Zadeh, L. A. (1992). Knowledge representation in fuzzy logic. In: *An Introduction to Fuzzy Logic Applications in Intelligent Systems* (R.R. Yager and L.A. Zadeh, eds.), pp. 1-25. Boston, MA: Kluwer Academic Publishers.

Network Database Systems

Frederick N. Springsteel
University of Missouri, Columbia

GLOSSARY

conference on data systems and languages (CODASYL) An industry-wide association responsible for the standardization of the language COBOL, with subgroup DBTG, who produced the 1971 DBTG report that described the network database (industry) standards.

data base task group (DBTG) The subgroup of CODASYL that proposed twin languages for network databases: their DDL and DML.

data definition language (DDL) A formal notation, or computer language, for implementing and modifying network schemas and record types for a network database.

data manipulation language (DML) A formal notation or computer language suitable for writing application programs that manipulate the data within a network database, using the database's conceptual schema of record- and set-types.

network database Any legacy database that was built on the network data model, where all records of the database are physically linked into an internal "network" of records in the 1960s sense, *not* a network-of-computers sense.

NETWORK DATABASES were first created in the 1960s to improve and extend the earliest computer information systems. They were built to assist the largest-to-date coordinated human project, the aerospace industry's Project Apollo and its many missions of manned space flight, including the first landings on the moon in 1969. Back then there were several competing vendors of network database systems. The changes in computer technology during 1970–2000 have been so great that none of these products are marketed today. Powerful network databases were used to build the *Apollo* vehicles for the U.S. space program. However, at that time there was a great need to coordinate terminology across the field and between vendors, due to a proliferation of many vendors' incompatible systems. The hierarchical data model (HDM), which may be more easily seen as a special case of the network data model (NDM) with further restrictions, existed concurrently with it.

Both hierarchical and network database systems are now called *legacy systems* because some of their implementations are still very much with us, but no new ones have been created since relational database systems became available and rapidly improved in the 1980s. Although the hierarchical database systems and their HDM were more restricted, logically the network database systems may be easier to explain. Also the latter's database design model, the NDM, was the first to become a database systems specification, via the Conference on Data Systems and Languages (CODASYL) Data Base Task Group (DBTG) 1971 report. CODASYL empowered its DBTG to devise standards for network database systems (NDBS), which it successfully did. What emerged from this volunteer, multiple-vendor-supported committee was a concrete, agreed-on form of the network data model.

I. THE NETWORK DATABASE SYSTEMS' DATA MODEL

It is generally accepted that the first general-purpose database management system (for large computers) was designed by Charles Bachman at General Electric in the early 1960s; it was called Integrated Data Store (IDS). It formed the basis for the famous network data model, which was later standardized by CODASYL's DBTG. IDS strongly influenced the early database systems, which were needed for large engineering projects, like the U.S. space program's Project Apollo. In 1973, Bachman won the first Turing Award (named after Alan Turing), the computer science equivalent of a Nobel Prize, for his pioneering database work.

Another important early NDBS was IDMS (Integrated Database Management System), which was developed by Cullinet Software for IBM mainframes. Cullinet later merged into Computer Associates, which continues limited product support. IDMS was probably the best known example of what was called a *CODASYL system* or a *DBTG system*. While several other NDBS based on the 1971 DBTG report were built during the 1970s, one updating of IDMS had a longer term effect: In 1983, Cullinet released an extended version of IDMS, called "IDMS/R," which—in addition to its network data facilities—included certain relational database features. When Computer Associates (CA) acquired Cullinet in 1989, it renamed this upgraded product "CA-IDMS" and even extended it to support SQL, the relational query language. To date, CA continues technical support of this "hybrid" database product's existing installations but is not actively marketing it for new data storage implementations.

One can best begin to understand network database systems, like IDS and IDMS, by learning about their basic database design model, the network data model. Conceptually, the NDM is a linear abstraction of the design concepts used to plan and implement the 1960s legacy network DB systems. Thus the NDM is more closely connected to physical-level design than its late-born cousin, the relational data model. The fundamental concept of the NDM is this: Data are represented by a collection of *records,* which are in turn collections of fields (or data items). Each field contains only one data value at a time, and each field of a certain name, over time, contains only values drawn from a fixed domain set. Suppose we have a concrete set of comparable persons, places, or things, about which we want to record data. If common records that have exactly the same fields can represent all these objects, we call this set of records a *record-type*. A record-type is also seen as a list of its field names. One can say that the "record-type" is an abstract concept representing the ever-changing set of actual records, the record set itself, which changes over time as records are added to and/or deleted from the data. However, the record-type's format always stays the same.

The necessary relationships between records are implemented by *links* (pointers) that are associations between exactly two records. A *link-type* is a (restricted) form of two-way relationship. Link-types can be classified by the numbers of records of each record-type that can take part in an association: thus they are called one-to-one, one-to-many, or many-to-many, where "many" means an indeterminate number, possibly one (or even zero). In the CODASYL DBTG standard, all links happen to be implemented as many-to-one relationships. However, the NDM can handle many-to-many cases: Multiple records can be linked via multiple many-to-one links to one record, as we shall see below.

Roughly, the NDM is that collection of concepts considered necessary for the design of NDBS. Clearly it is a prior step to master these concepts first, before one tries to design a NDBS using them. We will be conservative here in the number of fundamental terms. Later, terminology that is specialized to the CODASYL DBTG data definition language (DDL) will make these basic terms more concrete. For simplicity, we restrict our relationships to be of the one-to-many (or many-to-one) type because that is the only kind that ends up in the DBTG DDL.

A. Records and Record-Types

In essence the NDM has only two basic components: (1) records, which represent those real objects that the network database will track, and (2) links between precisely two records of different record-types. Think of a record as a collection of data fields, or record segments, which represent individual attributes or properties of the objects recorded. Each field takes its (current) value from its own fixed domain, or set of possible values, such as numbers or character strings. When the fields—say, `NAME`, `ADDRESS`, and `PHONE`—are each given values for a record of type `CUSTOMER`, then we have represented one object, an actual customer (Fig. 1). So a record-type is really a designation for a potential set of records, called the record set. The set of names for the fields and the domains of the fields constitute the (logical) record format.

NAME	ADDRESS	PHONE
John Q. Smith	111 Main Street	456-1234

Figure 1 The record type CUSTOMER.

For one familiar with the relational data model (RDM), close analogies exist between the record-related concepts of the NDM and the "table-of-rows" concepts of the RDM (Table I). However, there is a notable distinction between the records of an NDM record set and the tuples of the corresponding RDM relation. The RDM is "value oriented" because it is more mathematical: Relations are sets of tuples, and tuples are no more than (lists of) the values of their component attributes; that is, two tuples with the same schema and the same values are the *same* tuple! (Sets cannot have duplicate members.) The corresponding statement may not be true in the NDM, which is "object oriented": It supports *object identity,* in the form of record identity. In other words, two distinct records with exactly the same format and values in each field may exist without being identical! Perhaps they are (allowed) duplicates—which cannot exist in the same relation in the RDM—but more likely they are mere coincidences: two customers with the same (limited) data. In the RDM, some field(s) in each relation constitute a *logical key,* or unique identifier, like SSN, so duplication of these values should not occur, and it is illegal in the RDM. On the other hand, any records in the NDM can be seen as having an "invisible key" that is, in essence, the physical location (or logical address) of the record, which gives each record its "object identity." Via this unique identifier, two records are considered distinct even if they contain the same values in all of their corresponding (visible) fields. In fact, because of record identity it is feasible to have record types with no data items at all! This is what "virtual records" will be: mere locations for pointers.

Table I Analogous Terms

Network term	Relational term
Field	Attribute (column)
Record	Tuple (row)
Record set	Relation (set of tuples)
Record-type	Relation name
Record format	Relation schema

B. Links, Associations, and Set-Types

In the NDM a relationship between records is called a *link* because, when implemented in an actual NDB, this model uses physical pointers to connect two records of two different types, if and only if the records (tuples) are associated, many to one, by the corresponding relationship. In the RDM, or in the entity-relationship model (ERM), there is a similar well-defined notion of "binary (two-way) many-to-one relationship" but there the objects are only logically related.

One can draw a graph, called a *network;* like an ER diagram, the network (diagram) has *nodes* to represent record-types and *arrows* to connect two nodes if their record-types are associated and should be linked. If record-type T_1 is linked to record-type T_2 (e.g., ORDERS to CUSTOMER) and the link is many-to-one from T_1 to T_2, then we draw an arrow from T_1 (ORDERS) to T_2 (CUSTOMER); let us call this link BELONG_TO. Nodes and arrows are labeled by the names of their record-types and links, in the so-called network diagram (Fig. 2). The intent is that order records of the type ORDERS will be linked to a particular CUSTOMER record if and only if they represent the current orders of that customer.

C. Representing Various Associations as Links

Remember that the only type of relationships between records that we shall use in the NDM will be *binary* (between two record-types) and *many-one* (or one-one, as a special case). Thus all the many-one links can be shown as one-directional arrows, from the "many" record to the "one" record. The direction of the arrows here is contrary to that of some early authors, for example, to Bachman's 1969 diagrams, but it is consistent with the usage of Ullman. There is a good reason for the modern arrows' directionality: This orientation coheres better with the (relational) notion of *functional dependency,* a mathematical function is always many to one.

While it might seem at first a restriction to require all relationships to be binary, we can use the following device to represent an arbitrary number of objects in one relationship. Suppose that we need to represent an association A among record-types $R_1, R_2, \ldots, R_k$. We can create a new (logical) record-type T to represent k-tuples of the form $(r_1, r_2, \ldots, r_k)$ built up from the basic records (actual occurrences, *not* types) r_i where each such record is of type R_i, $i = 1, 2, \ldots, k$. The purpose

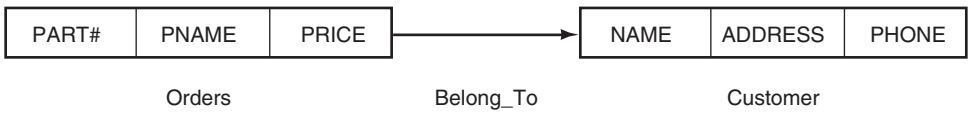


Figure 2 Network diagram for link BELONG_TO.

of T is to associate those r_i that stand in relationship to each other via A. The format of the new type T might even be empty (!); after all, the basic records r_i each already exist and have identities, that is, locations. However, there are times when it is useful to have information-carrying fields in the new logical record format for T, as we shall see below. In any case, we create links L_1, L_2, . . ., L_k where link L_i is from record-type T to record-type R_i, $i = 1, 2, \ldots, k$. The intention is that the record of type T for $(r_1, r_2, \ldots, r_k)$ is linked to the record r_i where each such record is of type R_i, for each i, so each link is binary and many-to-one.

As a special case, if the relationship is many-one from the first $(k - 1)$ types $R_1, \ldots, R_{k-1}$, to R_k, and this latter entity does not appear in any other relationship, then we can identify the record-type T with R_k, storing the attributes of R_k in T. For example, the relationship SUPPLIES is many-one from the pair SUPPLIERS and ITEMS to PRICE, and PRICE is in no other relationship. We may thus create a type T with links to SUPPLIERS and ITEMS and containing PRICE as a field. (We further illustrate this example in the next section.)

II. DATABASE DESIGN IN THE NETWORK DATA MODEL

To demonstrate design in the NDM, using the basic concepts above, we return to the SUPPLIES relationship of the previous section. Because it is *not* a binary relationship, we have to transform it into two of the latter. As an ERD the relationship is as shown in Fig. 3.

This diagram shows that the relationship is ternary, and many-one from the pair (SUPPLIER, ITEM) to PRICE. Our approach to replacing it by binary relationships is to ask in this, as in any similar modeling situation, what is the true nature of the relationship between SUPPLIER and ITEM that associates with some possible pairs of them exactly one PRICE? Clearly, the binary association—if any—of SUPPLIER to ITEM would be many-to-many, because many SUPPLIERs can relate to one ITEM, and many ITEMs can relate to one SUPPLIER. When each related pair has definite "intersection data," such as PRICE is here for a pair of a SUPPLIER and (supplied) ITEM, the standard way in NDM modeling to show many-to-many relationships is to create two many-one relationships (i.e., two links in the NDM) from the pair's intersection data, PRICE, to each record-type in the determining pair. That is, each pair of SUPPLIER and ITEM records is related to the associated PRICE record, which works because it takes both a supplier and an item to determine the price.

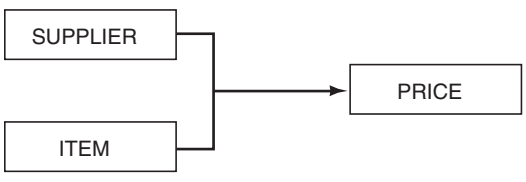


Figure 3 Ternary relation diagram for the SUPPLIES relationship.

This is exactly what we proposed for special cases in the previous NDM discussion: The new type T that in this example could be called OFFER, as it represents a willing SUPPLIER offer of a given ITEM at a given PRICE, contains only one field, PRICE, and so each PRICE record is linked to its associated SUPPLIER and ITEM records (Fig. 4).

In terms of logical record formats, we have now three record types—SUPPLIER (S#, SNAME), ITEM (ITEM#, INAME), OFFER (PRICE)—and two links: O.SUPPLIER, O.ITEM, which are the respective labels on the arrows just above. Putting this example into the context of a more complex ERD, with a one-to-one binary relationship, MANAGES, and an isa (subset) relationship between MANAGER and EMP(loyee), it yields the more complex relationship diagram that we see in Fig. 5.

After applying NDM concepts to this ERD, the resulting network diagram of Fig. 6 is not hard to understand. The only special remark needed is that the relationship MANAGES, originally between DEPT and MANAGER, will now be between EMP and DEPT, due to two simplifications: (1) When an ISA relationship shows the same or fewer fields in its subset type, MANAGER(ENAME), as in its superset type, EMP(ENAME, E#), then we can make the subset's "external" relationship, MANAGES, into one between DEPT and certain members of the superset EMP; (2) while one could

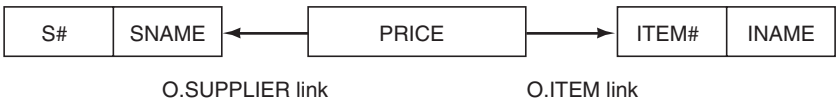


Figure 4 Network diagram for a ternary relation, OFFER.

easily add a 1-bit field to tell which EMPs are MANAGERs, this is not needed here because the subset MANAGER is implicit as those EMP records linked to DEPT via the link MANAGES (Fig. 6).

III. DATA DEFINITION IN THE NETWORK DATABASE LANGUAGE

To explain database design in concrete terms, we demonstrate data definition in approximately the data definition language adopted by the famous DBTG report. Moreover, to explain it briefly, we make use of the examples introduced above.

The DBTG report proposed its (schema) DDL as a formal notation, in linear format, for network diagrams. It also proposed a subschema DDL for defining *views,* which we omit here, plus a data manipulation language (DML) for writing applications that query and modify the contents of the database. We will examine the DML in the following sections.

A. Records in the Network Remain Records in the DDL

All of the record-related terminology we introduced in Sections I and II remains valid and will be used in the network DDL seen here. Sometimes, fields in record formats are called *data items* but we prefer to use *fields.* Recall that there is no requirement that two records with the same values (and type) be distinct; also record types with zero fields may exist. They may be needed to connect other records, and so these seemingly "empty" records would contain at least one pointer.

B. Links in the Network Become "Sets" in the DDL

To avoid confusion between the DBTG name for a link, *set,* and ordinary sets, we shall refer to links as *DBTG-sets.* Consider again the two links O.SUPPLIER and O.ITEM among the record-types SUPPLIER, ITEM, and OFFER (Fig. 6) of the previous section. These links derived from the ternary relationship SUPPLIES described earlier. The "owner" record of each link is the "one" object, the actual record at the arrow's head in a link occurrence, and the "members" are the "many" records on the other (origin) end of the link's arrow. For example, as the supplier "Best Supplies" is linked to the several prices that it offers for items, the linked list of owner and members (prices) is considered one DBTG-set: a set occurrence of the DBTG-set O.SUPPLIER. (Conventionally, the name of a set may include its owner's name.) The items that "Best" supplies—say brushes, rotors and combs—may have various prices linked to each item, depending on what price the other suppliers offer them for. But, in the DBTG-set O.ITEM, the owner record of *the* rotor that Best supplies has a unique (member record) price that is also a member in the DBTG-set O.SUPPLIER and is thus linked to owner-record "Best." It is precisely the *intersection* of the DBTG-sets at common price member-records, like

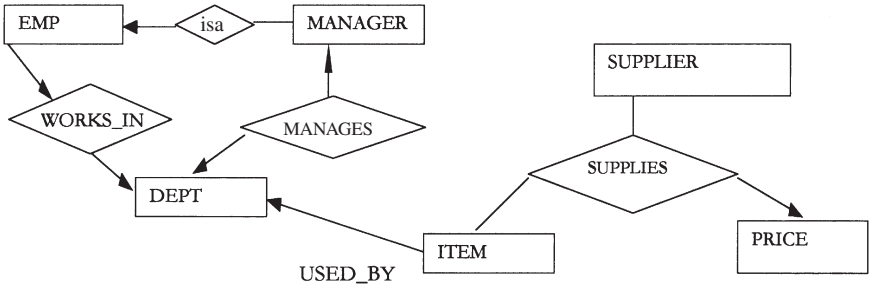


Figure 5 Example of more complex relationships.

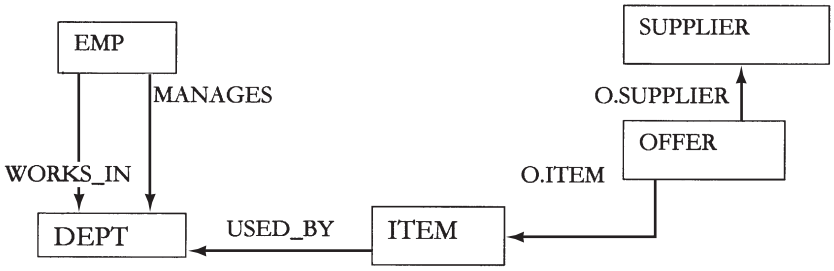


Figure 6 Network diagram for the example.

"$5.00," that actualizes in the DBTG DDL the ternary association SUPPLIES. This actualization makes it possible to "navigate" from the owner of one set, "Best Supplies," to the common member price ($5.00) and thence to this member's other owner, rotor, in the other DBTG-set. Note that the set occurrence diagram of Fig. 7 contains six price records, each involved in exactly two set occurrences. (There are also six set occurrences implied by Fig. 7, one for each owner record of the two DBTG-Sets: O.ITEM and O.SUPPLIER.)

The above two sets can be described in DBTG DDL briefly, without going into all of its low-level technical details, as follows:

```
RECORD SUPPLIER
  1 SNAME CHAR(20)
  1 SUPP#  INTEGER;
RECORD OFFERS
  1 PRICE REAL;

DBTG SET O.SUPPLIER
  OWNER IS SUPPLIER
  MEMBER IS OFFER;
```

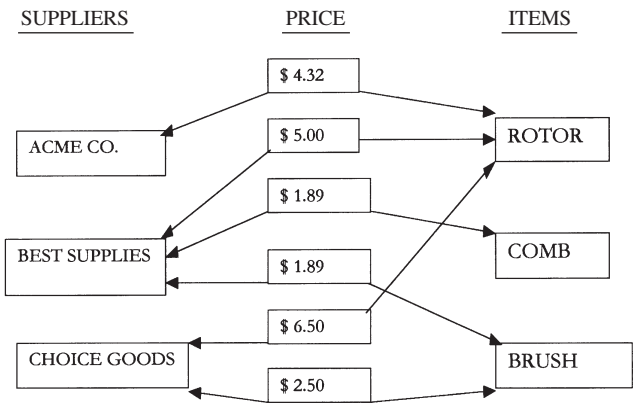


Figure 7 The set occurrences for the two SUPPLIES links.

```
RECORD ITEM
  1 ITEM#   INTEGER
  1 INAME CHAR(10);

DBTG SET  O.ITEM
  OWNER IS ITEM
  MEMBER IS OFFER;
```

To complete the DDL for Fig. 7, we need to declare two more record-types, EMP and DEPT, and three DBTG-sets: WORKS_IN, MANAGES and USED_BY:

```
RECORD EMP
  1 ENAME CHAR(30)
  1 E#    INTEGER;

RECORD DEPT
  1 DNAME CHAR(15)
  1 D#    INTEGER;

DBTG SET  WORKS_IN
  OWNER IS DEPT
  MEMBER IS EMP;

DBTG SET  MANAGES
  OWNER IS DEPT
  MEMBER IS EMP;

DBTG SET  USED_BY
  OWNER IS DEPT
  MEMBER IS ITEM;
```

IV. THE NETWORK DATABASE QUERY LANGUAGE

In this section and the next, we consider the data query and data modification aspects of the data manipulation language that is indicated by the DBTG standard for network databases. This section deals with the DML commands that query the database,

while the next covers the DML commands that update the network database.

In the DBTG proposal, the user is assumed to be an applications programmer and all network database programs are embedded in a host language (e.g., COBOL) and are coded in the commands of the DML, such as STORE (put a record into the database), FIND (locate a record in the database), and GET (read a record from the database). The rough format below suggests how the assignment of the value of $B+1$ to A might happen in a DBTG program:

```
GET(B)
MOVE (B + 1) TO A
STORE(A)
```

A. The Program Environment (or User's Working Area)

The memory, or workspace, environment in which a DBTG program operates in shown in Fig. 8. This area is also known as the user's working area, and it includes three kinds of data:

1. *Program variables,* which are defined in the DBTG/host-language program.
2. *Currency pointers,* which are pointer locations that refer to certain records in the database; we describe these various pointers more fully below.
3. *Record templates,* which are receiving variables for the various record types.

The template for a record type T consists of a named location for each field F of T, which location is called T.F in programs. An actual record t of type T can be stored in the database only after assembling the values of its component fields in the record template for T; then the STORE command copies the contents of the (valued) template into the record t in the database. Conversely, the GET command reads a (found) record from the database into the record's associated template. One also uses the template for passing *parameters* to certain commands, such as the FIND command.

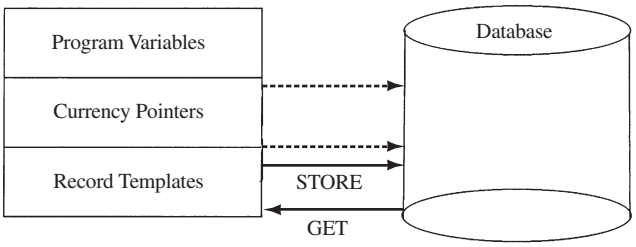


Figure 8 The user's working area.

B. Currency Pointers

For a DBTG program to run correctly, it must locate various records in the database with the FIND command, then process them by other commands. The network DBMS automatically keeps track of recently accessed records, via a group of *currency pointers*. The target locations that these pointers point to are recently accessed records, and those locations are available to the program, via the keyword CURRENT used in the following ways:

1. *The current of run-unit (CRU): Run-unit* means the DBTG program. The CRU points to the most recently accessed record, of any type.
2. *The current of record-type (CRT):* For each record type T, the CRT of T points to the most recently accessed record of this type, called the "current of T".
3. *The current of set-type (CST):* For a DBTG set S, consisting of owner type T and member type T', the most recently accessed record of type T or T' is called the "current of S." Note that it is a record, sometimes an owner, sometimes a member, rather than a set occurrence. Sometimes, abusing strict terminology, we speak of the set occurrence containing the "current of S" record as though it were a (fictious) "current S occurrence." In reality, an occurrence has no currency pointer.

Example:
Suppose that the data about suppliers from the O.SUPPLIER set in Fig. 7 is represented by the network database of the previous section. In particular, we focus on the set occurrence of the O.SUPPLIER set owned by the SUPPLIER record of "Best Supplies," in which set occurrence this supplier owns three OFFERS records, the prices for its supplied items of brush, comb, and rotor. Each solo PRICE item in each member record is also owned by its ITEM record in its respective O.ITEM set occurrence. To FIND the items supplied by "Best" we must visit each OFFERS record that it owns, where the prices are stored and are also linked in the rings to their owner ITEM records, per the O.ITEM set occurrence. By following those links we can find the owning ITEM, whose INAME field tells us an item that "Best" supplies. The record structure diagram is seen in Fig. 9.

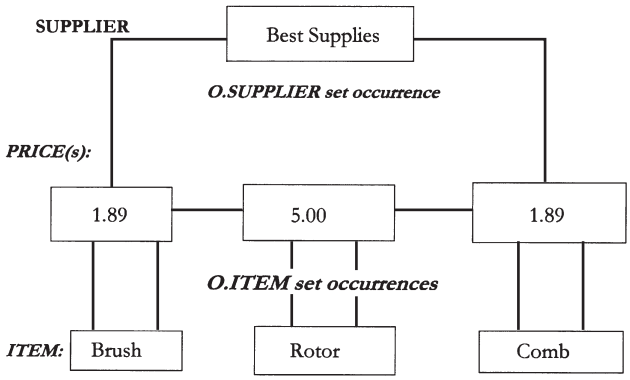


Figure 9 Set occurrences for "best supplies" and its items.

C. Effects of FIND on Currency Pointers

To explain the use of currency pointers, let us follow the effects that a sequence of commands to "find all items supplied by Best Supplies" would have on these pointers:

1. Find the `SUPPLIER` record for "Best (Supplies)."
2. Find the first member of Best's `O.SUPPLIER` set occurrence, the first record marked 1.89 in Fig. 9.
3. Go from the latter price to its owner in its `O.ITEM` set occurrence, i.e., the `ITEM` record for brush.
4. Find the second member of Best's `O.SUPPLIER` set occurrence, the second record, marked 5.00 in Fig. 9.
5. Go from the latter price to its owner in its `O.ITEM` set occurrence, i.e., the `ITEM` record for rotor.
6. Find the third (and last) member of Best's `O.SUPPLIER` set occurrence, the record marked 1.89 in Fig. 9.
7. Find the name of its owner record in `ITEM`, i.e., comb.

As these seven steps execute, they change the pointers of the CRU, and the CRTs of the `SUPPLIER, OFFERS`, and `ITEM` record types, and the CST of the `O.SUPPLIER` and `O.ITEM` sets, as follows. For example, the first step makes the "Best" record the current of run-unit, of the `SUPPLIER` record type, and of the `O.SUPPLIER` set type. Other currency pointers are undefined or retain their previous values. When the second step executes, the CRU moves to the (first) record marked 1.89, which also becomes the current `OFFERS` record and the CST of both the `O.SUPPLIER` and `O.ITEM` sets; other currency pointers are not changed. Continuing in this way, the history of the currency pointers values after steps 1 through 7 is given in Table II.

D. Navigation in the Database and the GET Command

Reading a record from the database to the workspace is a two-stage process: First, using the `FIND` command, we locate the desired record, so that it becomes the CRU. Then to copy the located record into its template in the workspace, an execution of the `GET` command suffices. The `GET` command always copies the current of run-unit into the template for whatever record type the CRU is. If one wishes to copy only a subset of the fields of the CRU, one lists the desired fields after the record-type name, which follows the keyword `GET`:

```
GET <record type>; [ <field(s)> ]
```

Table II Seven DBTG Commands' Effects on Currency Pointers

SUPPLIERS	OFFERS	ITEMS	O.SUPPLIER	O.ITEM	Run unit
1. Best Supplies	—	—	Best Supplies	—	Best Supplies
2. Best Supplies	1.89	—	1.89	1.89	1.89
3. Best Supplies	1.89	Brush	1.89	Brush	Brush
4. Best Supplies	5.00	Brush	5.00	5.00	5.00
5. Best Supplies	5.00	Rotor	5.00	Rotor	Rotor
6. Best Supplies	1.89	Rotor	1.89	1.89	1.89
7. Best Supplies	1.89	Comb	1.89	Comb	Comb

Example:
Suppose that the SUPPLIER record type is defined as above with fields SNAME and SUPP#. If the CRU is a SUPPLIER record, we can read the SNAME field by executing

```
GET SUPPLIER; SNAME
```

The SUPP# field in the template is not affected. A simple GET SUPPLIER reads all the fields into the template. If the CRU is not of the same record type as in the GET, then the system will tell the user of the error on execution.

E. The FIND Command and Its Variants

Searching for records with a certain value (or values) in a particular field (or fields) is done in DBTG by making the field(s) into a "calc-key," that is an *index* on the field(s). DBTG's proposal visualized implementing the index by a value calculated from the field, a *hash value,* but there are many possible implementations of indexes. The main idea is that, given certain field values, the index provides a fast method to locate (e.g., by random access pointers) all the records of one type with those values without linearly scanning the entire set of records, which may be huge! DBTG also includes the term *database-key* to mean a physical address pointer to a specific record in the database. Thus database keys always have unique referents, while general indexes seldom do. Indexes are sometimes crucial to quickly locate or "find" a given record. The FIND command in DBTG is really a group of somewhat different commands, distinguished by the keyword(s) after FIND. All of these variants have one common purpose: to locate a given record by some strategy special to each variant. While there is an extensive variety of FIND statements in extended DBTG, in this general article we only consider the following important subset of them:

1. Find a record given its database key, i.e., pointer to the record;
2. Find a record given a value for its index (its calc-key).
3. For a given record type, find (one at a time) all records with a given index's value.
4. Scan a set occurrence for the next member record.
5. Find the current of any record-type or of any set-type.

We shall give the essential FIND commands for executing DBTG's location functions, but simplify the process in two respects: we will not show all optional keywords; we will insert (nonessential) words like RECORD and SET to emphasize the semantic function in some cases:

1. To access a record by database key we write

   ```
   FIND  <record type> RECORD BY DATABASE-KEY <variable>
   ```

   where <variable> is in the workspace and holds a database-key as its value.
2. To find a record given value(s) for its index, we pass the value(s) to FIND by placing them in the field(s) in the correct template; then we execute

   ```
   FIND <record type> RECORD BY CALC-KEY
   ```

3. To find all records of one type with given value(s) in an index, we can use (2) to find the first one, then execute in a loop:

   ```
   FIND DUPLICATE  <record type> RECORD BY CALC-KEY
   ```

4. To scan a set occurrence for the member records, one at a time, we first use

   ```
   FIND OWNER OF CURRENT <set name> SET,
   ```

   which makes the owner of that set the CRU and the current of <set name>. Then the statement

   ```
   FIND NEXT <record type> RECORD IN CURRENT <set name> SET
   ```

   goes one more link around the ring from the current of <set name>, setting a system variable FAIL to "true" if the next record found is a set owner.
5. To find the current of any DBTG set or record type, which is sometimes needed to set the CRU before a GET operation, we can use:

   ```
   FIND CURRENT OF <set name> SET   or
   FIND CURRENT OF <record type> RECORD
   ```

Example 1:
To print all the (unknown number of) prices for the brush item, we could execute:
print "Brush prices"/*header*/

```
O.ITEM.ITEM: = "Brush"
FIND OFFERS RECORD BY CALC-KEY
while (NOT FAIL) do begin
  GET OFFERS;  PRICE
  print OFFERS.PRICE
  FIND DUPLICATE OFFERS RECORD BY CALC-KEY

end
```

Example 2:
The following code finds and prints all items ordered for use by the "Zanies" department. We start by finding the DEPT record for "Zanies" and this record owns the USED_BY set occurrence we need to scan, in which ring the desired items are the members. Each time around the loop, including for the first item found, we process another link by printing its field INAME.

```
DNAME:= "Zanies"
FIND DEPT RECORD BY CALC-KEY/* assume
  index for DEPT is DNAME*/
FIND OWNER OF CURRENT USED_BY SET
FIND FIRST ITEM RECORD IN CURRENT
  USED_BY SET/*FIRST before NEXT*/
while (NOT FAIL) do begin
  GET ITEM;INAME
  print ITEM.INAME
  FIND NEXT ITEM RECORD IN CURRENT
    USED_BY SET

end
```

V. DATA MODIFICATION COMMANDS OF THE NETWORK LANGUAGE

Commands to insert or delete a record exist in the DBTG DML; the record must be the CRU, and the action is with respect to the list of records of its type and possibly for a certain set occurrence. As well, a command exists to alter, or modify values, in the CRU. Data modification is complicated in the DBTG setting because the user has a choice of *existence constraints* that the system will enforce. For example, if there is a DBTG set S with owner type T and member type T', one can choose that whenever a type T' record is created, it is also inserted in some S set occurrence. To do so, when the set S is declared one should add INSERTION IS AUTOMATIC and give a set selection clause, which we will skip for brevity here, that gives the details of selecting a particular occurrence.

Another constraint, RETENTION IS MANDATORY, means that once a record is in one set occurrence, it cannot be moved to another one; since this is the normal case we will accept it as the default case. However, OPTIONAL in place of MANDATORY would allow INSERT and REMOVE commands to shift a record's set occurrence.

A. STORE, INSERT, and REMOVE

To store a new record of type T in the database we create the record r in the template for type T and then use the command

```
STORE T
```

which adds r to the list of records of type T and makes r the CRU, the CRT of T, and the CST of any set of which T is owner or member type. If T is the member of a set S which is declared to have automatic insertion, then r becomes a member of one set occurrence for each of these sets, depending on the chosen set selection clause.

The opposite of AUTOMATIC insertion is MANUAL: If set S is declared this way, then members are *not* inserted into any occurrence of S when records are stored, and we must (later) "manually" insert them, by using INSERT commands, which exist mainly for that purpose. For simplicity, here we can assume that insertion is AUTOMATIC.

Insertion: To insert an existing record r of type T that is a member of set S into a certain set occurrence, we first make this occurrence the CST of S by previous commands. Then we make r the CRU, and execute

```
INSERT T INTO S
```

Deletion: To remove the CRU, a record of type T, from its set occurrence for S, we execute

```
REMOVE T FROM S
```

which does not affect the existence of this record in T. However, one cannot execute REMOVE T FOR S without an error if RETENTION IS MANDATORY has been declared for S. (In both of these commands, S could be substituted by any proper list of sets!)

B. Record Modification

The command MODIFY <record type> copies the template of the indicated <record type> into the current of run unit. If the CRU is *not* of that record type, an error occurs. One can also modify a subset of the CRU record's fields:

```
MODIFY <record type>; <field(s)>
```

If T is the record type for the CRU, the values of the listed fields are copied from the template into the listed fields of the CRU, but its other fields are left alone. A summary of major data modifying commands in the network DML appears in Table II.

Table III DML Commands

Data modification command	Default	Option
STORE	STORE T	STORE T; <field(s)>
INSERT	INSERT T INTO S	—
REMOVE	REMOVE T FROM S	—
MODIFY	MODIFY T	MODIFY T; <field(s)>

VI. LONG-TERM LEGACY OF NETWORK DATABASE SYSTEMS

A fair question about network database systems is: Why did these extensively developed systems became "legacy" systems? Any rationale about why systems became obsolete, other than the fact that they failed the do-or-die test of the marketplace, must be semisubjective. However, in the objective spirit of information systems as a science, we can make some comparative comments on network systems in general, and on CODASYL systems.

In general, networks are complicated; their structure is complex, and their operators are complex, in that they must deal with each record separately, one at a time, as we have seen. Still, even if these systems had set-level operators, as some hybrid systems do, their navigation would still be complex. Their greater complexity, compared to relational systems, does not yield any additional functionality.

It was once claimed that network systems at least had good performance. That claim has been undone by three considerations:

1. Network data structures tend to fragment information. For example, the supplier number and name for a particular Order in a typical NDB structure is not part of the order record; instead it is in the supplier record that owns that order record's set occurrence.
2. Most of the optimizing of such systems had to be done manually, because database statistics did not support automatic optimization. Manual efforts are very costly.
3. Later relational systems undid early perceptions of better performance by networks.

While it is not surprising that network database systems were eventually superceded by technically superior database systems, these early systems provided crucial software for the coordination of pioneering space exploration, and for many other purposes. In fact, many of the business-related legacy systems still function today, and are being supported, having survived the fabled crisis of the year 2000 problem.

SEE ALSO THE FOLLOWING ARTICLES

Database Administration • Distributed Databases • Local Area Networks • Mobile and Wireless Networks • Network Environments, Managing • Wide Area Networks

BIBLIOGRAPHY

CODASYL (1971). *Report of the CODASYL data base task group.* New York: Association for Computing Machinery.

Date, C. (1990). *An introduction to database systems,* 5th ed. Reading, MA: Addison-Wesley.

Ozkarahan, E. (1990). *Database management: Concepts, design & practice.* Englewood Cliffs, NJ: Prentice-Hall.

Tsichritzis, D. C., and Lochovsky, F. H. (1982). *Data models.* Englewood Cliffs, NJ: Prentice-Hall.

Ullman, J. D. (1988). *Principles of database and knowledge-base systems, Vol. I.* Rockville, MD: Computer Science Press.

Network Environments, Managing

Ray Hunt and John Vargo
University of Canterbury, Christchurch, New Zealand

GLOSSARY

abstract syntax notation (ASN.1) A standard way to describe a message that can be sent or received in a network. ASN.1 is divided into two parts: (1) the syntax rules for describing the contents of a message in terms of data type and content sequence or structure and (2) how data items are encoded in a message. ASN.1 is defined by two ISO standards (ISO 8824/ITU X.208 specifies the syntax and ISO 8825/ITU X.209 specifies the basic encoding rules for ASN.1).

active directory service interface (ADSI) The interface to Microsoft's trademarked Active Directory which is an integral part of the Windows 2000 architecture. Like other directory services, such as Novell Directory Services (NDS), Active Directory is a centralized and standardized system that automates network management of user data, security, and distributed resources, and enables interoperation with other directories. Active Directory is designed especially for distributed networking environments.

application program interface (API) The specific method prescribed by an operating system or application program by which a programmer can make requests of the operating system or another application. An API can be contrasted with a graphical user interface or a command interface as alternative interfaces to an operating system or a program.

asynchronous transfer mode (ATM) A dedicated-connection switching technology that organizes digital data into 53-bytecells and transmits them over a physical medium using digital signal technology. Individually, a cell is processed asynchronously relative to other related cells and queued before being multiplexed over the tranmission path. Because ATM is designed to be easily implemented by hardware, faster processing and switch speeds are possible. The prespecified bit rates are either 155 or 622 Mbps.

common information model (CIM) A nonproprietary system used to describe overall management information. It is used to represent managed real-world objects using object-oriented paradigm concepts of classes and instances.

common management information protocol (CMIP) A network management protocol built on the open systems interconnection (OSI) communication model. The related common management information services (CMIS) defines services for accessing information about network objects or devices, controlling them, and receiving status reports from them.

common management information service (elements) (CMIS/CMISE) See CMIP.

common object request broker architecture (CORBA) An architecture and specification for creating, distributing, and managing distributed program objects in a network. It allows programs at different locations and developed by different vendors to communicate in a network through an "interface broker."

(distributed) component object model (COM/DCOM) A set of Microsoft concepts and program interfaces in which client program objects can request services from server program objects on other computers in a network. DCOM is based on the com-

ponent object model (COM), which provides a set of interfaces allowing clients and servers to communicate within the same computer.

desktop management task force (DMTF) The DMTF is an industry organization that is providing the development, adoption, and unification of management standards and initiatives for desktop, enterprise, and Internet environments. Working with technology vendors and affiliated groups, the DMTF endeavors to provide an integrated and cost-effective management through interoperable management solutions.

directory enabled networking (DEN) An industry-standard initiative and specification for how to construct and store information about a network's users, applications, and data in a central directory. A standard way of describing the network's elements in a central repository can enable applications to be developed that will automatically learn of user access privileges, bandwidth assignments, and the company's resource policies, providing services accordingly. DEN defines an object-oriented information model that is based upon CIM (see above). Both models are being mapped into the directory defined as part of LDAP (see below). DEN and CIM are an advance over and can be used with SNMP (see below).

distributed computing environment (DCE) An industry-standard software technology for setting up and managing computing and data exchange in a system of distributed computers and is typically used in a larger network of computing systems that include different size servers geographically scattered. DCE uses the client/server model and application programmers need not be aware of where their programs will run or where the data will be located.

(Microsoft's Windows) distributed interNet applications (DNA) Microsoft's Web Solution Platform—DNA—is a three-tier model comprised of business, data, and presentation tiers. The business tier focuses on providing business services using Microsoft-specific technologies. The data tier uses active directory service interfaces (ADSI) as an API to the active directory (AD) while the presentation tier uses XML and HTTP to encode and transport the data in a vendor neutral fashion.

distributed processing environment (DPE) An infrastructure for configuring and managing software. It consists of several components that permit the start-up, monitoring, and shut-down of software services on a network.

integrated services digital network (ISDN) A set of CCITT/ITU standards for digital transmission over telephone copper wire and other media. ISDN requires adapters at both ends of the transmission and can operate at spends of multiple 64 kbps blocks (up to 1.544 or 2.048 Mbps). It is normally offered as a Telco service.

interface definition language (IDL) A generic term for a language that lets a program or object written in one language communicate with another program written in an unknown language. In distributed object technology, it is important that new objects be able to be sent to any platform environment and discover how to run in that environment.

international standards organization (ISO) A worldwide federation of national standards bodies from around 100 countries. Among the standards it fosters is open systems interconnection (OSI), a universal reference model for communication protocols. Many countries have national standards organizations such as the American National Standards Institute (ANSI) that participate in and contribute to ISO standards making.

Internet Activities Board (IAB) Is the Internet Society overseer of the technical evolution of the Internet. The IAB supervises the internet engineering task force (IETF), which oversees the evolution of TCP/IP, and the internet research task force (IRTF), which work on network technology.

lightweight directory access protocol (LDAP) A software protocol for enabling anyone to locate organization, individuals, and other resources such as files and devices in a network, whether on the Internet or on a corporate intranet. LDAP is a lightweight version of directory access protocol (DAP), which is part of X.500, a standard for directory services in a network.

management functional areas/system management functions (MFA/SMF) Specifies the functions that can be performed resulting from using the five key management organization groups of: configuration, fault, performance, accounting, and security management.

management information base (MIB) A formal description of a set of network objects that can be managed using SNMP. The format of the MIB is defined as part of the SNMP. MIB-1, MIB-II refer to various versions used with SNMPv1, v2, v3, etc.

multi router traffic grapher (MRTG) A tool to monitor the traffic load on network-links. It generates HTML pages containing GIF images which provide a live visual representation of this traffic and is based on Perl and C operating under UNIX and Windows NT.

object management group (OMG) A nonprofit consortium that produces and maintains computer in-

dustry specifications for interoperable enterprise applications. Most of the companies that shape enterprise and internet computing today are represented. Best-known specifications include CORBA and others. (www.omg.org)

object request broker (ORB) A program that acts as a broker between a client request for a service from a distributed object or component and the completion of that request. Having ORB support in a network means that a client program can request a service without having to understand where the server is in a distributed network or exactly what the interface to the server program looks like. Components can find out about each other and exchange interface information as they are running.

open system interconnection (OSI) A standard description or reference model for how messages should be transmitted between any two points in a telecommunication network. Its purpose is to guide product implementers so that their products will interwork. The reference model defines seven layers of functions that take place at each end of a communication.

policy-based networking (PBN) The management of a network so that various kinds of traffic—data, voice, and video—get the priority of availability and bandwidth needed to serve the network's users. PBN ensures that one kind of servce does not preempt another and using policy statements, network administrators can specify which kinds of service to give priority at what times of day on what parts of their network. This kind of management is often known as *quality of service* and is controlled using policy-based network software.

protocol data unit (PDU) Used by the ISO-OSI protocol specifications, PDUs are units of data used to exchange information (user and control) between entities in a network. Common units include bit (layer 1), frame (layer 2), packet (layer 3), segment (layer 4).

Q3/Qx Interface in CMIP model and specify communication between operating system function layers. Q3 is defined by the ITU Q.812 standard. Devices that are not Q3 conformant must communicate via the Qx interface which is normally proprietary for the specific equipment being used.

remote monitoring management information base (RMON) Provides standard information that a network administrator can use to monitor a group of distributed LANs from a central site. RMON specifically defines the information that any network monitoring system will be able to provide. It is specified as part of the MIB and SNMP.

remote procedure call (RPC) A protocol that one program can use to request a service from a program located in another computer without having to understand network details. The requesting program is a client and the service-providing program is the server. RPC is a synchronous operation requiring the requesting program to be suspended until the results of the remote procedure are returned, although RPCs can be processed concurrently.

request for comment (RFC) A formal document from the internet engineering task force (IETF) that is the result of committee drafting and subsequent review by interested parties. Some RFCs are informational in nature. Of those that are intended to become internet standards, the final version of the RFC becomes that standard and no further comments or changes are permitted. Change can occur through subsequent RFCs that supersede or elaborate on previous RFCs.

Resource reSerVations Protocol (RSVP) A set of communication rules that allows channels or paths on the interent to be reserved for the transmission of video and other high-bandwidth messages. RSVP is part of the internet integrated service (IIS) model, which ensures best-effort service, real-time service, and controlled link-sharing.

simple network management protocol (SNMP) The protocol governing network management and the monitoring of network devices and their functions. It is not necessarily limited to TCP/IP networks although this is its primary focus. SNMP exists in three versions and is described formally by the IETF RFCs.

structure of management information (SMI) To support the heterogeneous network environment object attributes and protocol data units must be machine independent and well publicized. The IAB has thus defined the SMI and published the details as RFCs. The SMI describes a structure that contains management information and is specified using ASN.1 notation to achieve machine independence.

synchronous digital hierarchy (SDH) A standard technology for synchronous data transmission on optical media. It is the international equivalent of synchronous optical network. Both technologies provide faster and less expensive network interconnection than traditional PDH (plesiochronous digital hierarchy) equipment.

system management functions (SMF) See MFA.

telecommunication management network (TMN) An ITU-based strategic goal to create or identify standard interfaces that would allow a network to be managed consistently across all network element suppliers. The concept has fostered a series

of interrelated efforts at developing standard ways to define and address network elements.TMN uses the OSI management standards as its framework and it applies to both fixed and wireless networks.

telecommunications information networking architecture (TINA) Incorporates the philosophies of TMN (see below) within an architecture designed to address the distributed nature of modern telecommunications networks. Like TMN, TINA focuses on an object-orientated design which modularizes systems into manageable components and a distributed network of software components which can accommodate traffic flow, traffic load, and reliability requirements.

transmission control protocol/internet protocol (TCP/IP) The basic communication language or protocol of the internet. It can also be used as a communications protocol in a private network (intranet or extranet). TCP/IP is a two-layer program. The higher layer (TCP), manages the assembling of a message or file into smaller packets that are transmitted over the internet and received by a TCP layer that reassembles the packets into the original messages. The lower layer, internet protocol, handles the address part of each packet.

eXtensible markup language (XML) XML is a flexible way to create common information formats and share both the format and the data on the Web, intranets, and elsewhere. For example, computer makers might agree on a standard or common way to describe the information about a computer product (processor speed, memory size) and then describe the product information format with XML. Such a standard way of describing data would enable a user to send an intelligent agent (a program) to each computer maker's Web site, gather data, and then make a valid comparison. XML can be used by any individual or group that wishes to share information in a consistent way.

With their increased power and versatility, today's computer networks, are becoming used for an ever-growing variety of applications. Further, the networking structures are now very complex and diverse. For example, network management systems are required to manage various structures such as simple subnets, integrated gigabit campus-based local area networks (LANs), routing infrastructures, company intranets, extranet infrastructures, and a wide variety of corporate and telco-based networks. Further, the technologies are very diverse. Although there is considerable emphasis on transmission control protocol/Internet protocol (TCP/IP) and Web-based architectures today, such architectures are by no means universal. We do not want to end up managing "islands of architectures" which existed with legacy operating systems some years ago. Integration of these different applications, infrastructures, business requirements, and technologies is a significant challenge today.

With this growth and development comes the requirement for a high level of service and performance. It is essential to be able to offer an integrated management infrastructure which encompasses a range of platform, structural, and geographical boundaries.

This article commences by examining network management infrastructure addressing the five basic management concepts: configuration management, fault management, performance management, accounting management, and security management. The traditional open system interconnection (OSI) model is then developed to illustrate the functional, organizational and informational viewpoints extending these ideas by introducing abstract syntax notation (ASN), International Standards Organization (ISO) registration trees, and other basic conceptual tools.

The history and evolution of network management infrastructure is discussed with reference to the various protocols, particular the simple network management protocol (SNMP) and common management information protocol (CMIP) families. TCP/IP network management is crucial to the operation of many intranets and extranets today and this is discussed at length. The structure of TCP/IP management information is illustrated by way of management information bases(MIBs), remote monitoring MIBs (RMON), trees, etc. These concepts are then extended to examine other types of networks such as LANs, router-based networks, corporate networks and telco networks. The telecommunication management network (TMN) hierarchy is introduced and the various layers—business, service, network, and element management—are examined.

This article examines a wide variety of network management infrastructures and addresses architectures such as TMN, telecommunications information networking architecture (TINA), telco and corporate networks. This article further examines state-of-the-art technologies such as: managed objects, SNMPv3, CMIP, Q3 interfaces, TMN, TINA, and policy-based management.

I. INTRODUCTION

A. Network Management

The management of today's networks poses some complex problems. The initial decision of scope is a determinant of management complexity. Networking technology can be viewed using the OSI seven-layer model (Fig. 1), and it is important to ask how many of these layers should we manage.

Most networks are managed up to the transport layer; however, much has been written about the "cost of ownership" and "desktop management" which implies management higher up the architecture. Once a decision about scope has been made, the approach becomes better defined. The ISO model separates network management into five specific areas: configuration, fault, performance, accounting, and security. Each area has its own special requirements. These functional management areas are discussed in more detail in Section II.

B. Management Complexity

Many organizations have several specialized networks dedicated to specific business areas. For smaller organizations a simple LAN or subnet will often suffice. As the organization grows, it finds itself having to merge several subnets using Wide Area Network (WAN) technology (such as routers or switches). This upgrade requires the network manager to make a number of decisions about network architecture, protocols, and management. Very large organizations often have complex networks that require careful management to ensure continued reliable operation. The most complex networks are those operated and maintained by telecommunications providers. These key network architectures are discussed in Section IV.

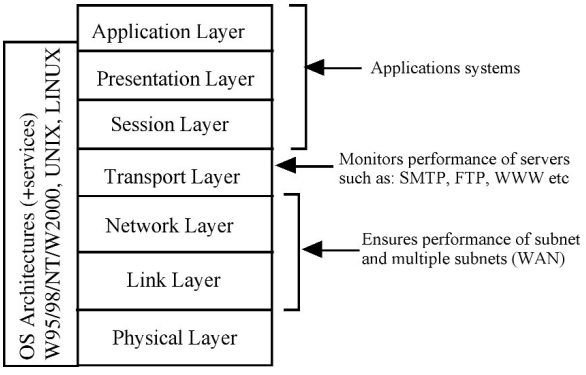


Figure 1 Management scope and the OSI seven-layer model.

C. Networking Technology

As networking became more diversified, the strategies for managing it had to become more complex. Simple strategies such as SNMP evolved into approaches such as RMON, RMON2, SNMPv2, and, more recently, SNMPv3. Each of these approaches has improved upon prior versions to provide more powerful tools for the network manager. Alternative management approaches such as CMIP have offered more power, but have demanded a heavy toll in terms of network bandwidth, implementation effort, and complexity. As networks and processors become faster, these approaches are beginning to find favor in systems such as TMN. Several alternative approaches attempt to incorporate existing technologies (such as SNMP) into Web-based browser interfaces, while other approaches involve adding network management functionality to well-established languages such as Java, tcl/tk, or C. A discussion of the two main approaches to network management is given in Section III, while Section IV overviews some of the more complex network infrastructures. Finally, the various enabling technologies for network management are discussed in Section V.

II. FUNCTIONAL NETWORK MANAGEMENT

A. Management Functions

Historically, network management required the detection and rectification of faults. As the size and complexity of networks have grown, such a simple approach is no longer appropriate. Outages in modern network infrastructure can have an unacceptable cost. Additionally, with the speed of modern desktop computers the network is becoming a potential bottleneck. Thus, network performance must be carefully monitored. Intelligent, proactive management strategies are required to ensure that the network meets users' objectives. The ISO network management model partitions the functions of network management into five conceptual areas:

- Configuration management
- Fault management
- Performance management
- Accounting management
- Security management

1. Configuration Management

Configuration management exists to collect and monitor configuration information so that the effects of changes in hardware and software can be managed. Once collected, this information provides the raw data from which a topographical view of the network may be constructed. This view would typically include information such as:

- *Devices* in the network, their version, location, and unique identifiers
- *Cabling* in the network, its capacity and location
- *Interconnections* or physical relationships between devices

This information is typically presented via a graphical user interface. In the event of part of the network failing, the configuration of the network must be altered to provide alternative paths for the flow of information (rerouting), thus avoiding interruption of service.

2. Fault Management

This function is required to detect abnormal network behavior. Fault management follows a sequence of actions: error detection, error diagnosis, and error recovery.

Error detection monitors events such as alarm signals from network devices (when thresholds are exceeded or in the event of hardware failure), deterioration of performance, or application failures. Error detection facilities also include an error log for future analysis.

Error diagnosis involves the analysis of detected errors in an effort to determine the cause of an error and a course of action to rectify it. Recent approaches to error diagnosis include the use of artificial intelligence techniques such as deductive reasoning.

Error recovery involves a range of measures proportional to the error's magnitude. Simple errors may require the fine-tuning of a device on the network, where more serious errors may mandate the replacement of a faulty device. Persistent performance failures are usually an indicator of poor network health. Remedying such problems typically involves reconfiguration of the problematic section of the network.

3. Performance Management

Performance management is central to the long-term management of the network. By gathering statistical data about the behavior of managed objects and traffic flows between them, trends in network performance can be predicted.

Using analytical modeling potential bottlenecks may be discovered and scenarios (such as increased traffic at various points of the network) may be assessed. Thus, performance modeling provides valuable feedback on both the short- and long-term health of the network. It facilitates the proactive upgrading and reconfiguration of the network to meet the changing needs of the users.

4. Accounting Management

Accounting management is primarily concerned with determining the utilization of network services by individuals or groups of users and regulating such usage according to policy. By regulating usage according to available resources, network services may be apportioned fairly among users while also reducing network congestion.

The most immediate application of such facilities is commercial, i.e., charge the user based on their usage. A second dimension is to identify the urgency of service required by users and to charge them according to demand. Such quality of service options provide users with increased flexibility for their communications requirements.

5. Security Management

Network security requires that access to network resources be controlled by policies to prevent (intentional or accidental) sabotage. Furthermore, access to sensitive information should be restricted to those with appropriate authorization. Security management involves:

- Definition of user sets (varying levels of authorization)
- Identification of sensitive network resources
- Mapping sensitive network resources to user sets
- Monitoring access points (firewalls are a security subsystem)
- Logging unauthorized access attempts and intrusion detection

III. NETWORK MANAGEMENT ARCHITECTURES

Figure 2 shows a typical network management architecture. The network management system at the top of the figure contains a management entity, usually an application responsible for automation at the lower level and assisting in higher level network management. A managed device can be anything that imple-

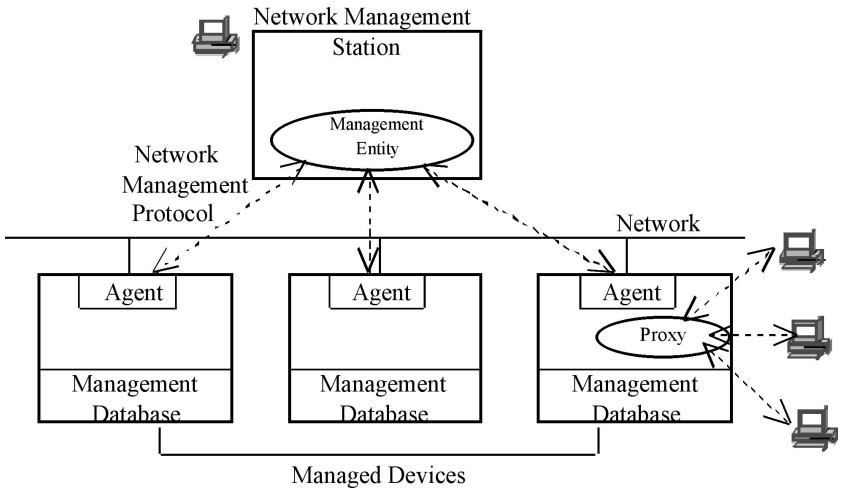


Figure 2 Typical network management architecture.

ments the management protocols. These devices must also contain an agent which is responsible for maintaining a local database of information about the device, responding to requests, and sending messages. While the structure of the local database is well defined, the interface between the database and the agent is implementation dependent. On the bottom right of the figure are three devices that are attached to a managed device via a proxy agent. Proxy agents provide a means of managing foreign devices that do not support the standard transport or management protocols.

A. Open System Interconnection Network Management

1. The Open System Interconnection Management Model

The OSI management model generalizes Figure 2 to three separate viewpoints of management:

- An organizational viewpoint
- An informational viewpoint
- A functional viewpoint

The *organizational* viewpoint allows the topology and scope of the network to be defined. It allows the OSI abstractions to be mapped onto real-world entities.

For example, Fig. 3 shows a small router network consisting of two LANs. The two sites in this example are managed by two organizations. Organization A has drawn this diagram to illustrate the topology of their network. The scope of their management system is indicated by the dashed outline.

The *informational* viewpoint specifies the aspects of entities or managed objects in the network. These managed objects are defined in terms of their attributes and the operations that can be performed upon them. The collection of objects and their attributes make up the MIB.

Continuing with the sample network, Fig. 4 shows the four managed entities within LAN A; each entity has its own attributes and operations and this corresponds to an object-oriented modeling approach. The standard ISO10040/ITU X.701 is a formal definition of the objects in a MIB and is expressed using the ASN.1 and basic encoding rules (BER) widely used to formally describe data structures.

Three independent tree-like structures are used to provide the required level of formalism and rigor to describe OSI-managed objects as shown in Fig. 5.

- The ISO registration tree
- An inheritance tree which shows how object classes are derived from other classes
- A containment tree which defines an object in terms of other objects in a hierarchy

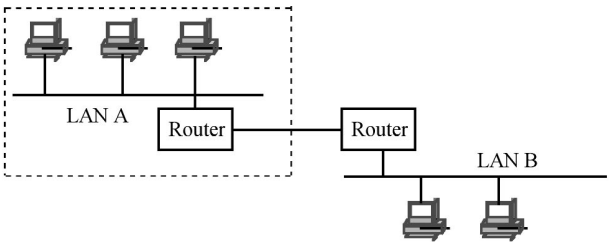


Figure 3 An example of the scope and topology of a small network.

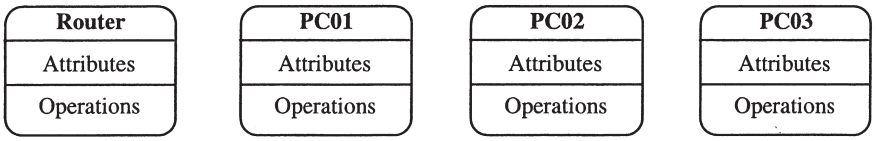


Figure 4 An example of managed objects within LAN A.

The ISO registration tree provides a library of well-defined, registered, basic object class definitions that may be used to build new class descriptions using the object-oriented approach of reuse. The inheritance tree provides an object hierarchy from a system-wide context, while the containment tree provides an object hierarchy in the context of a single managed object. The containment tree essentially defines the MIB structure.

The *functional* viewpoint defines five specific management functional areas (SMFAs) that correspond directly to the five functions of management discussed in Section II.A. One of these (performance management) is shown in Fig. 6.

Returning to the previous example, Fig. 6 shows how performance measurement is achieved with this architecture. One of the personal computers (PCs) in the LAN runs a network management system. An important part of this system is performance management. Performance management is achieved using a subset of systems management functions (SMFs), each of which map onto common management information service elements (CMISE). These elements may be thought of as simple subroutines or functions, and OSI management specifies groupings in order to accomplish more complex tasks. Each CMISE obtains information from a managed entity using a communications protocol, in this case it is the CMIP which is discussed further in Section V.D. The managed entity returns information (such as interface speed, number of bytes in/out, errors) to the managing entity (e.g., IBM's Tivoli) via CMIP. The common management information service (CMIS) then returns its results to the SMF, which in turn provides information to the high-level performance management application.

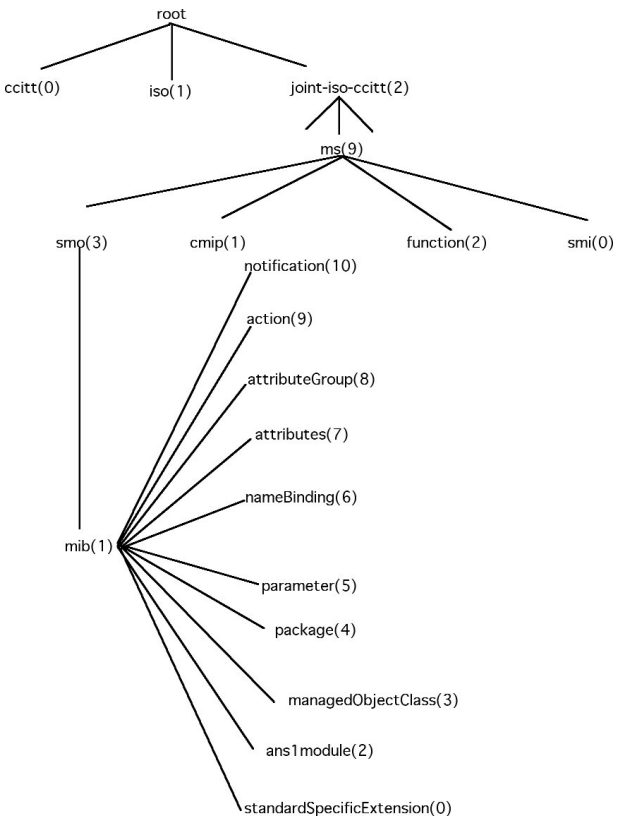


Figure 5 ISO registration tree.

Figure 7 shows the classical view (from ISO10040/ITU X.701) of how the high-level view of management functionality is broken down into low-level functions. Each management functional area (MFA) (see Section II) may be implemented using one or more SMFs (for example, the security management MFA requires the security alarm reporting SMF). To date, 13 SMFs have been specified; each is defined in terms of low-level CMISE. The CMIS defines seven primitives, each of which includes a standard modifier (request, indication, response, confirm). These are used by the SMFAs. The CMIS must communicate with managed entities and it does so using the CMIP. The CMIP mandates the procedure and description of 10 primitives and is discussed further in Section V.D.

B. Transmission Control Protocol/Internet Protocol Network Management

1. The Transmission Control Protocol/Internet Protocol Management Model

The TCP/IP management model generalizes Fig. 2 to two basic components: objects and a communication mechanism. In TCP/IP network management objects

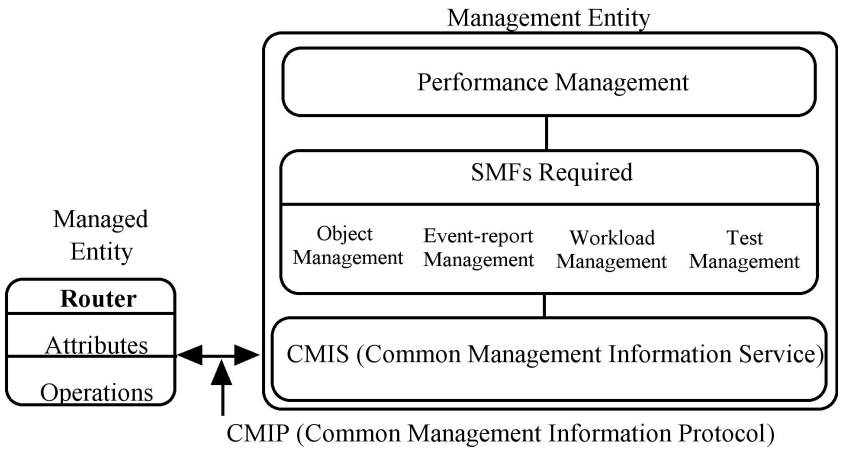


Figure 6 An example of the performance measurement function.

are defined in terms of their attributes such as packets per second, error rate, etc.; this corresponds to an informational viewpoint as discussed in Section III. A. Communication of attributes via a common protocol in a well-defined fashion corresponds to a functional viewpoint as was discussed in Section II.

To support the heterogeneous network environment the object attributes and protocol data units must be machine independent and well publicized. To this end the Internet activities board (IAB) has defined the structure of management information (SMI) and published the details of this as request for comments

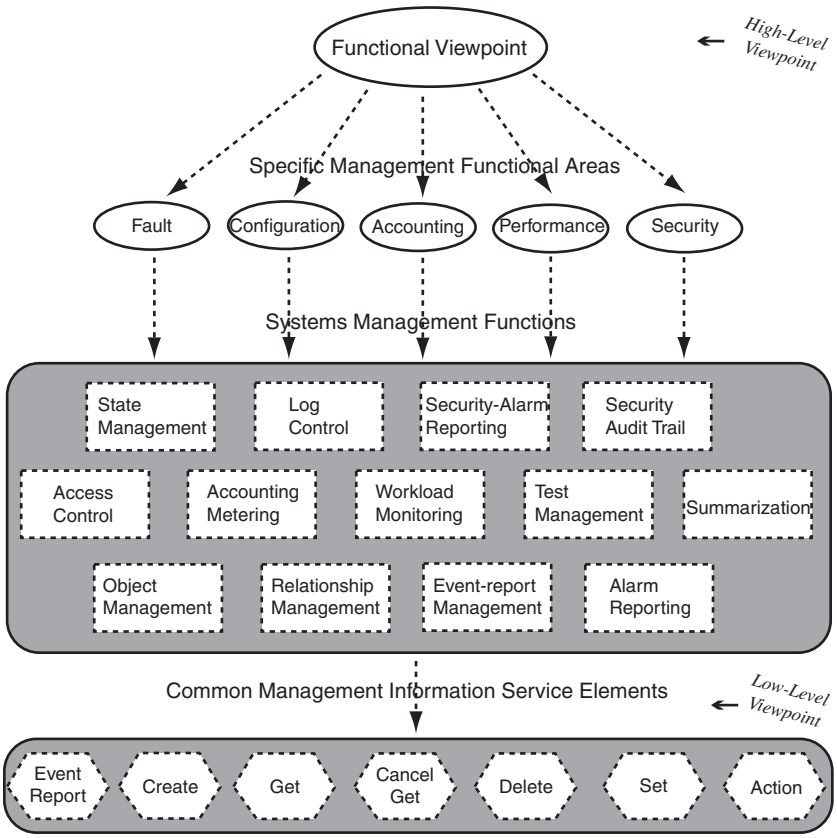


Figure 7 OSI network management: functional viewpoint.

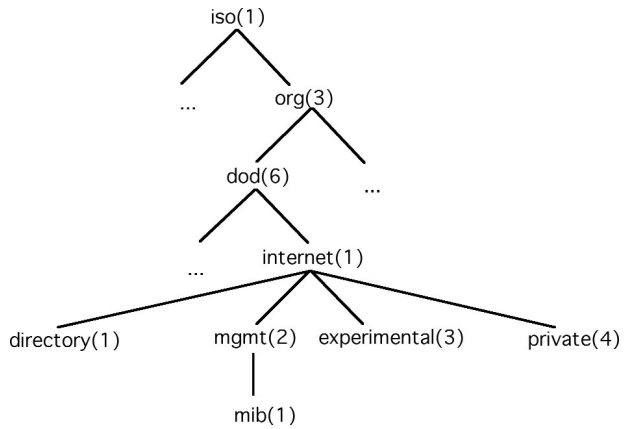


Figure 8 The structure of management information—an overview.

(RFCs). The SMI describes a structure that contains management information; this description is written using ASN.1 to achieve machine independence.

Figure 8 shows that SMI is effectively a tree data structure containing management information. Objects in this tree are identified by specifying the path to the object using a "dot" notation, for example, the "MIB" object may be defined as iso.org.dod.internet.mgmt.mib or 1.3.6.1.2.1.

The part of the SMI identified as the MIB contains management information for many devices. The information contained within the MIB is compiled from the ASN.1 definitions published as various RFCs. For example, RFC1156 contains the definition for the first defined MIB (MIB-I). Objects within the MIB are grouped into relevant groups. As new technology appears, RFCs may be published defining additional objects to be added to this structure. This extensibility provides future proofing for the SMI concept.

The functional viewpoint addresses the five SMFAs discussed in Section II.A. It does so by providing just five functions within SNMP; this minimal approach ensured that SNMP was kept both simple and easy to implement. Unfortunately, it also resulted in some deficiencies. SNMP is further discussed in Section V.B.

IV. MANAGING NETWORK INFRASTRUCTURES

A. Size Does Matter

The complexity of a network is affected by factors such as size, scope, technology, and heterogeneity. Enlarging a network will typically result in substantial increases in network traffic leading to congestion and costly delays. Unless properly adapted, management techniques will quickly become ineffective in this new environment.

The larger the scope of a network management policy, the more things there are to be managed. A simple policy may involve maintaining network performance and reliability; however, in a large network such an ad hoc approach quickly becomes problematic as technicians find themselves struggling to fix problems in a timely fashion. Network managers will need to carefully define the scope of a management system to determine the most efficient and effective approach.

As networks evolve they may incorporate new technologies. An Ethernet LAN may be split into two smaller LANs connected by a router or a switch. So in addition to the Ethernet technology routing or switching technologies have to be managed. Network managers must decide upon a solution that is able to scale well by comfortably incorporating new technologies or be confident the network will not grow beyond a well-specified limit. The latter approach may be advocated by the cost conscious, but this relies on estimates of a network's future use. With the technological growth of today, such fortune-telling is guesswork at best.

The following sections outline some of the more common network infrastructures beginning with the humble LAN and ending with the complex networks maintained by a telecommunications company.

B. Local Area Networks

An LAN is probably the simplest network to manage. Due to its localized nature a LAN typically utilizes a small number of transmission technologies. This clustering also eases the tasks of monitoring and fault detection. Furthermore, an LAN is typically the responsibility of one organization which alleviates the "delineation of responsibility" problem.

It is important to realize that LANs have special management needs. Many organizations will invest in an LAN to streamline intraoffice communications or to share resources (printers, faxes, databases). With these increased efficiencies jobs can be completed more quickly, so the organization rapidly becomes dependent upon the LAN to complete its day-to-day operations.

As the scope of the LAN is expanded to meet user demand, its complexity significantly increases. It is imperative that the network be well managed to cope with this growth.

C. Router Networks

Informally, a router network is a collective of subnetworks that may be owned by different entities (thus, management delineation may be problematic) and connected over some medium that may be owned by yet another entity.

When several subnetworks need to be joined, the connection may be made using a switch or a router. Switches may only be used to connect subnets that use similar protocols (they operate at layer two of the OSI model), whereas routers may connect subnetworks that use different LAN technologies (they operate at layer three of the OSI model).

Switching technology was considered to be both complex and wasteful of resources. Lately, with the introduction of modern switching hubs in the LAN, switching has been shown to be faster, simpler, and cheaper than routing. Switching does not involve the disassembly and reassembly associated with internet protocol (IP) routing, instead it switches packets at very high speed. Unfortunately, switches are unable to process frames and control their distribution, leading to problems with scalability and management. They also cannot recognize layer three protocols such as Appletalk, IPX, and DECnet.

The versatility of a router makes it an attractive solution; however, with careful design a network may be constructed using a combination of switches (for subnet segments) and reserving routers for points where they are strictly necessary.

D. Telco Networks

1. The Element Level

At the lowest level of a telecommunications network lies the individual network components such as switches, routers, hubs, and interface equipment; each is suited to a particular task. Components may be manufactured by one or more suppliers, each offering their own proprietary software designed more to allow technicians to troubleshoot the component rather than to provide information useful for higher level management. Management of these "elements" requires that each component be continuously monitored to ensure its optimal performance. Element management software must be able to monitor and regulate many network components and produce information that can be utilized by higher level management applications.

2. The Network Level

Figure 9 shows a typical collection of telco networks, each of which may utilize elements from several manufacturers. Some networks may be functionally dependent upon others (in our example, the satisfactory operation and hence management of the frame relay network depends upon the operation of the underlying asynchrome transfer mode [ATM] network, which in turn is dependent upon the underlying synchronous digits hierarchy [SDH] network and the various vendors systems which make up this lowest level). A network management tool at the frame relay network level which does not take account of the underlying network infrastructures will be inadequate. Primary concerns for each network include issues such as connectivity, performance, load balancing, alternative routing, etc. Information from individual network elements must be available to provide a picture of the overall state of the frame relay network. Using this information, network management systems can make decisions about how to reroute traffic to alleviate network congestion or alert personnel of more serious problems.

3. The Service Level

A service provided to customers may incorporate several networks. Key concerns at this level of management include cost reduction, customer service, and rapid creation of new services. Such a system must also provide service personnel with the ability to create, monitor, and maintain services for individual customers. Additionally, this abstraction permits the

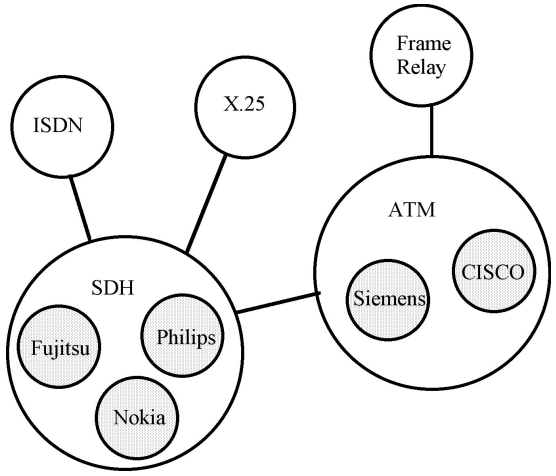


Figure 9 A telco network architecture.

exchange of underlying network technology, for example a customer that finds their service to be too slow over the frame relay network may be quickly changed over to an ATM service.

A typical telecommunications service is shown in Fig. 10. In this example the customer purchases a frame relay service, but the performance of this service depends upon the networks that the service utilizes. Thus, several networks must be monitored to ensure that performance criteria of the service are met.

Monitoring of services is essential to ensure that customers expectations (often in the form of service level agreements) are being consistently met. Service maintenance ensures that if quality can be improved or if a problem occurs, the necessary improvements are made without delay.

4. The Business Level

Finally, a management system must provide some support for business management. Decisions about which services are successful, profit and cost centers, and overall business performance must be supported by the system.

5. The Telecommunications Management Network

With the diverse requirements of each management level, how can one system provide the support mandated by an organization?

Using the examples from Figs. 9 and 11 and combining them yields Fig. 11. Where the lowest layer contains the network elements, the second layer represents network and service elements, and the top layer contains the services. This layering allows the design of an architecture that mandates software operating at specific levels and communicating between adjacent levels via a well-defined interface. Such architecture is defined by the TMN. The TMN is a hierarchy that distributes functionality over five levels (Fig. 12).

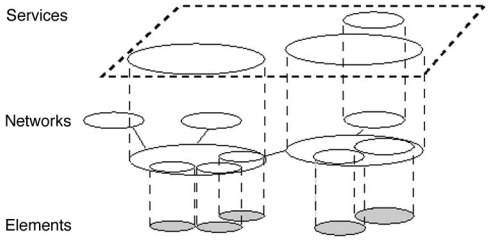


Figure 11 A hierarchy of management services.

The lowest layer contains the network elements (such as routers or switches). Management information pertaining to network elements percolates up to the network management layer which provides the five functions of management network wide.

Various networks, subnetworks, and elements may be integrated at this level. The service management layer allows us to view a collection of network elements as a service (for example, as shown in Fig. 10; frame relay is a service level view of the ATM network). At the top of the framework is the business management layer which provides a management view composed of services.

Communication between each layer of the TMN hierarchy is shown in Fig. 13. Each layer contains an operating system function (OSF) which processes information for monitoring and/or controlling purposes. The communication between OSFs within the four upper TMN layers is defined by the standard ITU Q.812

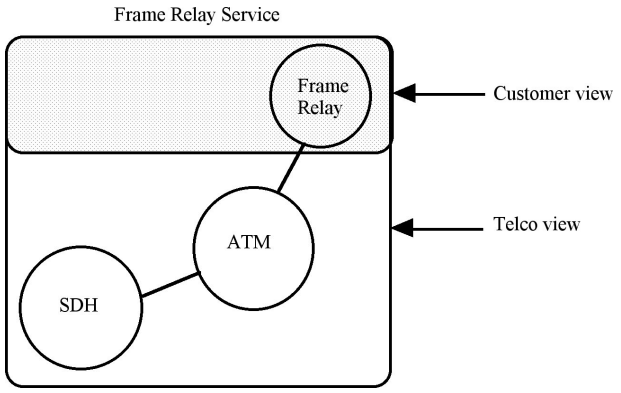


Figure 10 A typical telecommunications service.

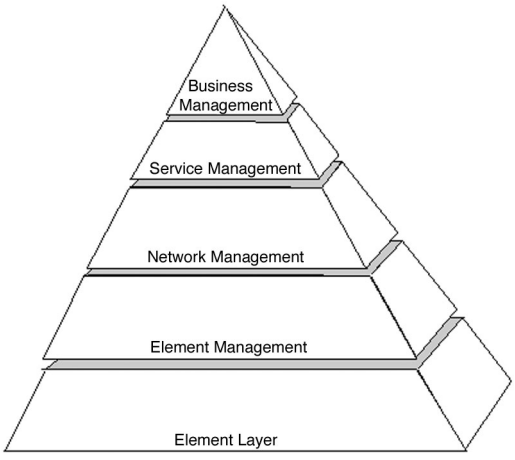


Figure 12 TMN hierarchy.

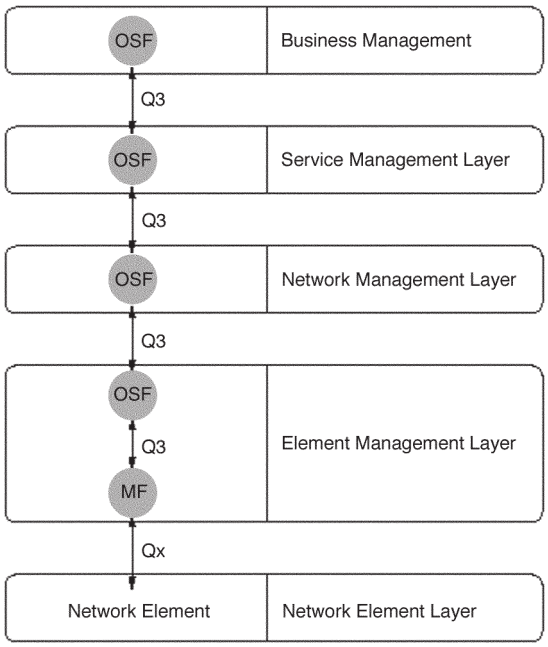


Figure 13 The Q3 interface.

and is know as the Q3 interface. Communication between the element (bottom) layer and the element management layer can be achieved via either the Q3 or Qx interface. Devices that are not Q3 conformant must communicate via the Qx interface, and these specifications are discussed further in Section V.E.

There are many TMN products and vendors and some of the better known include NetExpert *OSI,* TeMip *Digital,* Openview *HP,* Tivoli *IBM,* and ISM *BULL.* With a legacy of proprietary service management, telcos may use TMN to realize effective cost management. Willets and Adams (1996) provide an excellent treatment of this topic in *The Lean Communications Provider.*

E. Telecommunications Information Networking Architecture

TINA incorporates the philosophies of TMN within a detailed architecture designed to address the distributed nature of modern telecommunications networks. Like TMN, TINA focuses on:

- An object-orientated design which modularizes systems into manageable components
- A distributed network of software components which can accommodate traffic flow, traffic load, and reliability requirements

Like TMN, TINA is an architecture which supports interoperability, portability, and reuse of software components, as well as independence from specific technologies. It specifies a layered architecture which separates applications, services, resources, and elements using well-defined interfaces. TINA adds to TMN by detailing the *distributed processing environment* (DPE).

Figure 14 represents the TINA DPE as a layer between applications and network hardware. The DPE provides applications with an interface to various services. The functionality behind this interface is transparently implemented on one or more platforms. Each implementation then communicates with network hardware via a kernel transport network (e.g., TCP/IP). Key benefits of the DPE include design portability and interoperability. TINA applications will be supported by any compliant DPE platform, and applications running on different DPE platforms are able to communicate via a standard mechanism. Thus, the DPE is a powerful abstraction layer that benefits from the distributed processing and object orientation paradigms to resolve the problems of heterogeneity and distribution.

Another important distinction between TMN and TINA is the concept of a TINA *session.* Sessions are state containers; they hold information used by all entities involved in the provision of a service. The session concept has been specialized to support three key functions: access (establishment and set up of a session), service (actual provision of the service), and communications (provides facilitates to set up connections and manage networks). An example of these concepts is shown in Fig. 15, which demonstrates the interactions between a customer and service provider in which service is offered, access rights, specified service provided followed by billing. Being implemented at the service level means that commonly used connectionless protocols (such as SNMP) can be easily integrated into a TINA application.

F. Corporate Networks

For large corporations the approach to network management is pragmatic. Corporate network managers must attempt to provide a reliable network without a large human resource requirement. As a result, sophisticated network management systems are not a common feature. Instead, a simple approach is used with the rationale that simple designs have simple failures which are easy to fix.

Tools such as Netview (an IBM version of OpenView) and Ciscoworks (a tool for managing CISCO

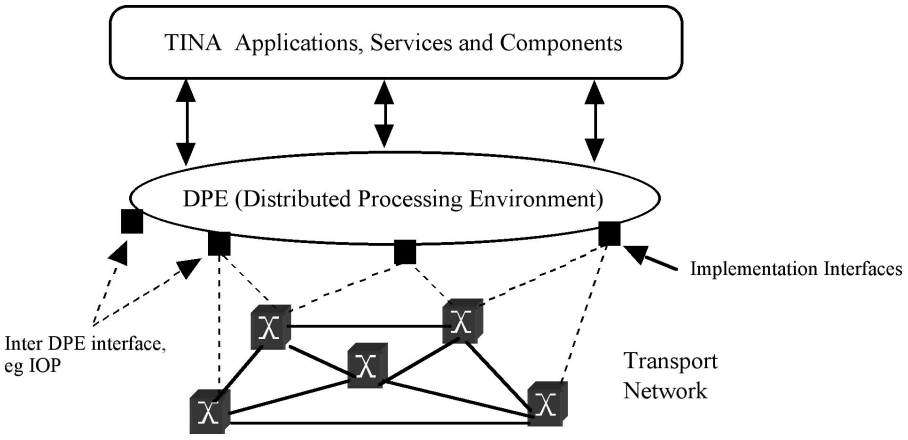


Figure 14 The TINA DPE.

routers) provide a basis upon which to build a network, but require substantial effort to customize. These tools often fail to operate correctly or simply do not scale well for the organization.

Instead, many corporations elect to use a selection of simpler management tools such as PING, traceroute and multi-router traffic grapher (MRTG) combined with some purpose-built software. MRTG uses SNMP to read MIB variables and then produces graphs representing the traffic load on network-links. MRTG generates hypertext markup language (HTML) pages containing graphic interchange format (GIF) images which provide a live visual representation of this traffic. For example, a study of one large corporate network revealed that management of their large national IP network was effectively achieved using basic tools such as PING, traceroute, and MRTG together with NetView and some in-house tools.

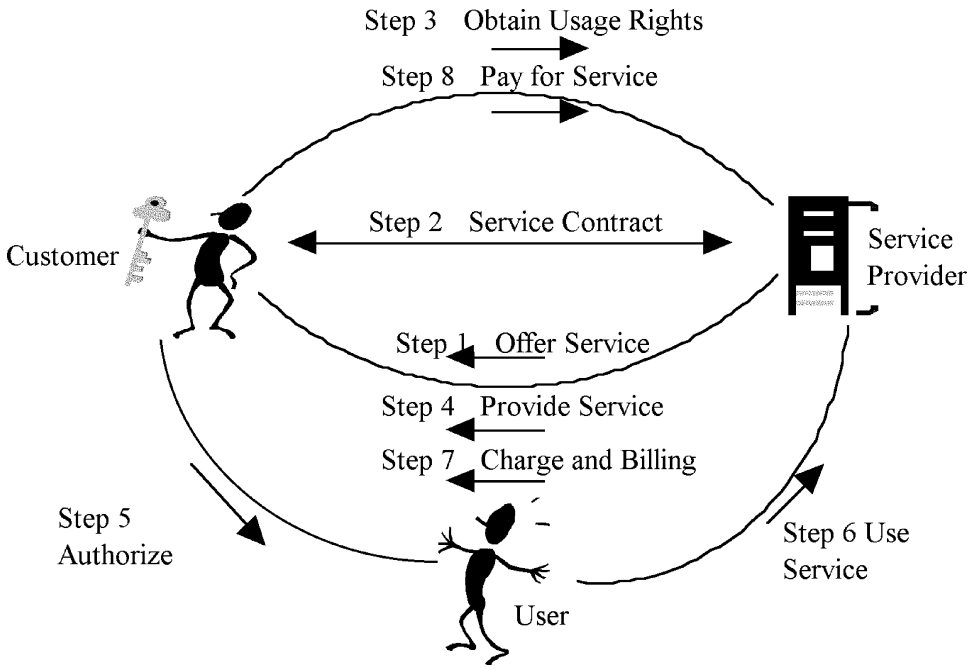


Figure 15 TINA session concept.

In recent years there has been a move away from proprietary networks (such as SNA [Systems Network Architecture] and DECNet [Digital Equipment Corporation Network]) toward IP-based networks. These networks are mature and well defined and as a result are less expensive to build, and many management tools are becoming available. With this trend towards IP-based networks, many corporations are concentrating on managing an IP-based network, relegating the lower level network details to a telecommunications carrier. Figure 16 illustrates this relationship, where the service provided by the telco is managed at the element layer of the corporation. Such a strategy leaves the telcos to manage the lower level networks while the corporation concentrates on management of higher level network services, and business solutions. In spite of this move toward IP-based networks, many management products still operate in a proprietary fashion (such as Netscape's Directory Server which only works with Netscape browsers). This limits the usefulness of many such management tools. Other management tools that operate in a more heterogeneous way are limited in their functionality (such as SMS operating on Windows NT). With no source code available, limited support, and no real way to tailor the product to specific needs, such tools become accessories to network management rather than a foundation.

In conclusion, corporate networks are complex and often require specialized management. Management systems purchased "off the shelf" cannot be efficiently adapted to meet the requirements of the corporation. Furthermore, resource constraints rule out producing their own management system. What emerges is a compromise between existing simple reliable tools and some in-house software to produce a simple, inexpensive, and reliable network management system.

V. ENABLING TECHNOLOGIES

A. Brief History

Management of TCP/IP networks was traditionally achieved using tools such as PING and traceroute. These tools are adequate for testing end-to-end connectivity and path response times, but they offer no assistance in determining the "health" of the network.

The development of real management solutions was driven by the difficulty organizations were experiencing managing their parts of the Internet. While CMIS

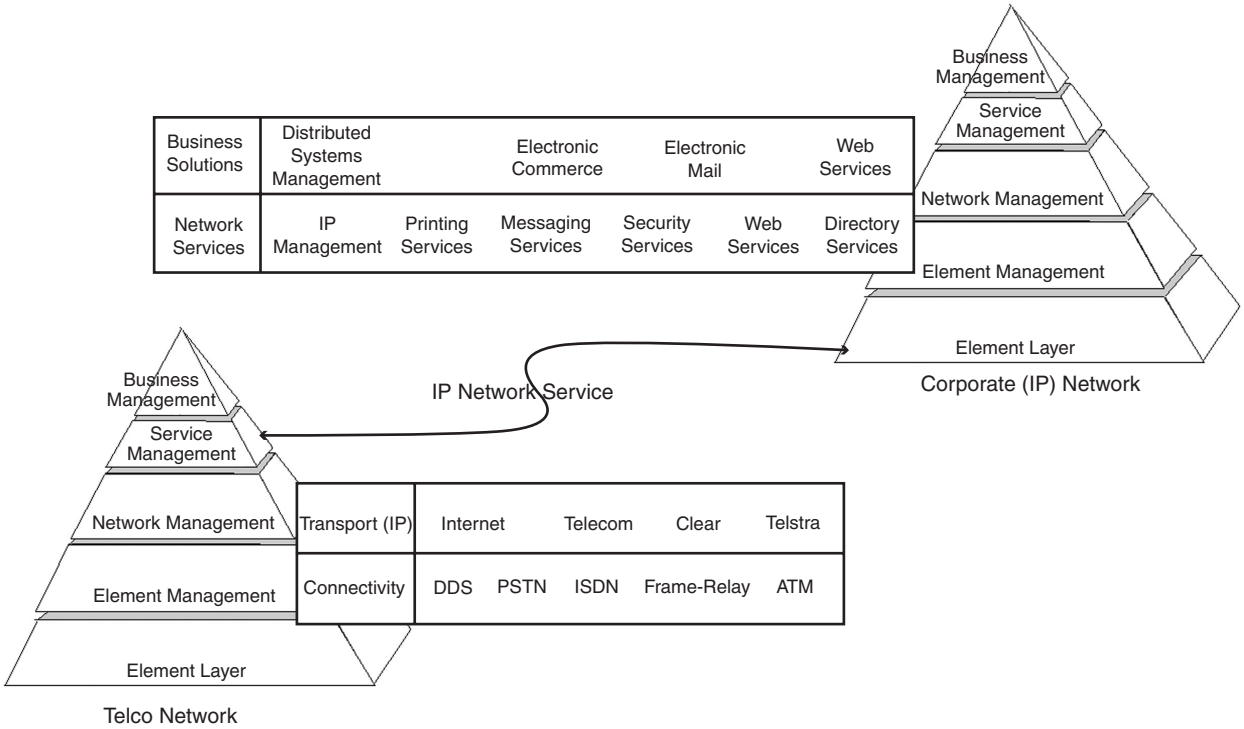


Figure 16 Network solutions architecture.

and CMIP were available at this time, the structure and definition of management information was poorly defined thus making implementation problematic. Several initiatives started in the late 1980s and included:

- High-level entity management system (HEMS)—a research project that was in use only at its development sites
- Simple gateway management protocol (SGMP)—a simple protocol which was easily implemented on a number of platforms and quickly became used in several networks outside its development environment
- CMIP over TCP/IP (CMOT)—an idea that was not immediately implemented yet remained popular as a future solution

Figure 17 shows the arrival and development of various protocols over time. At the first ad hoc network management review meeting in 1989, the lack of use of HEMS resulted in its withdrawal. CMOT was to remain on the drawing board as a possible future solution. SGMP was both implemented and widely deployed—as such it was decided that it be upgraded slightly to become the short-term network management solution and renamed to simple network management protocol (SNMP). In April 1989 the IAB upgraded its status to "recommended," thus becoming the only operational standard for TCP/IP network management. In late 1991 RMON was proposed, and it defined a set of managed objects and functions to facilitate remote network monitoring within the SNMP framework. Several efforts resulted in various implementations of SNMPv2 (1992) and SNMPv3 (1998).

B. Simple Network Management Protocol

Various versions of SNMP have evolved, but the most widely used is now SNMPv2c, or more commonly referred to as SNMPv2, and is specified in the following RFCs:

- RFC 1901—Introduction to Community-Based SNMPv2
- RFC 1905—Protocol Operations for SNMPv2
- RFC 1906—Transport Mappings for SNMPv2

SNMP specifies community string-based and party-based modes of operation, for example, SNMPv2c and SNMPv2p. Community string has the lowest security level and provides access to all attributes in a managed device under single password control. Party offers a higher level of security and permits access to defined subsets of attributes under password control.

SNMPv3 was proposed in 1998 and combines user-based security with protocol operations and data types derived from SNMPv2p. Good principles of software engineering were used to make the design modular, allowing subsequent improvements and backward compatibility to be easily implemented. This approach also provides support for a variety of security protocols. Full details of SNMPv3 are specified in the following RFCs:

- RFC 2571—An Architecture for Describing SNMP Management Frameworks
- RFC 2572—Message Processing and Dispatching
- RFC 2573—SNMPv3 Applications
- RFC 2574—User-Based Security Model for SNMPv3
- RFC 2575—View-Based Access Control Model for SNMPv3

Figure 18 shows a typical SNMP protocol configuration. Each managed object in the network must implement SNMP, UDP, and IP together with an agent process responsible for maintaining the object's MIB and responding to SNMP messages. Network management protocol stacks sit alongside protocol stacks for the various network devices—server, router, etc. in every device that has to be managed.

Although the simplicity mandated by the creators of SNMP resulted in easy implementation and widespread deployment, it also exposed some serious deficiencies. The most important of these are in the areas of functionality and security. SNMP supports only a centralized model of management. With large networks scalability becomes an issue. Retrieval of large volumes of data (such as tables) requires several SNMP messages, resulting in increased network management traffic which is clearly inefficient. Only trivial authen-

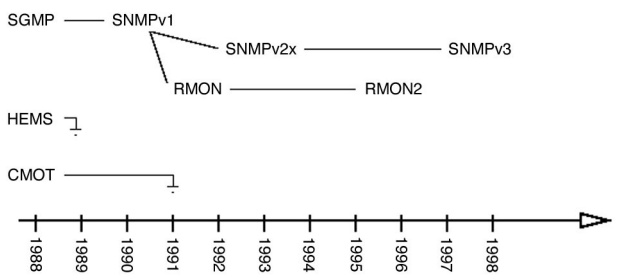


Figure 17 A timeline of management protocol developments.

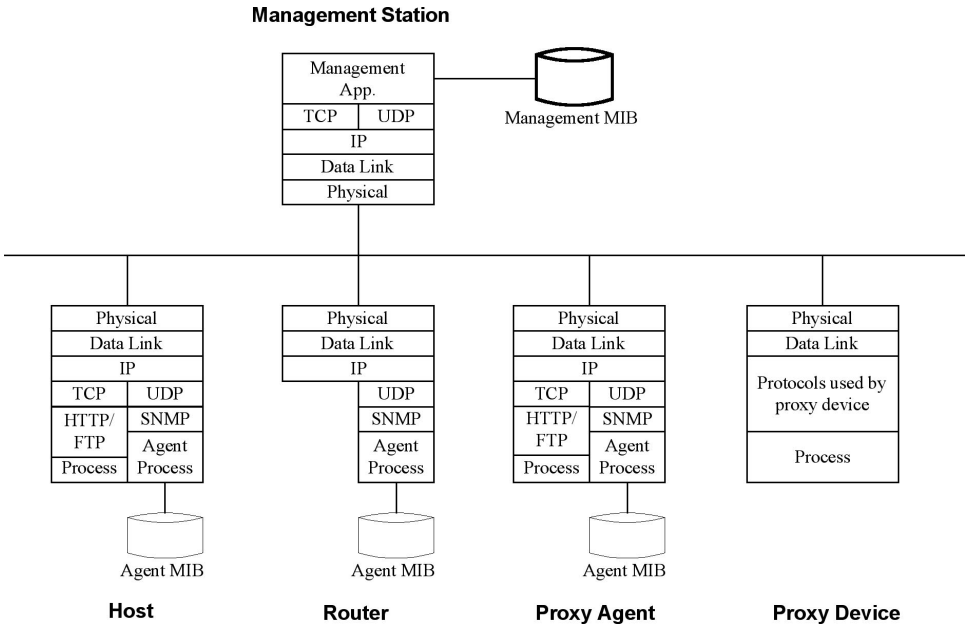


Figure 18 SNMP protocol architecture.

tication is required (passwords or "community strings" are sent in the clear), thus control applications are insecure. Subsequent enhancements to SNMP have addressed these problems.

C. Remote Monitoring Management Information Base

An important part of network management lies in monitoring parts of the network in order to provide summary information such as error or performance statistics. Traditionally, purpose-designed network monitors were used for this purpose. However, as computer memory became cheaper, the logic of the network monitor was incorporated into other network devices such as file servers. Some application level program was required to gather the statistics from each network monitoring station in order to build a central view of the network—SNMP does not have this functionality. To address this disparity the RMON (remote monitoring) MIB was developed, which is described in RFC 1757. This specification provided a repository for statistics in the form of an MIB accessible to SNMP. Previously, the central management station would gather low-level data from devices around the network and perform the analyses required to generate required statistics. As networks became larger, management stations had to become more powerful, resulting in scalability problems.

The sample network shown in Fig. 19 is initially configured without an RMON. The management station must poll each machine on the network at regular intervals to gather and then analyze this data. At points D, E, and C the traffic from each sub-network would be acceptable. When this traffic is directed through points B and A there is a great potential for bottlenecks. Thus, the central LAN must be able to deal with large volumes of management traffic which may require costly upgrades.

Furthermore, what happens when the WAN link at point G becomes congested or goes down? By

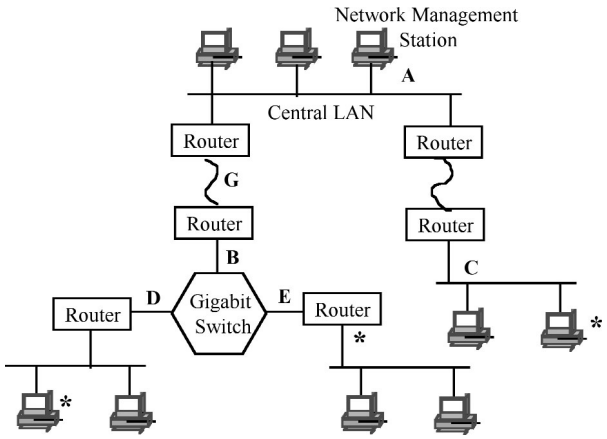


Figure 19 Applying RMON to a network.

introducing RMON facilities into the network at each of the asterisked points, potential bottlenecks can be alleviated. The RMON entity within each sub-network gathers raw data and generates the required statistics. The management station may then poll each RMON entity for the (less voluminous) statistics when required (preemptive monitoring). If a subnetwork is isolated from the central LAN for a period of time, the RMON entity may continue monitoring (off-line operation). RMON entities can be configured to report concurrently to each manager (multiple managers).

A shortcoming of RMON is its inability to operate above layer two as it was originally designed to provide remote monitoring services for physical networks and as a result the RMON MIB only contains object definitions for these layers.

RMON2 is a modification to RMON that includes monitoring of protocol traffic within layers three to seven. Hence traffic at specific network addresses (e.g., IP numbers) or even between specific hosts (e.g., Web servers) may be monitored. This allows the management system to determine the ultimate sources and destinations of traffic (rather than just that coming in and going out of the subnetwork).

D. Common Management Information Protocol

CMIP defines the procedures for the transmission of management information. Each CMIP protocol data unit (PDU) is a building block used in constructing the more complex CMIS; this correspondence is shown informally in Fig. 20. A full discussion of CMIP is beyond the scope of this article and for more detail the reader should refer to the standard ISO 9596/ITU X.711.

Any management system must try to meet three competing objectives:

- It should not seriously degrade the network.
- Decisions must be made and action taken quickly (to minimize instabilities).
- A wide range of services should be provided to allow detailed monitoring and control.

In the case of OSI/CMIP management systems, Objectives 2 and 3 above are met at the cost of bandwidth. Due to the range of available services and the bulk of the OSI MIB, resulting implementations tend to generate large volumes of network traffic.

This bulk and complexity have hindered its adoption by users and vendors alike. CMIP is a more complex and comprehensive management protocol than SNMP, and therefore various forms of it find favor in large telco networks rather than in LANs or more moderate-sized networks. However, with recent advances in networking technologies (such as Gigabit Ethernet) and the increasing demands upon network management systems, CMIP has been reborn under the umbrella of Q3 discussed in Sections IV.D.5 and V.E.

E. The Telecommunication Management Network Standard Interfaces Q3 and Qx

The Q3 and Qx interfaces are described in the standard ITU Q.812 and were introduced in Section IV.D.5

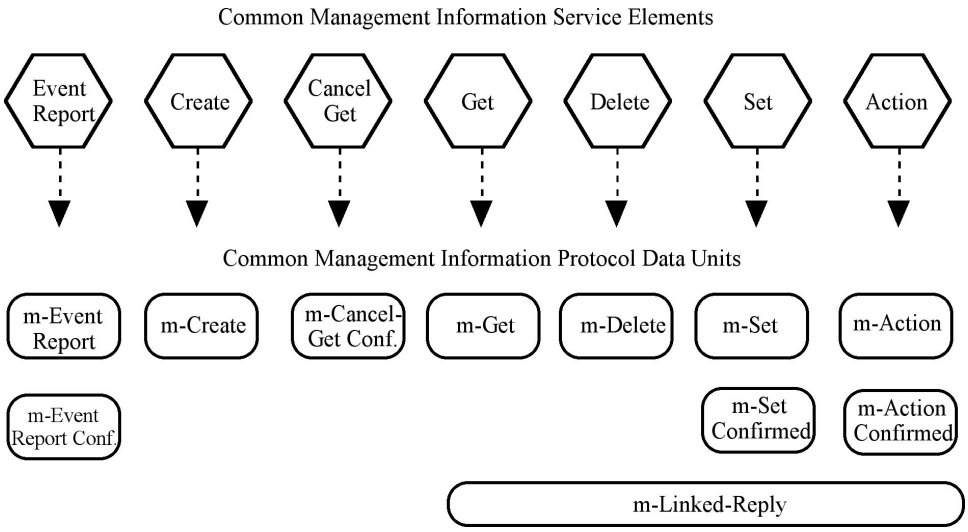


Figure 20 Correspondence between CMIS primitives and CMIP data units.

as a means of communication between the layers of the TMN hierarchy. The Q3 interface exists between OSFs, Q-adaptors (QAs), and network elements (the lowest layer in Fig. 21). A QA enables a TMN platform to manage network elements that do not have TMN-compliant interfaces—for example, an SNMP QA might translate between SNMP and CMIP. In the element management layer of Fig. 21 a performance monitoring function (an OSF) is communicating directly with a Q3-compliant switch (a network element). Additionally, the same OSF also communicates with a non-Q3-compliant switch via a QA. In this way international standard management protocols such as SNMP and CMIP and accompanying infrastructures can be mapped to certain proprietary interfaces, particularly at the network element layer.

The Qx interface carries non-TMN-conformant protocol information from network elements at the lowest layer. For example, the QA shown in Fig. 21 communicates with a non-Q3-compliant network element and then sends the information via Q3 to an OSF.

The F interface exists between a workstation and an OSF. For example, Fig. 21 also illustrates a workstation requesting information from the OSF at the business management layer. Currently, the F interface is vendor specific, although it is expected that it will become a more vendor-independent interface standard in the future.

The X interface provides a means of communication between two different management systems. Figure 21 shows company A (a parent company) requesting business information from the OSF at the business management layer. Also shown is the dependence of company B (a service consumer) upon company X. Exchange on management information between IBM's Tivoli and HP's Openview is an example of the X interface implementation. The X interface could be implemented via some form of proxy agent until its implementation becomes more widely accepted.

Q3 is described as the "third level" interface between OSs and network elements. Originally, three interfaces were defined in the TMN physical model: Q1, Q2, and Q3. The numbers correspond to an increasing level of intelligence or complexity of each interface. The requirements for Q1 and Q2 were never clearly defined, so it was decided that they be merged

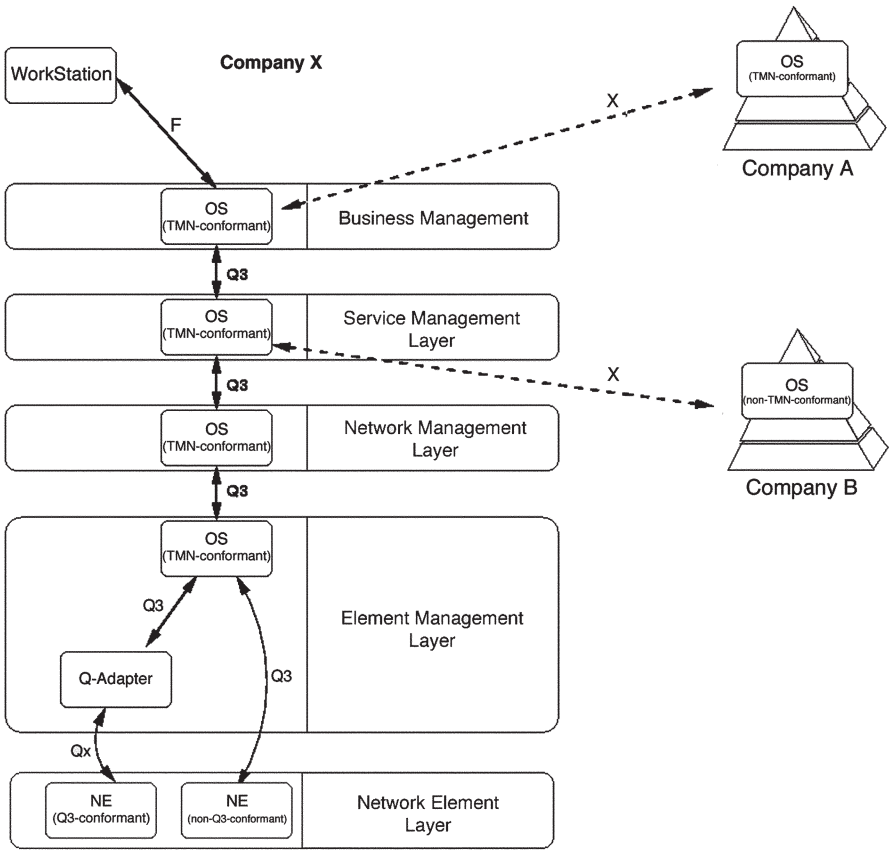


Figure 21 Interfaces between TMN functional components.

into a single interface called Qx. Q3 is essentially CMIP, whereas Qx encompasses several protocols (such as SNMP, SS7, or even an ASCII message-based protocol).

Several large corporations and telecommunications providers are embracing TMNs in an effort to streamline their network management. This surge of interest has resulted in the implementation of CMIP—more than a decade after it was designed.

F. Recent Developments

1. Web-Based Enterprise Management

Web-based enterprise management (WBEM) specifies a group of technologies which provide Web access to both managed data and managed elements. Technologies which are likely to be utilized for WBEM are the common information model (CIM), the extensible markup language (XML), and the hypertext markup language (HTTP). Figure 22 illustrates how these technologies might be integrated.

CIM has evolved from the proposed Hyper Media Management Schema (HMMS) and may yet grow to incorporate directory enabled networking (DEN) which is a proposed repository of policy-based management information used to integrate disparate parts of the network. CIM is a nonproprietary system used to describe overall management information. Implementation neutrality provides a common understanding of management data across different management systems and thus simplifies the integration of management information from varied sources into a common repository. The CIM system is used to represent real-world objects being managed. CIM uses an object-oriented paradigm, where manageable objects are modeled using the concepts of classes and instances.

XML provides a means of encoding the CIM metaschema for distribution between CIM-enabled applications, while HTTP provides the means to retrieve management information from management servers as shown in Fig. 22. XML provides an open industry standard for describing structured data over the Web with built-in validation through the use of document type definitions (DTDs). The use of HTTP provides the benefits of application neutrality and simplicity. These factors should assist in the development of WBEM implementations.

2. Distributed Object Management Architecture (CORBA and DCOM)

The rapid growth of the Web combined with developments in high-speed and multimedia networks have brought distributed processing into the fore. To simplify the management of software-based components, two similar distributed object models have emerged—CORBA (common object request broker architecture) and DCOM (distributed component object model) which is Microsoft's version of CORBA.

DCOM is the distributed extension to the component object model (COM) that builds an object-based remote procedure call layer on top of distributed computing environment remote procedure call (DCERPC) to support the management of remote objects. DCE is a client/server software technology used for setting up and managing computing and data exchange in a system of distributed computers. DCE is normally used in geographically disparate networks comprising heterogeneous systems (e.g., servers). Us-

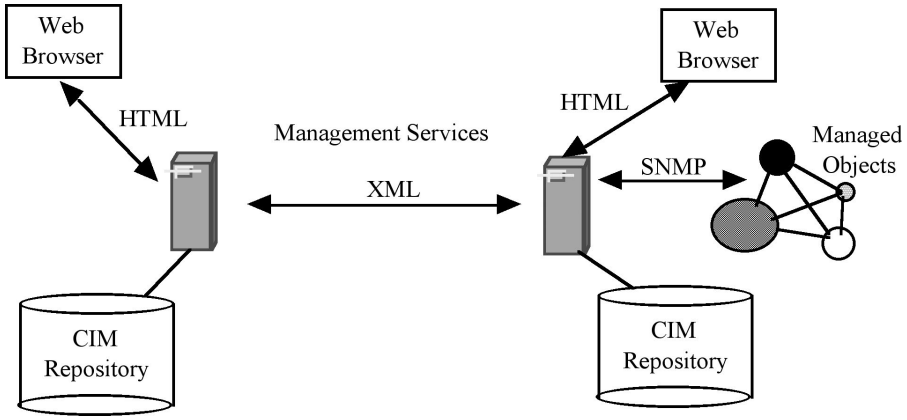


Figure 22 Interaction of WBEM enabling technologies.

ing DCE, application users can use applications and data at remote servers. Programs can be run from and data can be located at a variety of locations.

A COM server can create object instances of multiple object classes. A COM object can support multiple interfaces (set of functionally related methods), each representing a different view or behavior of the managed object. For example, a network router might have two interfaces, each representing a different version of the driver software. Alternatively, the implementation of IP security (IPSec) for running an encryption tunnel in a virtual private network (VPN) will result in a logical IP and mask interface—in addition to the conventional IP and mask numbers. A COM client interacts with a COM object by acquiring an interface pointer from the object through which methods may be invoked. In these examples, therefore, a network management system can operate independently of the particular driver software version or IP/mask specifications.

CORBA is a similar distributed object framework proposed by a consortium of over 700 companies called the object management group (OMG). The core of the CORBA architecture is the object request broker (ORB) that acts as the object bus over which objects transparently interact with other objects located locally or remotely. A CORBA object is represented to the outside world by an interface (e.g., the router interfaces given in the example above) with a set of methods, and an instance of an object is identified by an object reference.

Both DCOM and CORBA are based upon the client/server paradigm. To request a service, a client invokes a method implemented by a remote object, which acts as the server in the client/server model. The service provided by the server is encapsulated as an object, and the interface of an object is described in an interface definition language (IDL).

In both DCOM and CORBA, the interactions between a client process and an object server are implemented as object-oriented remote procedure call (RPC)-style communications. Figure 23 shows a typical RPC structure for both DCOM and CORBA. The application (e.g., network management program for the routers) need only know about the DCOM or CORBA client interface and nothing of the architecture which underlies it. To invoke a remote function, the client makes a call to the client "stub" which encapsulates the requested parameters and passes the session unit to the underlying transport network (e.g., TCP/IP) for relay to the server.

At the server side, the transport network delivers the message to the server stub, which then unpacks the request message and calls the actual function or

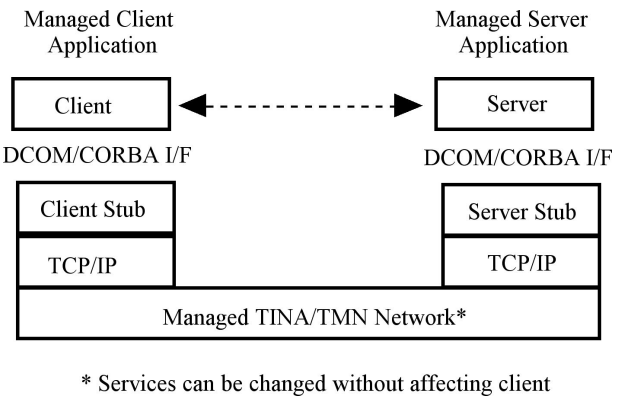


Figure 23 Separation of management functions using DCOM or CORBA interfaces.

the object. In this manner an underlying managed transport network—commonly provided by TINA or TMN-based telco network providers—can continue to provide transparently managed services to its client in spite of changes to management architecture, providing the appropriate DCOM or CORBA-based applications programming interfaces (APIs) are supported as can be seen in Fig. 23.

3. Policy-Based Networking

Policy-based networking (PBN) has been around for some time as an idea. The objective of PBN is to take a set of policies governing the availability of computing resources to various users. The system stores these policies in a repository for future reference and, where necessary, then distributes the policies to other relevant network entities. Each of these entities then implements the parts of the policy that affect them.

Several initiatives currently under development are working toward the goal of PBN: directory enabled networking (DEN), lightweight directory access protocol (LDAP), and resource reservation protocol (RSVP). Figure 24 illustrates relationships between these developing technologies.

DEN is the proposed repository of policy information for PBN. DEN provides the ability to relate users profiles to other user information such as an IP address. The key to the success of DEN will be in the standardization of the object definitions that comprise the repository, which must be both comprehensive and extensible. DEN could be likened to the SNMP SMI (see Section III.A.1).

LDAP is the proposed means by which the standardized DEN object definitions will be propagated throughout the network. Unlike DEN, LDAP has been

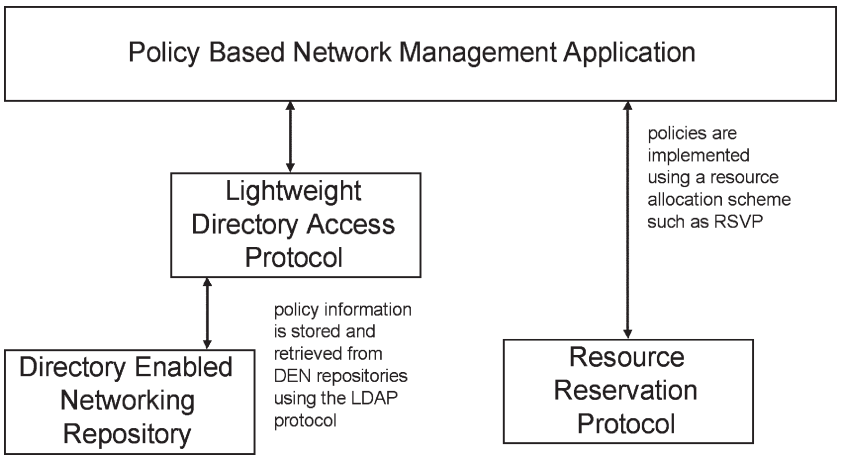


Figure 24 Relationship of PBN enabling technologies.

through the RFC process and implementations exist—the latest of which is LDAP3.0. LDAP is to DEN as SNMP is to MIBs—a platform-independent communications protocol for accessing a repository of management information.

RSVP is one of several possible resource allocation schemes that might be used to implement network policies. For more information about this and other allocation schemes, see RFC 2205.

DEN is still awaiting ratification by the desktop management task force (DMTF) before being incorporated into the CIM. No Internet resource allocation schemes (including RSVP) are yet widely implemented. While vendor-specific solutions currently implement sections of PBN functionality for their products, the goal of PBN cannot be realized until all of the component technologies become standardized.

Microsoft's *Windows Distributed interNet Applications* (DNA) architecture utilizes these technologies. DNA is a three-tier model composed of business, data, and presentation tiers. The business tier concentrates on providing business services using Microsoft specific technologies. The data tier uses active directory service interfaces (ADSI) as an API to the active directory (AD) schema; AD is a Microsoft implementation of DEN mixed with CIM. The presentation tier uses XML and HTTP (see Section V.F.1) to encode and transport the data in a vendor neutral fashion.

VI. CONCLUSIONS

This article has addressed a number of the functional and organizational aspects of modern network management. It has discussed the diverse requirements for the management of different network architectures. Furthermore, it has examined the key technologies such as SNMPv3, RMON2, CMIP, Q3, TMN, TINA, and distributed object Web-based management.

In particular, the article has discussed the evolution of SNMP up to SNMPv3 and associated RMON development. The rebirth of CMIP via the Q3, Qx, and evolving F and X interface specifications has given substantial impetus to large telco-based networking infrastructure. As the cohesion provided by TMN and TINA finds favor, CMIP may become more widely implemented. Lightweight CMIP implementations for managed devices are likely to be the subject of future research. It is still an open question as to whether the simplicity and versatility of (the now secure) SNMPv3 will be more desirable than the heavyweight power of CMIP.

Web-based management systems are now starting to be based upon object models such as CORBA and DCOM in conjunction with hyper media and extensible mark up language tools. These systems have benefits for both broadband and multimedia networks, where distributed processing and time-critical application are commonplace. Other recent and ongoing developments include PBN using directory systems such as RSVP and LDAP.

SEE ALSO THE FOLLOWING ARTICLES

Database Administration • Extranets • Firewalls • Integrated Services Digital Network • Transmission Control Protocol/In-

ternet Protocol • Wide Area Networks • XML (Extensible Mark-up Language)

BIBLIOGRAPHY

Desk Top Management Task Force (DMTF). (2001). www.dmtf.org/education/ (Presentations, White Papers, Tutorials).

ISO 10040 Information technology—Open Systems Interconnection. Systems management overview, 1998.

ISO 8824/8825 Information technology—Open Systems Interconnection. Specification of Abstract Syntax Notation One (ASN.1), 1998.

ISO 9596-1 Information technology—Open Systems Interconnection. Common management information protocol—Part 1: Specification, 1998.

ISO standards are available from www.iso.ch/iso/en/ISOOnline.frontpage.

McCloghrie, K., and Rose, M. (May 1990). Structure and Identification of Management Information for TCP/IP Based Internets, RFC 1155.

Network Management Forum. (2001). www.nmf.org.

Simple Times, The. (2001). www.simple-times.org.

Stallings, W. (1999) SNMP, SNMPv2, SNMPv3, and RMON 1 and 2, 3rd edition. (1996). Reading, MA: Addison-Wesley.

Willets, K., and Adams, E. (1996). *The Lean Communications Provider.* New York: McGraw-Hill.

Neural Networks

Melody Y. Kiang
California State University, Long Beach

GLOSSARY

epoch A complete iteration of the training cycle that allows all examples to be fed to the network once.
input function A function used by the neural networks to aggregate inputs from different sources.
learning algorithm A systematic method used by the neural network to adjust its connection weights during the training process.
neuron The equivalent of a processing unit in the artificial neural network.
supervised learning A type of learning method that requires both input and corresponding output to be known.
transfer function A function used to convert calculated input to output (also called output function).
unsupervised learning A type of learning method that does not require the expected output for a given input to be known.

NEURAL NETWORKS, also known as connectionist models, parallel distributed processing models, neurocomputers, and artificial neural networks, are a network of many simple highly interconnected processing units (also called neurons or nodes) operating in parallel. The processing units respond to external inputs through dynamically changing connection strengths between the nodes. The processing result is not stored or output to a specific memory location. It is represented by the overall state of the network after it has reached some equilibrium condition. Knowledge within a neural network is acquired through a learning process. Knowledge is stored both in the way the processing units are connected and the strengths of interneuron connections.

I. HISTORY OF NEURAL NETWORK DEVELOPMENT

For many centuries researchers have been trying to build intelligent computer systems that can perform the daily routines of mankind for the purpose of relieving humans from tedious tasks or hazardous work environments. Several problems have been encountered in conventional artificial intelligence research. For example, human memory can store a huge amount of information and find relevant items very quickly, far exceeding what a machine can do. To make intelligent computer systems requires an understanding of how humans process information. The inspiration for neural network architecture originally came from studies of the biological nervous systems. People have long recognized that human beings and animals are much better and faster at processing and recognizing speech and images than most sophisticated computers. The field of neural networks began as an approach to imitating human intelligence for building artificial intelligence systems that can learn from experience.

The study of neural networks is highly interdisciplinary and has attracted the attention of people from areas such as mathematics, computer sciences, psychology, and statistics. As with other areas of artificial

intelligence, there are two schools of research. Cognitive scientists are interested in building neural networks to mimic brain behavior in order to further an understanding of cognition. Computer scientists are more focused on the mathematical properties and problem solving performance of neural networks. The following discussion relates to the latter definition.

A. The Origin of Neural Networks

Neural networks began with the pioneering work of McCulloch and Pitts in 1943. They outlined the first formal model of an elementary neural network and demonstrated that it can be used to represent logical relations such as "AND" or "OR" relations. Later they realized that such a model could be used to represent the brain's process of pattern recognition and classification. In 1949, Donald Hebb first proposed a learning scheme for updating neuron's connections known as the Hebbian learning rule. Minsky built and tested the first neurocomputers in 1954. In 1958, Frank Rosenblatt developed the first neural network architecture, called perceptrons, which allow dynamic modification of the strengths of the interneuron connections. The neural network machine is capable of learning to classify certain patterns.

B. Problems of Two-Layer Networks

The subject of neural networks, once viewed as the theoretical foundation for building artificial intelligent systems in the 1950s and 1960s, was proven to be too limited by Minsky and Papert. In their book, *Perceptrons,* published in 1969, they demonstrated that the simple two-layer perceptron is incapable of usefully representing or approximating functions outside a very narrow and special problem domain. In the case of the well-known exclusive-or (XOR) function, they showed that the function cannot be learned by a two-layer network. It took nearly 20 years for researchers to resume an interest in neural networks after breakthroughs that overcame the limitations cited earlier and after new hardware development increased the processing capabilities.

C. Multilayer Networks

Several neural network researchers began to explore the ability of multilayer feedforward networks to approximate general mappings from one finite dimensional space to another. For example, Rumelhart, Hinton, and Williams developed a *backpropagation* learning algorithm to train a multilayer network that can reproduce the XOR function. The publication of two volumes on parallel distributed processing in 1986, edited by McClelland and Rumelhart, introduced new learning rules and network topologies which opened a new era for the development of neural networks. Since then, neural network research has virtually exploded with impressive successes across a wide range of applications in various disciplines.

D. Fault-Tolerance and Distributed Representation

The renewed interest in neural networks has manifested in the study of a new class of computation models called the connectionist models which have limited analogy, if any, to their neurophysiology origin. Connectionist systems provide massive parallel processing capabilities that are essential in many domains, such as pattern recognition, concept classification, speech processing, and real-time process control. Fault-tolerance is another appealing property that has profound implications in the design and fabrication of integrated circuits. Neural networks use distributed representation to store knowledge which means that a concept is represented not by a single neuron, but by a pattern of activation over a large number of neurons. The advantage of distributed representation is that when a small random subset of the network is altered it does not change the macroscopic behavior of the network. However, the disadvantage is that it is hard to interpret or modify the connection strength of the network by an outside observer. Existing computers are serial machines based on the Von Neumann architecture proposed some 50 years ago. These machines are designed to execute serial threads of instructions and are vulnerable even to minute component failures. Due to its distributed representation nature, a connectionist system can tolerate minor component failures without impairing the entire system.

II. PRINCIPLE COMPONENTS OF NEURAL NETWORKS

A neural network consists of a number of interconnected homogeneous processing units. Processing units, the neural network equivalent of neurons, are generally simple computation devices. Their behavior can be modeled by simple mathematical functions. A

unit i receives input signals from other units plus a bias or offset term θ_i, aggregates these signals based on an input function I_i, and generates the activation state a_i. The activation state, a_i, is then transferred to an output value $Output_i$, often times a binary output signal of 1 or 0, corresponding to the action of fire or do not, based on an output function O_i (sometimes called a transfer function) (see Fig. 1). No assumption is imposed on the form of input/output functions at each unit other than to be continuous and differentiable. The output signal is then routed to one or another processing units and possibly back to itself via the interconnection as input signal. A processing unit may receive one or many inputs from other units, but produces a single output value each time. The routing of the signals is directed by the topology of the network.

A. Connection Weights and Directions

The configuration of a neural network is represented by a weighted directed graph (WDG) with nodes representing units and links representing connections. Net topology determines how the processing units are connected to each other. Processing units are connected via a network of links carrying the output of one processing unit as input to another processing unit. Those links (connections) could be unidirectional (one-way connection) or bidirectional (two-way connection). There are weights associated with the links that represent the connection strengths between two processing units. For example, if processing unit j is connected to unit i such that the output of unit j is the input to unit i, then the strength of connection between node j and i is stored in weight w_{ij} as shown in Fig. 2. For a two-way connection, there is another link (w_{ji}) that sends output from node i as an input to node j. Usually, the values of the two weights are not the same. Processing units usually operate in parallel and are configured in certain patterns. There are two kinks of connection possible between two nodes: excitatory and inhibitory. A link with a positive weight value represents an excitatory connection while a negative weight value means an inhibitory connection. In the excitatory connection, the output from node i increases the potential of the nodes receiving input from node i to fire. On the other hand, in the inhibitory connection, the output from node i reduces the probability of the receiving nodes to fire.

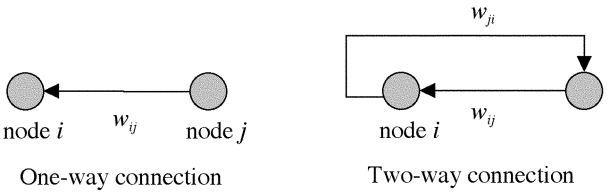


Figure 2 One-way and two-way connections between two processing units.

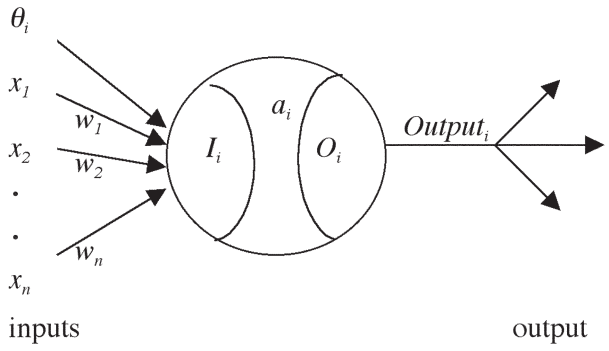


Figure 1 Principle components of a processing unit i.

B. Input Functions

A node receives input from one or more other nodes via the connections among them. The inputs may be either external inputs, such as the input portion of a training example, or internal inputs from other processing units. Each link between the input and the node has a weight associated with it. We assume there are n input signals $x_1, x_2, \ldots, x_n$ and weights $w_1, w_2, \ldots, w_n$. The input function is the function we use to aggregate the input signals to produce the activation state (a) of the node.

1. Dot Product Input Function

Different input functions have been proposed and the most popular input function is the one that simply takes the weighted sum of all inputs plus a bias or offset term θ. The bias input value θ is normally set to 1.0 initially and can be learned during the training process,

$$a = w_1x_1 + w_2x_2 + \ldots + w_nx_n + \theta.$$

This may be represented more compactly as

$$a = \sum_{i=1}^{n} w_ix_i + \theta.$$

This is called the dot product input function. Intuitively, the dot product scales each input according to

its relative influence on the activation state of the node. Therefore, we can consider the absolute magnitude of the weights corresponding to the relative importance of the inputs. In some network implementations, the absolute values of the weights are limited to the interval 0 to 1. This normalization prevents the input to a node from exploding to large values hence dominating the learning process. This is especially important for certain types of networks such as Kohonen's Self-Organizing Map networks. Some implementations also require numeric inputs to be normalized in the range 0 and 1 before the start of the training process. This is done by an input preprocessing function.

2. Input Preprocessor

The learning rule used in the training process usually uses each input in the calculation of the gradient with respect to the weights. As a result, if the absolute value of an input is larger, the weight adjustments associated with that input would also be large. In some cases, the adjustments may become too large and cause the processing element to saturate its output and make learning extremely slow or not at all. If some inputs have very small values, their information content may be lost or not effectively used by the network. The solution to this problem is to preprocess all inputs so that they have the same value range. This enhances the fairness of training by preventing an input with large magnitudes from having excess influence on the training process.

a. Mean Standard Deviation Preprocessing Function

There are different preprocessing functions proposed to normalize inputs. A preprocessing procedure occurs before the start of the training process and is associated only with the input layer of the network. A mean standard deviation method is the most popular preprocessing function implemented in neural network applications. Each input is calculated separately. The input is modified by subtracting the mean for that input and then dividing by the standard deviation for that input:

$$x_i^{\text{new}} = (x_i^{\text{old}} - \text{mean}_i)/\text{StdDev}_i \qquad \text{for each } x_i.$$

x_i^{new} is then used as the input to the network during the training process.

b. Max Min Preprocessing Function

Another popular input preprocessing function, the Max Min method, modifies the input by subtracting the average of the maximum and minimum values of that input and then dividing by the difference of the two values. The application of the preprocessing function can usually improve the performance of the network without loss of information.

c. Categorical Variables

Neural networks also accept categorical input values. Categorical variables may be two-state (i.e., *Employed* = {*No, Yes*}), or multiple-state such as *Marital_Status* = {*Single, Married, Divorced, Widowed*}. Unlike two-state categorical variables which can be easily represented by a 0 or 1 value, multiple-state variables are more difficult to handle. One approach is to assign a numeric value to represent a particular state (i.e., *Single*=1, *Married*=2, *Divorced*=3, *Widowed*=4). However, when training the network, this implies a certain ordering of the four states which may not be true (e.g., *Single* < *Married* < *Divorced* < *Widowed*).

d. One-of-*N* Coding

A better way is to use a number of numeric variables to represent each categorical variable. This is known as a *one-of-N* coding scheme. The number of variables used should equal the number of possible states of that variable. Statisticians call these dummy variables. Each dummy variable is given the value 0 except for the one corresponding to the correct category, which is given the value 1 (e.g., *Single*={1,0,0,0}, *Married*= 0,1,0,0}, *Divorced*={0,0,1,0}, *Widowed*={0,0,0,1}). One problem with the one-of-*N* approach is when the number of possible states is large, it requires a large number of dummy variables to represent one input value and may end up generating a network too complicated to be effectively trained in a reasonable time.

3. Distance Input Function

Besides the dot product input function, other popular input functions implemented in neural network applications include a distance input function that sums the absolute difference of the input and weight vectors:

$$a = |x_1 - w_1| + |x_2 - w_2| + \ldots + |x_n - w_n|.$$

This input function is useful when dealing with inputs and weights constrained to take on values in the interval [0, 1].

4. Squared Euclidean Distance Input Function

Another distance input function calculates the squared Euclidean distance between the input and weight vectors:

$$a = (x_1 - w_1)^2 + (x_2 - w_2)^2 + \ldots + (x_n - w_n)^2.$$

This function will always return a nonnegative value and is minimized ($a = 0$) when the two vectors are the same ($x = w$).

5. Radial Basis Input Function

Another input function, the radial basis function (RBF), allows a node to form elliptic regions of activity or inactivity. The function is defined as

$$a = (x_1 - w_1)^2 w_2{}^2 + (x_2 - w_3)^2 w_4{}^2 + \ldots + (x_n - w_{2n-1})^2 w_{2n}{}^2.$$

For each input to the processing unit, there are two weight values associated with it, hence it requires more time to process.

C. Output Functions

The output function, also known as the transfer function, acts on the value returned by the input function that is also known as the activation state *(a)* of the node. While the activation state of a node may be a large value, applying a transfer function can limit the output to a finite interval. There are many different output functions implemented in neural network applications, being linear or nonlinear, and each is designed for a particular type of problem. It is the nonlinearity of the output function that gives a neural network its advantage over traditional regression techniques.

1. Sigmoid Function

The most popular transfer function among neural network applications is the sigmoid (S-shaped) function that produces a continuous value in the 0 to 1 range (see Fig. 3). It has the form

$$\text{Output}_i = 1 \,/\, 1 + e^{\lambda * - a_i},$$

where $\lambda > 0$ and is proportional to the neuron gain determining the steepness of the function near $a_i = 0$ as shown in Fig. 4. The appropriate value of λ is determined by the model builder. The value of Output_i will always be in the [0, 1] range. The sigmoid output function is used in many popular networks including the *backpropagation* and *Hopfield* networks.

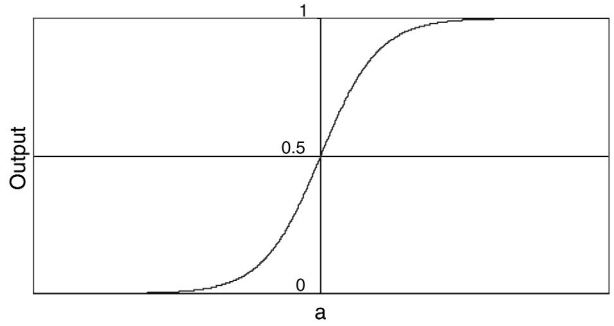


Figure 3 The sigmoid function.

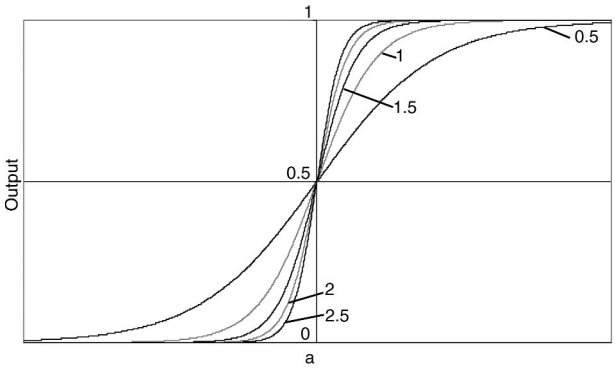


Figure 4 The effect of different λ values on the output functional form.

2. Bipolar Sigmoid Function

In some applications, a bipolar sigmoid transfer function is used, yielding output values in the range [−1, +1]. The function has the form

$$\text{Out}_i = 1 - e^{\lambda * - a_i} / 1 + e^{\lambda * - a_i}.$$

When $a_i = 0$, the value of Out_i is also zero. Depending on the nature of the application, the value of Out_i can be further transferred into the value of 1, 0, or −1 according to the following rules:

$$\text{Output}_i = \begin{cases} 1, & \text{if } \text{Out}_i > 0 \\ 0, & \text{if } \text{Out}_i = 0 \\ -1, & \text{if } \text{Out}_i < 0. \end{cases}$$

It is suggested that the bipolar sigmoid is preferable to the sigmoid on classification tasks because of its [−1, 1] range value. We can map −1 to represent nonmembership in the class, and 1 to represent membership. In some cases, the output of a node can be a stochastic function of a_i and can depend in a probabilistic fashion on its activation value; that is, the calculated value of a_i determines the probability p for a node to fire ($\text{Output}_i = 1$).

3. Linear Output Function

The simplest output function is the linear output function that just passes on the value returned by the input function as

$$\text{Output}_i = \lambda * a_i.$$

Again, λ is a parameter and sometimes is set to equal 1. This function has been used in *brain-state-in-a-box* neural networks.

4. Threshold Linear Output Function

Another version of the linear transfer function incorporates a threshold value T to determine the final output value. The function is defined as

$$\text{Output}_i = \begin{cases} 0, & \text{if } a_i \leq T \\ a_i - T, & \text{if } a_i > T. \end{cases}$$

This function allows the node to fire only after it has reached the threshold value T, and the output value is the value of the activation value minus the threshold.

5. Step Function

A similar output function called the step function also uses a threshold value T to determine the output value. The difference is there are only two possible output values 0 or 1. The function returns 0 when the activation value of the node is lower than or equal to the threshold value, and returns 1 otherwise. The function is defined as

$$\text{Output}_i = \begin{cases} 0, & \text{if } a_i \leq T \\ 1, & \text{if } a_i > T. \end{cases}$$

6. Signum Function

When $T = 0$, it is called a signum function. Another form of a signum output function is

$$\text{Output}_i = \begin{cases} -1, & \text{if } a_i \leq 0 \\ 0, & \text{if } a_i = 0 \\ 1, & \text{if } a_i > 0. \end{cases}$$

Both *perceptron* and *bidirectional associative memory* networks implement signum output functions.

7. Winner-Take-All Function

A special output function used by some Kohonen networks is called the winner-take-all function. This output function acts on the entire layer of nodes at once, finding the winner node that is closest to the input in certain measurements and sets its output to 1. All other nodes within the same layer have output 0. The most common way of measuring the distance between a node in the Kohonen layer and an input pattern is the Euclidean distance measure.

III. TYPES OF NEURAL NETWORKS

Neural networks can be categorized by the learning algorithm and network topology they apply and the type of data they accept.

A. Supervised vs Unsupervised Learning

The two primary categories of learning algorithms are supervised and unsupervised. In supervised learning, the network acquires knowledge by comparing the generated output with the known correct output. A set of training cases consisting of input data and corresponding output has to be available. The connection strengths are adjusted to tune the network over a number of training cycles. An example of supervised learning is a network being trained to recognize handwritten characters. On the other hand, in unsupervised learning, the network does not require the knowledge of corresponding output for comparison and learning. Input data presented to the network consist of the complex characteristics of the entities of interest. Through repeated training cycles the learning algorithm adjusts the connection strengths and causes the network to represent a simplified feature map of the characteristics from which meaningful information can be identified. An example of unsupervised learning is a network being trained to group customers into different market segments based on their purchasing behaviors.

B. Feedforward vs Feedback Networks

There are two types of network topologies, feedforward and feedback. In a feedforward network, the connections between processing units do not form cycles. Hence feedforward networks usually respond to an input quickly and require less time to train. In a feedback or recurrent network, there are cycles in the connections. Each time an input is presented, the neural network must iterate for a significant amount of cycles before it produces a response. Therefore, feedback networks are usually more difficult to train than feedforward networks. Some kinds of learning algorithms can be implemented as either feedforward or feedback networks. For example, LVQ and cascade correction networks are feedforward-only networks while the Boltzmann Machine is a feedback network. The backpropagation learning algorithm has been implemented in both feedforward and feedback networks.

C. Categorical vs Quantitative Input

Types of neural networks can further be distinguished by the kinds of data they accept, categorical or quantitative. Categorical variables take only a finite number of possible values. Symbolic values such as "green" and "red" can be encoded using *one-of-N* coding into numbers before being given to this type of network. Quantitative variables can store numerical measurements of some attribute, such as weight and height of a person. The selection of the network architecture depends on the type of problem under analysis.

IV. NEURAL NETWORK ARCHITECTURES

In a neural network system, there are many interconnected nodes. Normally nodes are grouped into layers. A neural network can have three types of layers—input layer, hidden layer(s), and output layer. The nodes within the same layer may communicate with each other, or they may not have any connection. Normally, in many of the popular network designs, the nodes on a layer do not have interconnections. The nodes of one layer always are connected to the nodes of at least another layer. There are different types of connections between layers: fully connected, partially connected, feedforward, feedback, hierarchical, and resonance.

A. Fully vs Partially Connected Networks

In the fully connected network, each node on the first layer is connected to every node on the second layer while in the partially connected layers, not all nodes on the first layer have to be connected to every node in the second layer. For partially connected layers, there are many possible ways to connect the nodes between two layers. Figure 5 shows examples of a fully connected layer and one possible partially connected layer based on the same topology.

In the feedforward-only (or feedforward, for short) connection, the nodes on the first layer send their output to the nodes of the second layer, but do not allow any input back from the nodes on the second layer. On the other hand, the feedback network allows output from the second layer as input to the first layer. Both feedforward and feedback networks can be fully or partially connected. Figure 6 shows examples of feedforward and feedback connections.

B. Hierarchical Network

For a network of two-layer design, there is one input layer and one output layer. The nodes in the input layer do nothing but pass external inputs to the nodes connected to them. The number of nodes in the input layer corresponds to the number of input variables. For a network with more than two layers, the additional layers are called hidden layers because their activation is not directly visible from outside the network. Hierarchical connection applies only to networks with one or more hidden layers. The layers are placed in a linear order, so that the input to the first layer (input layer) is the external input, the input to the second layer is the output of the first layer, and so on. The last layer in the sequence is the output layer. The nodes of one layer communicate only with the nodes on the next level. If a design is not hierarchical, the nodes of one layer may communicate with nodes on all other layers. The hierarchical network is a special case of a feedforward network. Figure 7 shows an example of a hierarchical connection. In the example, there are three input nodes, two hidden nodes, and three output nodes. Again, a hierarchical network can be fully or partially connected.

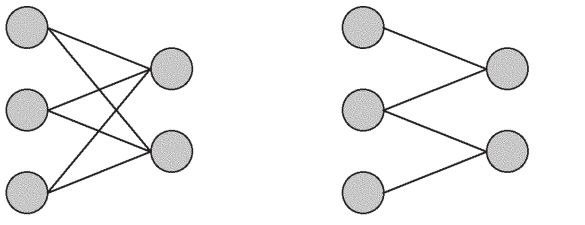


Figure 5 Examples of interlayer connections.

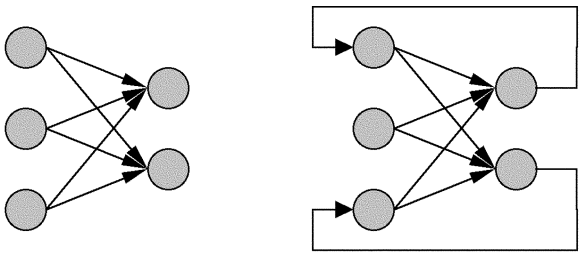


Figure 6 Examples of feedforward and feedback connections.

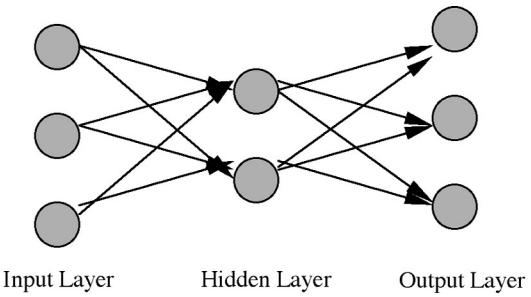


Figure 7 A hierarchical 3-layer connection example.

C. Multilayer Normal Feedward Network

A multilayer normal feedforward network, a fully connected hierarchical network, is the most popular network architecture implemented in neural network applications. Figure 7 is also an example of a multilayer normal feedforward network. Another version called a multilayer full feedforward network is a fully connected network but is different in two ways. First, all the nodes in the network receive external inputs, and second, each layer also receives input from the nodes in layers before it.

The connection strengths determine the relationship between the input and the output of the network and thus, in a way, represent the knowledge stored in the network. The neural network acquires this knowledge through a process of training during which the connection strengths between the nodes are modified. Once trained, the neural network retains this knowledge and it can be used for the particular task it was designed to do.

V. LEARNING ALGORITHMS

A neural network has to be configured such that the application of a set of inputs generates the desired set of outputs. The weights can be set explicitly using *a priori* knowledge when it's available. However, it is very difficult to assign appropriate weights to all links for a given problem, especially when there is little or no information about the population distribution. A general solution is to determine the weights through a training process that requires a substantial number of examples from which to learn and generalize. A typical learning algorithm will search through the space of all possible weights to arrive at a set of weights offering the best fit with the given examples. Various learning algorithms to set the connection strengths exist. Once a neural network architecture has been selected, we can start training the network by providing it with representative data. During the training process, the network learns by continuously adjusting its connection weights according to the input and output values provided. There are three kinds of weight modification: creation of new connections, loss of existing connections, and modification of the strengths of connections that already exist. The rules we use to systemically modify the connection weights are called learning algorithms.

A. Supervised vs Unsupervised Learning Algorithms

A number of different learning algorithms have been proposed and can be broadly categorized into two groups, supervised and unsupervised learning. Each learning rule has associated with it a number of parameters that determine its behavior and performance.

1. Supervised Learning Algorithms

Some well-known neural networks that implement supervised learning are the *Hebbian* network by Hebb, *Perceptron* by Minsky and Papert; *Adaline* by Widrow and Hoff, *Boltzman Machine* by Ackley *et al., backpropagation* by Rumelhart *et al., Cascade Correlation Network* by Fahlman and Lebiere, *Learning Vector Quantization* by Kohonen, and *ARTMAP* by Carpenter *et al.*

a. The Hebbian Learning Rule

Virtually all learning rules, supervised or unsupervised, can be considered as a variant of the *Hebbian* learning rule suggested by Hebb in his classic book *Organization of Behaviour* (1949). The basic idea is that if two processing units i and j are simultaneously excited, increase the strength of the connection between them. The Hebbian rule has a number of interpretations. If unit i receives input from unit j, the simplest version of Hebbian learning recommends modifing the weight w_{ij} with

$$\Delta w_{ij} = \alpha O_i O_j,$$

where $\Delta w_{ij} = w_{ij}^{\text{new}} - w_{ij}^{\text{old}}$ is the change to be made to the weight from unit j to unit i, and α is a positive constant of proportionality representing the learning rate. O_i and O_j are output from unit i and unit j, respectively. Another interpretation of Hebb's rule is to use not the actual output of unit i but the desired output of unit i,

$$\Delta w_{ij} = \alpha\ d_i\ O_j$$

in which d_i is the desired output provided as training data during the training process. This interpretation is used in many neurocomputing engineering applications of the Hebb rule. In this interpretation, in order for the Hebbian learning rule to learn the domain knowledge correctly, the data for the cases in the training data set should be orthogonal. This means there is no correlation among the data in the training cases. If a correlation exists, the system output would contain an error that is proportional to the correlation among the data in the training data set.

i. Learning Coefficient The learning coefficient (also classified as the learning rate) α varies in the range between 0 and 1. The higher the value of α, the faster the network learns. However, a higher learning rate also results in lower generalization capability. The generalization capability is important in neural network systems because it is the key to the fault-tolerance feature of neural networks that allows the system to keep functioning with noisy and incomplete input data. The appropriate setting for the learning rate α is problem dependent and is usually determined through a series of trial-and-error. In some implementation of the learning rules, the value of the learning rate is reduced as training progresses.

b. The Delta Learning Rule

Another common learning rule, the delta rule, developed by Widrow and Hoff uses not the actual or the desired output of unit i but the difference between the actual and desired activation for adjusting the weights,

$$\Delta w_{ij} = \alpha\ O_j\ (d_i - O_i),$$

where O_j is the output from unit j as input to unit i, d_i is the desired output of unit i, and O_i is the actual output of unit i. This rule is also known as the *Widrow–Hoff* rule or *least mean square* (LMS) rule. The delta rule is the prominent example of optimizing an objective function in determining the weight values. In the case of the delta rule, the objective is to minimize the sum of the squared errors, where an error is defined as the distance between an actual output of the system from the desired (or the correct) output, for a given input data. The delta rule essentially implements gradient descent in the sum-squared error for a linear transfer function.

c. The Generalized Delta Rule

The generalized delta rule is similar to the delta rule, but with the derivative of the transfer function added to the equation,

$$\Delta w_{ij} = \alpha\ O_j(d_i - O_i)\, f\left(\sum_{j=1}^{n} w_{ij}^{\text{old}}\ O_j\right),$$

where $f\,(\Sigma_{j=1}^{n}\ w_{ij}^{\text{old}}\ O_j)$ is the derivative of the transfer function that maps the total input to the unit to an output value. This rule is used when the transfer function is nonlinear, such as a sigmoid function.

i. Backpropagation Learning Algorithm The best-known supervised learning algorithm is *backpropagation*. The backpropagation learning algorithm, first designed to train a feedforward network, overcomes some of perceptron's limitations by making it possible to train a multiple-layer network. Backpropagation applies the generalized delta learning rule and is capable of exploiting the regularities and exceptions in the training sample. The learning algorithm consists of two phases: forward-propagation and backward-propagation. In forward-propagation, each input is fed to the input layer, and an output O_i is generated on the basis of the current weights. The value of O_i is then compared with the actual (or desired) output d_i by calculating the squared error at each output unit. Output differences are summed up to generate an error function E. The objective of the learning process is to minimize E by changing weights so that all input vectors are correctly mapped to their corresponding output vectors.

The second phase performs a gradient descent in the weight space to locate the optimal solution. The gradient vector of the error surface is calculated. This vector points along the line of steepest decent from the current point, so we know that if we move along it step-by-step through the weight change, we will decrease the error. A sequence of such moves will eventually reach the local minimum. The difficult part is to decide how large the steps should be. A large step size allows the network to converge more quickly, but may also overstep the optimal solution, oscillate between two or more states, or even go off in the wrong direction when the error surface is eccentric. On the other hand, a very small step size will result in more learning iterations and longer training time. The direction and magnitude of weight change are controlled by the learning coefficient α. The total squared error calculated in the first phase is propagated back, layer by layer, from the output units to the input units in the second phase. Weight adjustments are determined on the way of propagation at each level. The weights can be updated in two ways. They can be updated after each sample, or be accumulated and updated after a complete run of all examples. The two phases are executed in each iteration of the backpropagation algorithm until the objective function E converges. A momentum term is usually added to the learning algorithm to encourage movement in the same direction. This is done by gradually increasing the learning

rate when a number of steps have been taken in the same direction. This added feature gives the network the possibility of escaping local minimum, and also moving rapidly over flat spots and plateaus. Although the backpropagation algorithm does not guarantee a global optimal solution, Rumelhart *et al.* reported that solutions obtained from the algorithm come close to the optimal ones in their experiments.

2. Unsupervised Learning Algorithms

Popular neural networks that use unsupervised learning are *Linear Autoassociator* by Anderson *et al., Self-Organizing Map* by Kohonen, *Hopfield Network* by Hopfield, *Brain-State-in-a-Box* by Anderson *et al.,* and *Adaptive Resonance Theory (ART)* by Carpenter and Grossberg. In the following, we briefly discuss the self-organizing map network developed by Kohonen.

a. The Kohonen Self-Organizing Map Network

Teuve Kohonen developed the SOM network between 1979 and 1982 based on the earlier work of Willshaw and von der Malsburg. The SOM network performs unsupervised learning; its primary use is for tasks such as clustering, pattern recognition, and various optimization problems. It is designed to capture topologies and hierarchical structures of higher dimensional input spaces.

The SOM network typically has two layers of nodes, the input layer and the Kohonen layer. The input layer is fully connected to the Kohonen layer that is also the output layer. The Kohonen layer, the core of the SOM network, functions similarly to biological systems in that it can compress the representation of sparse data and spread out dense data using usually a one- or two-dimensional map. This is done by assigning different subareas of the Kohonen layer to the different categories of information; therefore the location of the processing element in a network becomes specific to a certain characteristic feature in the set of input data. The resulting network resembles the tree structure that can be derived by the conventional clustering method.

The network undergoes a self-organization process through a number of training cycles, starting with randomly chosen weights for the nodes in the Kohonen layer. During each training cycle, every input vector is considered in turn and the winner node is determined by the winner-take-all function. The weight vectors of the winning node and the nodes in the neighborhood are updated using a weight adaptation function based on the following Kohonen rule,

$$\Delta w_i = \alpha\ (X - w_i^{\text{old}}), \qquad \text{for } i \in N_r,$$

where α is the learning coefficient, X is the input vector, and N_r is the collection of all nodes in the neighborhood of radial distance r. For a two-dimensional Kohonen layer, there could be up to a total of eight neighboring nodes when $r = 1$ (see Fig. 8). The process will adjust the weights of the winning node along with its neighbor nodes closer to the value of the input pattern. The neighborhood size *(r)* can change and is usually reduced as training progresses.

VI. TRAINING AND TESTING

Training is the process by which a neural network learns. Training involves iterative presentation of the training data to the network with adjustment of weights to minimize the total network output error. A neural network needs to be trained so that the application of a set of inputs produces the desired set of outputs. When setting up a neural network, the first thing to do is to select a network architecture for the given task. The proper choice of network architecture can have a significant effect on the training and eventually the performance of the network. It also determines what learning rules are applicable. Once a network architecture has been selected, data can be collected for training the network. In supervised training, the network is trained by providing it with input and matching desired output patterns; each pair is also referred to as a sample. When the output layer uses an unsupervised learning method, desired outputs in the training set are not required.

A. The Training Process

Before training starts, the initial network weights are randomly assigned. The training progresses iteratively, through a number of epochs. On each epoch, the whole training samples are each submitted in turn to the network. The desired and actual outputs are then

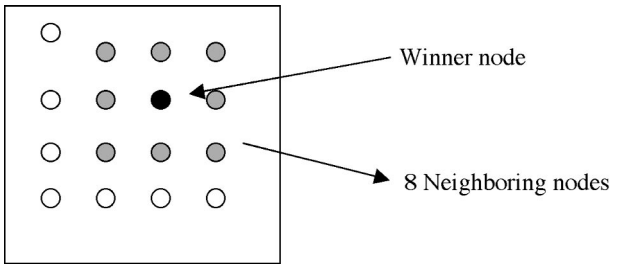


Figure 8 A 4 × 4 Kohonen layer with radial distance *(r)* = 1.

compared and the difference calculated. The difference, also known as the error term, is used to adjust the weights. The network weights are changed by an amount proportional to the error term. The rate of change is controlled by the parameter α, learning coefficient, and usually is implemented as a slowly decreasing function of time in order to guarantee convergence. After all samples have been processed by the network once, that is, after the completion of one epoch, then the whole process repeats.

B. Stopping Rules

The termination of the training process is controlled by one or more stopping rules. The most commonly used stopping conditions are when a given number of epochs has elapsed, when the error term reaches a predefined satisfactory level, and when the error stops improving after a certain number of epochs. For a complex network, it may take ten or hundreds of thousands of epochs before the training process terminates.

C. The Testing Process

After a network is fully trained, the performance of the network can be verified by using the test data that are new data the network has never been trained on. Testing is the process by which test data are presented to the network. The test set can be used to evaluate network performance during and after training. During the testing process, outputs and errors are computed, but the network weights are not adjusted.

D. Overfitting Problem

One known problem with network training is that a network may be over trained. This is also referred to as an overfitting problem. The problem can be recognized when the network performs very well with the training data, but deteriorates significantly when new data are used. The best way to avoid overfitting is to use a large number of cases for training. An overfitting problem is unlikely to occur when you have at least 30 times as many training cases as there are weights in the network. At least 5 times as many training cases as weights are needed when the data are noise-free. One approach to discover an overfitting problem is to test the network performance while the network is trained. A general guideline is to stop the training process when the error term of the test data stops improving.

E. Continuous Learning

One advantage of neural networks over statistical approaches is that a neural network allows adaptive adjustment to the weights as new examples become available. This is especially important when the underlying group distributions may be changing. Statistical methods assume old and new examples are equally valid, and when new data become available, the model needs to be rebuilt by using the entire data set. When the underlying distributions of the problem do not change, both old and new data can be used to retrain the network. However, in situations where the new data are drawn from a different distribution, weighting the importance of old data equally with new data may result in a predictive model with low accuracy. An important feature of a neural network is that it allows incremental learning, that is, the importance of old data can be reduced gradually as new samples are fed into the network. This is usually implemented by using a sliding window scheme that retains part of the old examples and combines it with the new ones to create a new training set. The exact proportion of old sample to be preserved depends on the stability of the distribution and the level of noise in the sample.

VII. NEURAL NETWORK APPLICATIONS

Although neural networks can be used for any application that involves function computation, they are best used for classification, regression, and function approximation/mapping problems that are tolerant of some imprecision. Many decision-making tasks are instances of classification problems or can be easily formulated into a classification problem, e.g., prediction and forecasting tasks, diagnosis tasks, and pattern recognition.

A. Regression and Classification

A neural network that applies supervised learning with quantitative target values is capable of performing regression analysis. The target value is the desired output value that we use to compare with the actual output from the network. Most regression algorithms can also be used for classification by encoding categorical

target values as 0/1 binary variables. The outputs of the network are posterior probabilities.

B. Linear vs Nonlinear Problems

Neural networks are very sophisticated modeling techniques, capable of modeling extremely complex functions, especially for problems that are nonlinear. For many years linear modeling has been used in most problem domains due to its well-known optimization strategies. Where the linear assumption does not hold, which is more often than rare, the models suffer accordingly.

C. Strengths and Weaknesses of Neural Networks

Neural networks have made strong advances in areas such as continuous speech recognition and synthesis, handwriting recognition, vision, and pattern recognition where traditional systems have been found intractable. However, neural networks do not do well at tasks that require precise, numerical computations. They are good at solving the kinds of problems people can solve easily. Neural networks should be considered as an alternative problem-solving tool only when there is no simple solution available using existing technology.

D. Neural Network Application Requirements

Other requirements for applying neural networks include a well defined input space, a fixed number of inputs, a large amount of training data available, and a well defined performance measure. This could take the form of a desired output from the network for each input example, or some other quantifiable goal or performance measure. The form of performance measure will guide the selection of the training algorithm.

E. Supervised Learning Applications

If a desired output is available for each training example, a supervised learning method should be used. Most applications such as problems involving classification or regression fall into this category. Examples include applications in bankruptcy prediction, stock market prediction, financial time series forecasting, speech recognition, image or character recognition, medical diagnosis, and robot control. Popular supervised learning algorithms include different variations of backpropagation learning, the Boltzmann Machine, Kohonon Learning Vector Quantization (LVQ), and Cascade Correlation. Each algorithm is actually a family of learning rules that vary with the parameter settings that can significantly shape performance and behavior.

F. Unsupervised Learning Applications

If a desired output is not known for each training example but some knowledge of relationship among a data set needs to be extracted, an unsupervised learning method may be appropriate. A clustering task that groups subjects into clusters of similar elements belongs to this category. An unsupervised learning method is a classification method where the network discovers its own classes. Examples include applications in market segmentation and group technology in flexible manufacturing systems. It has also been used a lot as a preprocessor for other statistical or neural network methods. When functioning as a preprocessor, the unsupervised learning networks have been applied to perform tasks such as dimension reduction, exploratory data analysis, and novelty detection. Popular unsupervised learning algorithms include various versions of ART for competitive learning, the Kohonen Self-Organizing Map for dimension reduction, and the Hopfield net for auto-association.

G. Future Developments

Significant progress has been made in the field of neural networks in the last two decades. It is continuously attracting a great deal of attention and research interests. The integration of neural networks with other artificial intelligence techniques such as fuzzy logic and genetic algorithms has been explored. Advancement beyond existing neural network applications is well expected, and a true thinking machine based on neural network techniques could be possible in the foreseeable future.

SEE ALSO THE FOLLOWING ARTICLES

Artificial Intelligence Programming • Engineering, Artificial Intelligence in • Evoluntionary Algorithms • Expert Systems Construction • Goal Programming • Health Care, Information Systems and • Hybrid Systems • Industry, Artificial Intelligence in • Knowledge Representation • Medicine, Artificial Intelligence in • Psychology • Robotics

BIBLIOGRAPHY

Bigus, J. P. (1996). *Data mining with neural networks: Solving business problems—From application development to decision support*. New York: McGraw–Hill.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford: Oxford Univ. Press.

Fausett, L. (1994). *Fundamentals of neural networks: Architectures, algorithms, and applications*. Englewood Cliffs, NJ: Prentice Hall.

Kohonen, T. (1995). *Self-organizing maps*. New York/Berlin: Springer-Verlag.

Masters, T. (1995). *Advanced algorithms for neural networks: A C++ sourcebook*. New York: John Wiley.

Reed, R. D., and Marks, R. J. II (1999). *Neural smithing, supervised learning in feedforward artificial neural networks*. Cambridge, MA: MIT Press.

Ripley, B. D. (1996). *Pattern recognition and neural networks*. Cambridge, UK: Cambridge Univ. Press.

Rumelhart, D. E., McClelland, J. L., and the PDP Research Group. (1986). *Parallel distributed processing*, Vols. I and II. Cambridge, MA: MIT Press.

Smith, M. (1996). *Neural networks for statistical modeling*. New York: Van Nostrand–Reinhold.

Number Representation and Computer Arithmetic

Behrooz Parhami
University of California, Santa Barbara

GLOSSARY

array multiplier A one-sided tree multiplier that isn't particularly fast, but has a regular structure suitable for pipelining and VLSI/FPGA realization.

carry lookahead Determining carries by receiving information from many lesser significant positions, instead of just the neighboring position.

convergence method Any computation method that produces a sequence of values, with each one closer than the previous value to a desired result.

fast carry network A circuit that computes all the intermediate carries needed in an adder much faster than would be possible with a ripple-carry scheme.

fixed-point number A number in which the position of the radix point is fixed and thus need not be explicitly specified.

floating-point number A number in which the position of the radix point is variable and thus must be explicitly specified by means of a scale factor or exponent.

full-adder A logic circuit that can add two digits and an incoming carry, producing a sum digit and an outgoing carry.

high-radix arithmetic Arithmetic on numbers represented in a radix greater than 2.

redundant number A number represented in radix r using more than r different digit values.

residue arithmetic Arithmetic performed by manipulating not the original operands, but their residues with respect to a set of predefined moduli.

ripple-carry adder An adder in which carry into stage $i + 1$ is determined based on carry into stage i, thus leading to long carry propagation delay.

rounding Getting rid of extra digits in a computation result to make it fit in a fixed format, while controlling the amount or the direction of the resultant error.

signed magnitude A method of representing signed values by means of attaching a sign bit to an unsigned magnitude.

table-lookup scheme Obtaining the result of a desired operation by consulting a table of precomputed values.

tree multiplier The fastest possible multiplier in which all the required partial products are generated at once and added up in parallel to yield the final product.

two's complement A number representation scheme that converts negative values into unsigned values by adding a suitably large power of 2 to them.

ARITHMETIC is a branch of mathematics that deals with numbers and numerical computation. Arithmetic operations on pairs of numbers x and y include addition, producing the sum $s = x + y$, subtraction, yielding the difference $d = x - y$, multiplication, resulting in the product $p = x \times y$, and division, generating the quotient $q = x/y$ (and, in the case of integer division, the remainder $z = x \bmod y$). Subtraction and division can be viewed as operations that undo the effects of addition and multiplication, respectively. Computer arithmetic is a branch of computer engineering that deals with methods of representing integers and real values (e.g., fixed- and floating-point numbers) in digital systems and efficient algorithms for manipulating such numbers by means of hardware circuits or software routines. On the hardware side, various types of adders, subtractors, multipliers, dividers, square-rooters, and circuit techniques for function evaluation are considered. Both abstract structures and technology-specific designs are dealt with. Software aspects of computer arithmetic include complexity, error characteristics, stability, and certifiability of computational algorithms.

I. NATURAL NUMBERS

When we think of numbers, it is usually the *natural numbers* that first come to our mind; the type of numbers that sequence book or calendar pages, mark clock dials, flash on stadium scoreboards, and guide deliveries to our houses. The set {0, 1, 2, 3, . . .} of natural numbers, also known as *whole numbers* or *unsigned integers,* forms the basis of arithmetic. We use natural numbers for counting and for answering questions that ask: How many?

Four-thousand years ago, Babylonians knew about natural numbers and were proficient in arithmetic. Since then, representations of natural numbers have advanced in parallel with the evolution of language. Ancient civilizations used sticks and pebbles to record inventories or accounts. When the need for larger numbers arose, the idea of grouping sticks or pebbles simplified counting and comparisons; for example, 27 was represented by five groups of five sticks, plus two sticks. Eventually, objects of different shapes or colors were used to denote such groups, leading to more compact representations.

Numbers must be differentiated from their representations, sometimes called *numerals.* For example, the number "twenty-seven" can be represented in different ways using various numerals or *numeration systems;* these include:

||||| ||||| ||||| ||||| ||||| ||	Sticks or *unary* code
27	Radix-10 or *decimal* code
11011	Radix-2 or *binary* code
XXVII	Roman numerals

However, we don't always make the distinction between numbers and numerals and may use "decimal numbers" in lieu of "decimal numerals."

Roman numerals were used in Europe during the Middle Ages. Even when Europeans learned that the Arabs had a better way of representing numbers, it took them centuries to adopt the *Arabic system* based on numerals, or *digits,* 0–9 and a radix of 10. In these *decimal numbers,* the worth of each position is 10 times that of the adjacent position to its right, so that the string of digits "5327" represents five thousands, plus three hundreds, plus two tens, plus seven ones.

Other radices have also appeared over the ages. For several examples, see A. Glaser's *History of Binary and Other Nondecimal Numeration.* Babylonians used radix-60 numbers, which make dealing with time easy. The radices 12 *(duodecimal)* and 5 *(quinary)* have also been used. The use of radix-2 *(binary)* numbers became popular with the onset of electronic computers, because their use of binary digits, or *bits,* having only two possible values 0 and 1, is compatible with electronic signals. Radix-8 *(octal)* and radix-16 *(hexadecimal)* numbers have been used as shorthand notation for binary numbers. For example, a 24-bit binary number can be represented as an 8-digit octal or a 6-digit hexadecimal number by taking the bits in groups of threes and fours, respectively.

In a general radix-r *positional number system,* with a fixed word width of k, a number x is represented by a string of k digits x_i, with $0 \leq x_i \leq r - 1$:

$$x = \Sigma_{i=0}^{k-1} x_i r^i = (x_{k-1}x_{k-2} \ldots x_1x_0)_r \qquad (1)$$

For example:

$$27 = (1 \times 2^4) + (1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) = (11011)_{\text{two}}$$

In a k-digit radix-r number system, natural numbers from 0 to $r^k - 1$ can be represented. Conversely, given a desired representation range $[0, M - 1]$, the required number k of digits in radix r is obtained from the following equation:

$$k = \lceil \log_r M \rceil = \lfloor \log_r (M - 1) \rfloor + 1 \qquad (2)$$

For example, representing the decimal number 3125 requires 12 bits in radix 2, five digits in radix 5, and four digits in radix 8.

Given a number x represented in radix r, one can obtain its radix-R representation in two ways. If we wish to perform arithmetic in the new radix R, we simply evaluate a polynomial in r whose coefficients are the digits x_i. This corresponds to Eq. (1) and can be performed more efficiently by using Horner's rule which involves repeated steps of multiplying by r followed by addition:

$$(x_{k-1}x_{k-2} \ldots x_1x_0)_r = ((\ldots(x_{k-1}r + x_{k-2})r + \ldots x_1)r + x_0) \quad (3)$$

This method is particularly suitable for manual conversion from an arbitrary radix r to radix 10, given the relative ease with which we can perform radix-10 arithmetic.

To perform the radix conversion using arithmetic in the old radix r, we repeatedly divide the number x by the new radix R, keeping track of the remainder in each step. These remainders correspond to the radix-R digits X_i, beginning from X_0. For example, we convert the decimal number 23 to radix 2 as follows:

23 divided by 2 yields 11 with remainder 1
11 divided by 2 yields 5 with remainder 1
5 divided by 2 yields 2 with remainder 1
2 divided by 2 yields 1 with remainder 0
1 divided by 2 yields 0 with remainder 1

Reading the computed remainders from bottom to top, we find $23 = (10111)_{\text{two}}$. Using the same process, we can convert 23 to radix 5 to get $23 = (43)_{\text{five}}$.

II. DIGIT SETS AND ENCODINGS

The standard digit set used for radix-r numbers is $\{0, 1, \ldots, r-1\}$. This digit set leads to a unique representation for every natural number. The binary or radix-2 digit set is $\{0, 1\}$ which is conveniently representable by electronic signals. Typically, low voltage is used to represent 0 and high voltage denotes 1, but the reverse polarity can also be used. Conceptually, the decimal digit values 0 through 9 could be represented by 10 different voltage levels. However, encoding everything with 0s and 1s makes it easier for electronic circuits to interpret and process the information speedily and reliably.

One way to encode decimal digits using binary signals is to encode each of the digits 0–9 by means of its 4-bit binary representation. The resulting binary-coded decimal (BCD) representation is shown below:

Digit	BCD representation
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1

The use of digit values 0 through $r - 1$ in radix r is just a convention. We could use more than r digit values (for example, digit values -2 to 2 in radix 4) or use r digit values that do not start with 0 (for example, digit set $\{-1, 0, 1\}$ in radix 3). In the first instance, the resulting number system possesses redundancy in that some numbers will have multiple representations. More on this in Section XVI.

Of course, any finite set of symbols, not just digits, can be encoded using binary signals. The American Standard Code for Information Interchange (ASCII) is one such convention that represents upper- and lower-case letters, numerals, punctuation marks, and other symbols in an 8-bit *byte*. For example, the 8-bit ASCII codes for the 10 decimal digits are of the form 0011xxxx, where the "xxxx" part is identical to the BCD code discussed earlier. ASCII digits take twice as much space as BCD digits and thus are not used in arithmetic units. Even less compact than ASCII is the 16-bit Unicode which can accommodate symbols from many different languages.

III. INTEGERS

The set $\{\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots\}$ of integers is also referred to as *signed or directed (whole) numbers*. The most straightforward representation of integers consists of attaching a sign bit to any desired representation of natural numbers, leading to *signed-magnitude* representation. The standard convention is to use 0 for positive and 1 for negative and attach the

sign bit to the left end of the magnitude. Here are two examples:

+27 in 8-bit signed-magnitude binary code	00011011
−27 in 2-digit decimal code with BCD digits	1 0010 0111

Another option for encoding signed integers in the range $[-N, P]$ is the *biased representation*. If we add the positive value N (the bias) to all numbers in the desired range, unsigned integers in the range $[0, P + N]$ result. Any method for representing natural numbers in $[0, P + N]$ can then be used for representing the original signed integers in $[-N, P]$. This type of biased representation has only limited application in encoding of the exponents in floating-point numbers (see Section XI).

By far the most common machine encoding of signed integers is the 2'*s-complement representation*. In the k-bit 2's-complement format, a negative value $-x$, with $x > 0$, is encoded as the unsigned number $2^k - x$. Figure 1 shows encodings of positive and negative integers in the 4-bit 2's-complement format. Note that the positive integers 0 through 7 (or $2^{k-1} - 1$, in general) have the standard binary encoding, whereas negative values −1 through −8 (or -2^{k-1}, in general) have been transformed to unsigned values by adding 16 (or 2^k, in general) to them.

Two important properties of 2's-complement numbers are worth noting. First, the leftmost bit of the representation acts as the sign bit (0 for positive values, 1 for negative ones). Second, the value represented by a particular bit pattern can be derived without a need to follow different procedures for negative and positive values. We simply use Eq. (1), as we did for unsigned integers, except that the sign bit is considered to have a negative weight. Here are two examples:

$$(01011)_{\text{2's-compl}} = (-0 \times 2^4) + (1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) = +11$$

$$(11011)_{\text{2's-compl}} = (-1 \times 2^4) + (1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) = -5$$

The reason for the popularity of 2's-complement representation will become clear when we discuss addition and subtraction in Section VI.

Other complement representation systems can also be devised, but none is in widespread use. Choosing any *complementation constant M*, that is at least as large as $N + P + 1$, allows us to represent signed integers in the range $[-N, P]$, with the positive numbers in $[0, +P]$ corresponding to the unsigned values in $[0, P]$ and negative numbers in $[-N, -1]$ represented as unsigned values in $[M - N, M - 1]$. Sometimes, M itself is used as an alternate code for 0 (actually, −0). For example, the k-bit 1'*s-complement system* is based on $M = 2^k - 1$ and includes numbers in the range $[-(2^{k/2} - 1), 2^{k/2} - 1]$, with 0 having two representations: the all-0s string and the all-1s string.

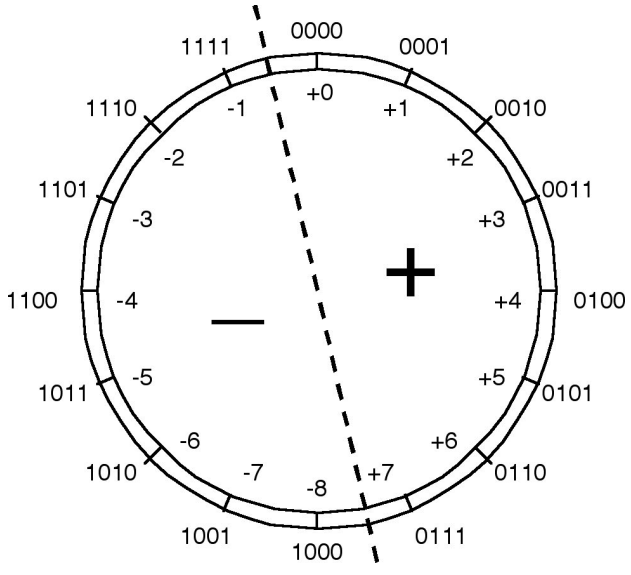


Figure 1 Schematic representation of 4-bit 2's-complement code for integers in [−8, +7].

IV. COUNTING

The natural numbers are ordered. Each natural number x has a successor $succ(x)$, which is the next number in this order and, except when $x = 0$, it has a predecessor $pred(x)$. *Counting* is the act of going through the successor or predecessor sequence, beginning with some initial value. So, (3, 4, 5, 6, . . .) is an up-counting sequence and (15, 14, 13, 12, . . .) is a down-counting sequence. For a detailed treatment of various types of counters, see R.M.M. Oberman's *Counting and Counters,* Macmillan, 1981. Hardware circuits that mimic this process, advancing to the successor number or retrogressing to the predecessor each time a count control signal is asserted, are known as *up-counters* and *down-counters,* respectively. A k-bit modulo-2^k down-counter would go into negative 2's-complement values if it is decremented past zero. An *up/down-counter* can go in either direction, depending on the value of a direction control signal.

In what follows, we focus on up-counters that hold unsigned values. *Asynchronous counters* can be built of

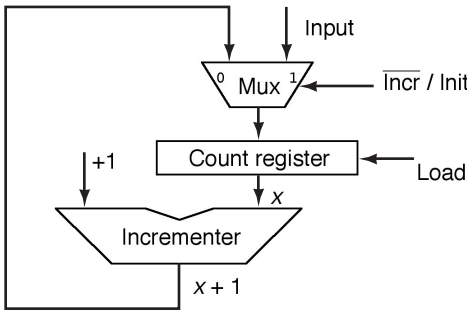


Figure 2 Synchronous binary counter with initialization capability.

edge-triggered storage elements (flip-flops) and nothing else. The more commonly used *synchronous counters* are built of a register, a hardware incrementer, and some logic that allows the incremented count or an initial value to be loaded into the register when appropriate control signals are asserted. Figure 2 shows the block diagram for a synchronous counter.

The counter design shown in Fig. 2 is adequate for most applications. It can be made faster by using a fast incrementer with carry-lookahead feature similar to that used in fast adders, to be discussed in Section VII. If still higher speed is required, the counter can be divided into blocks. A short initial block (say, 3 bits wide) can easily keep up with the fast incoming signals. Increasingly wider blocks to the left of the initial block need not be as fast because they are adjusted less and less frequently.

V. FIXED-POINT NUMBERS

A fixed-point number consists of a *whole* or *integral* part and a *fractional* part, with the two parts separated by a *radix point* (*decimal point* in radix 10, *binary point* in radix 2, and so on). The position of the radix point is almost always implied and thus the point is not explicitly shown. If a fixed-point number has k whole digits and l fractional digits, its value is obtained from the formula:

$$x = \sum_{i=-l}^{k-1} x_i\, r^i = (x_{k-1}x_{k-2} \ldots x_1x_0 \cdot x_{-1}x_{-2} \ldots x_{-l})_r \quad (4)$$

In other words, the digits to the right of the radix point are given negative indices and their weights are negative powers of the radix. For example:

$$2.375 = (1 \times 2^1) + (0 \times 2^0) + (0 \times 2^{-1}) + (1 \times 2^{-2}) + (1 \times 2^{-3}) = (10.011)_{\text{two}}$$

In a $(k + l)$-digit radix-r fixed-point number system with k whole digits, numbers from 0 to $r^k - r^{-l}$, in increments of r^{-l}, can be represented. The step size or resolution r^{-l} is often referred to as *ulp,* or *unit in least position*. For example, in a (2 + 3)-bit binary fixed-point number system, we have $ulp = 2^{-3}$ and the values $0 = (00.000)_{\text{two}}$ through $2^2 - 2^{-3} = 3.875 = (11.111)_{\text{two}}$ are representable. For the same total number $k + l$ of digits in a fixed-point number system, increasing k will lead to enlarged *range* of numbers, whereas increasing l leads to greater *precision*. Therefore, there is a trade-off between range and precision.

Signed fixed-point numbers can be represented by the same methods discussed for signed integers: signed-magnitude, biased format, and complement method. In particular, for 2's-complement format, a negative value $-x$ is represented as the unsigned value $2^k - x$. Figure 3 shows encodings of positive and negative integers in the (1 + 3)-bit fixed-point 2's-complement format. Note that the positive values 0 to 7/8 (or $2^{k-1} - 2^{-l}$, in general) have the standard binary encoding, whereas negative values $-1/8$ to -1 (or -2^{-l} to -2^{k-1}, in general) are transformed to unsigned values by adding 2 (or 2^k, in general) to them.

The two important properties of 2's-complement numbers, previously mentioned in connection with integers, are valid here as well; namely, the leftmost bit of the number acts as the sign bit, and the value represented by a particular bit pattern can be derived

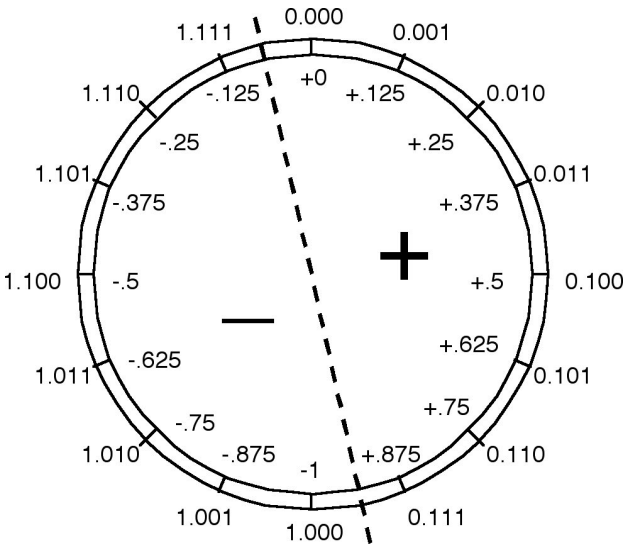


Figure 3 Schematic representation of 4-bit 2's-complement encoding for (1 + 3)-bit fixed-point numbers in the range [−1, +7/8].

by considering the sign bit as having a negative weight. Here are two examples:

$$(01.011)_{2\text{'s-compl}} = (-0 \times 2^1) + (1 \times 2^0) + (0 \times 2^{-1}) + (1 \times 2^{-2}) + (1 \times 2^{-3}) = +1.375$$

$$(11.011)_{2\text{'s-compl}} = (-1 \times 2^1) + (1 \times 2^0) + (0 \times 2^{-1}) + (1 \times 2^{-2}) + (1 \times 2^{-3}) = -0.625$$

Conversion of fixed-point numbers from radix r to another radix R is done separately for the whole and fractional parts. Converting the whole part was discussed in Section I. To convert the fractional part, we can again use arithmetic in the new radix R or in the old radix r, whichever is more convenient. With radix-R arithmetic, we simply evaluate a polynomial in r^{-1} whose coefficients are the digits x_i. The simplest way to do this is to view the fractional part as an l-digit integer, convert this integer to radix R, and divide the result by r^l.

To perform radix conversion using arithmetic in the old radix r, we repeatedly multiply the fraction y by the new radix R, noting and removing the integer part in each step. These integer parts correspond to the radix-R digits X_{-i}, beginning from X_{-1}. For example, we convert 0.175 to radix 2 as follows:

0.175 multiplied by 2 yields 0.350 with integer part 0
0.350 multiplied by 2 yields 0.700 with integer part 0
0.700 multiplied by 2 yields 0.400 with integer part 1
0.400 multiplied by 2 yields 0.800 with integer part 0
0.800 multiplied by 2 yields 0.600 with integer part 1
0.600 multiplied by 2 yields 0.200 with integer part 1
0.200 multiplied by 2 yields 0.400 with integer part 0
0.400 multiplied by 2 yields 0.800 with integer part 0

Reading the recorded integer parts from top to bottom, we find $0.175 = (0.00101100)_{\text{two}}$. This equality is approximate because the result did not converge to 0. In general a fraction in one radix may not have an exact representation in another radix. In any case, we simply carry out the process above until the required number of digits in the new radix have been obtained.

VI. ADDITION AND SUBTRACTION

We will cover only integer addition and subtraction. Fixed-point numbers that are both in the same format can be added like integers by simply ignoring the implied radix point. When two bits are added, the sum is a value in the range [0, 2] which can be represented by a *sum bit* and a *carry bit*. The circuit that can compute the sum and carry bits is known as a half-adder (HA) with its truth table and symbolic representation shown in Fig. 4. The carry output is the logical AND of the two inputs, while the sum output is the exclusive OR (XOR) of the inputs. By adding a carry input to an HA, we get a binary full-adder (FA) whose truth table and schematic diagram are depicted in Fig. 5.

Inputs		Outputs	
x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

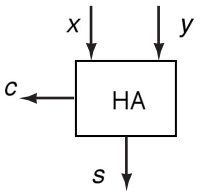


Figure 4 Truth table and schematic diagram for a binary HA.

An FA, connected to a flip-flop for holding the carry bit from one cycle to the next, functions as a *bit-serial adder*. The inputs of a bit-serial adder are supplied in synchrony with a clock signal, one bit from each operand per clock cycle, beginning from the least-significant bits. One bit of the output is produced per clock cycle and the carry from one cycle is held and used as input in the next cycle. A *ripple-carry adder*, on the other hand, unfolds this sequential behavior into space, using a cascade of k FAs to add two k-bit numbers (Fig. 6).

The ripple-carry design of Fig. 6 becomes a radix-r adder if each binary FA is replaced by a radix-r FA that accepts two radix-r digits (each encoded in binary) and a carry-in signal, producing a radix-r sum digit and a carry-out signal. Because the carry signals are always binary and their propagation is independent of the radix r, in the rest of this section and in Section VII, we do not deal with radices other than 2.

An adder/subtractor for 2's-complement numbers can be built by adding a controlled complementation circuit (a two-way multiplexer with y and its bitwise complement as the two inputs) to an adder of any type. The resulting circuit is shown in Fig. 7. To jus-

Inputs			Outputs	
x	y	c_{in}	c_{out}	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

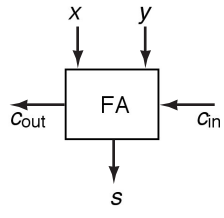


Figure 5 Truth table and schematic diagram for a binary FA.

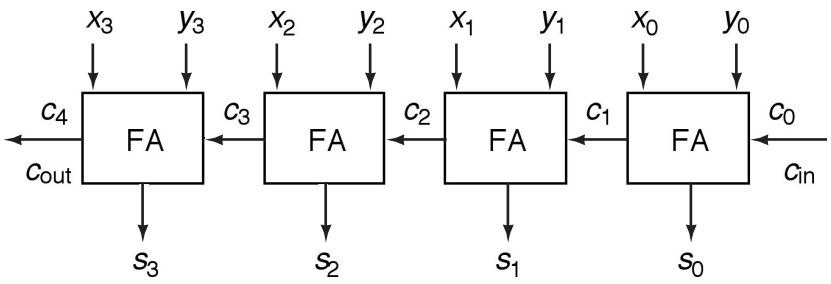


Figure 6 Four-bit ripple-carry binary adder.

tify this design, note that $x - y$ can be computed by forming the 2's-complement of y and adding it to x. However, the 2's-complement of y, which is $2^k - y$, can be computed by adding 1 to $(2^k - 1) - y$, the bitwise complement of y. The addition of 1 is accomplished by inserting a carry-in of 1 into the adder when it is used to perform subtraction.

VII. FAST ADDERS

Ripple-carry adders are quite simple and easily expandable to any desired width. However, they are rather slow, because carries may propagate across the full width of the adder. This happens, for example, when the two 8-bit numbers 10101011 and 01010101 are added. Because each FA requires some time to generate its carry output, cascading k such units together implies k times as much signal delay in the worst case. This linear amount of time becomes unacceptable for wide words (say, 32 or 64 bits) or in high-performance computers, though it might be acceptable in an embedded system that is dedicated to a single task and is not expected to be fast.

A variety of fast adders can be designed that require logarithmic, rather than linear, time. In other words, the delay of such fast adders grows as the logarithm of k. The best-known and most widely used such adders are *carry-lookahead adders*. The basic idea in carry-lookahead addition is to form the required intermediate carries directly from the inputs rather than from the previous carries, as was done in Fig. 6. For example, the carry c_3 in Fig. 6, which is normally expressed in terms of c_2 using the carry recurrence

$$c_3 = x_2y_2 + (x_2 \oplus y_2)c_2$$

can be directly derived from the inputs based on the logical expression:

$$c_3 = x_2y_2 + (x_2 \oplus y_2)x_1y_1 + (x_2 \oplus y_2)(x_1 \oplus y_1)x_0y_0 + (x_2 \oplus y_2)(x_1 \oplus y_1)(x_0 \oplus y_0)c_0$$

To simplify the rest of our discussion of fast adders, we define the *carry generate* and *carry propagate* signals and use them in writing a carry recurrence that relates c_{i+1} to c_i:

$$g_i = x_iy_i$$

$$p_i = x_i \oplus y_i$$

$$c_{i+1} = g_i + p_ic_i$$

The expanded form of c_3, derived earlier, now becomes:

$$c_3 = g_2 + p_2g_1 + p_2p_1g_0 + p_2p_1p_0c_0$$

It is easy to see that the expanded expression above would grow quite large for a wider adder that requires c_{31} or c_{52}, say, to be derived. A variety of *lookahead carry networks* exist that systematize the preceding derivation for all the intermediate carries in parallel and make the computation efficient by sharing parts of the required circuits whenever possible. Various designs offer trade-offs in speed, cost, layout area (amount of space on a VSLI chip needed to accomodate the required circuit elements and wires that interconnect them), and power consumption. Information on the design of fast carry networks and other types of fast adders can be found in books on computer arithmetic.

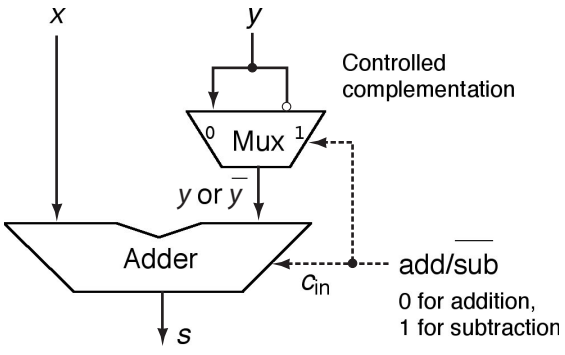


Figure 7 Two's-complement adder/subtractor.

Here, we present just one example of a fast carry network. The building blocks of this network are the *carry operator* which combines the generate and propagate signals for two adjacent blocks $[i + 1, j]$ and $[h, i]$ of digit positions into the respective signals for the wider combined block $[h, j]$. In other words

$$[i + 1, j] \diamond [h, i] = [h, j]$$

where $[a, b]$ stands for $(g_{[a,b]}, p_{[a,b]})$ representing the pair of generate and propagate signals for the block extending from digit position a to digit position b. Because the problem of determining all the carries c_i is the same as computing the cumulative generate signals $g_{[0,i]}$, a network built of $\diamond$ operator blocks, such as the one depicted in Fig. 8, can be used to derive all the carries in parallel. If a c_{in} signal is required for the adder, it can be accommodated as the generate signal g_{-1} of an extra position on the right.

For a k-digit adder, the number of operator blocks on the critical path of the carry network exemplified by Fig. 8 is $2(\lceil \log_2 k \rceil - 1)$. Many other carry networks can be designed that offer speed-cost trade-offs.

An important method for fast adder design, that often complements the carry-lookahead scheme, is *carry-select*. In the simplest application of the carry-select method, a k-bit adder is built of a $(k/2)$-bit adder in the lower half, two $(k/2)$-bit adders in the upper half (forming two versions of the $k/2$ upper sum bits with $c_{k/2} = 0$ and $c_{k/2} = 1$), and a multiplexer for choosing the correct set of values once $c_{k/2}$ becomes known. A hybrid design, in which some of the carries (say, c_8, c_{16}, and c_{24} in a 32-bit adder) are derived via carry-lookahead and are then used to select one of two versions of the sum bits that are produced for 8-bit blocks concurrently with the operation of the carry network, is quite popular in modern arithmetic units.

VIII. MULTIPLICATION

The simplest machine multipliers are designed to follow a variant of the pencil-and-paper multiplication algorithm depicted in Fig. 9, where each row of dots in the *partial products bit-matrix* is either all 0s (if the corresponding $y_i = 0$) or the same as x (if $y_i = 1$). When we perform a $k \times k$ multiplication manually, we form all of the k partial products and add the resulting k numbers to obtain the product p.

For machine execution, it is easier if the *cumulative partial product* is initialized to 0, each row of the bit-matrix added to it as the corresponding term is generated, and the result of addition shifted to the right by one bit to achieve proper alignment with the next term, as depicted in Fig. 9. In fact, this is exactly how *programmed multiplication* is performed on a machine that does not have a hardware multiply unit. The recurrence equation describing the process above is:

$$p^{(j+1)} = (p^{(j)} + y_j x 2^k)\ 2^{-1} \text{ with}$$
$$\text{|— add —|}$$
$$\text{|— shift right —|}$$
$$p^{(0)} = 0 \text{ and } p^{(k)} = p \quad (5)$$

Because by the time we are done, the right shifts will have caused the first partial product to be multiplied by 2^{-k}, we premultiply x by 2^k to offset the effect of these right shifts. This is not an actual multiplication

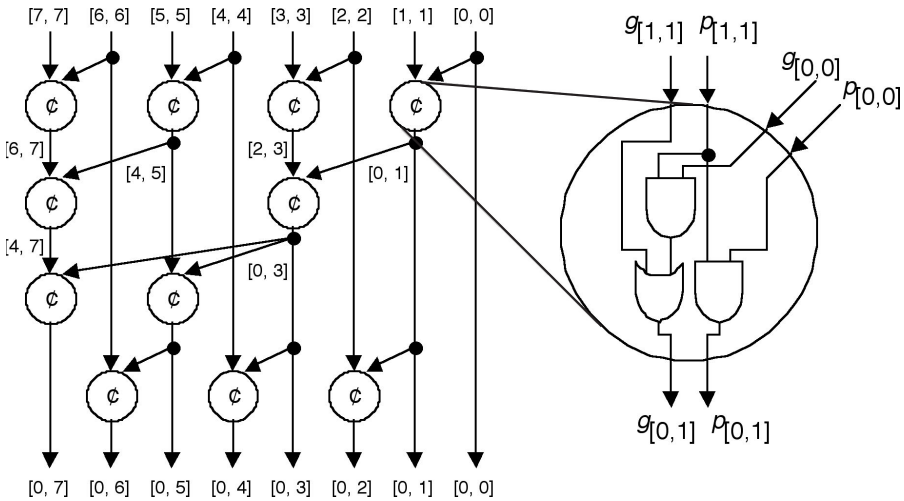


Figure 8 Brent-Kung lookahead carry network for an 8-digit adder.

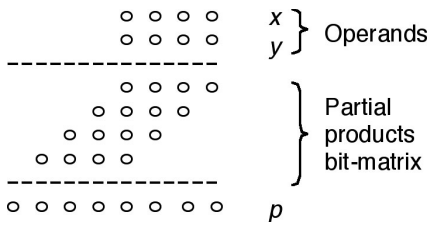


Figure 9 Multiplication of 4-bit numbers in dot notation.

but is done by aligning x with the upper half of the $2k$-bit cumulative partial product in the addition steps. Figure 10 depicts a possible hardware realization of the foregoing *shift-add multiplication* algorithm. The shifting of the partial product need not be done in a separate step but can be incorporated in the connecting wires that go from the adder output to the doublewidth register.

After k iterations, recurrence (5) leads to:

$$p^{(k)} = xy + p^{(0)}2^{-k}$$

Thus if $p^{(0)}$ is initialized to $2^k z$ (z padded with k zeros) instead of 0, the expression $xy + z$ will be evaluated. This *multiply-add operation* is quite useful for many applications and is performed at essentially no extra cost compared to plain shift-add multiplication.

The preceding bit-at-a-time multiplication scheme can be easily extended to a digit-at-a-time algorithm in a higher radix such as 4, 8, or 16. In this case, the multiplexer in Fig. 10 (which is really a bit-by-number multiplier) must be replaced with a digit-by-number multiplier circuit (perhaps implemented as a multi-

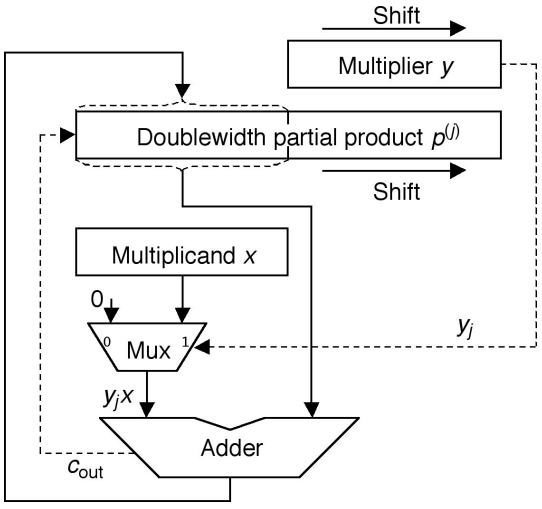


Figure 10 Hardware multiplier based on the shift-add algorithm.

operand adder), the single-bit shifts replaced by h-bit shifts for radix-2^h algorithm, and the number of iterations reduced to from k to k/h. These faster *high-radix multipliers* may still be too slow for some applications. In such cases, fully combinational *tree multipliers* are used in which the addition of the partial products bit-matrix is done by means of a tree-structured combinational circuit.

IX. FAST MULTIPLIERS

Instead of developing the partial products one at a time in radix 2 or in radix 2^h, we can form all of them simultaneously, thus reducing the multiplication problem to n-operand addition, where $n = k$ in radix 2, $n = k/2$ in radix 4, and so on. For example, a 16×16 multiplication becomes a 16-operand addition problem in radix 2 or an 8-operand addition problem in radix 4.

In *tree multipliers,* the n operands thus formed are added in two stages. In stage 1, a tree built of *carry-save adders* or similar *compression circuits* is used to reduce the n operands to two operands that have the same sum as the original n numbers. A carry-save adder (see Section XVI) reduces three values to two values, for a reduction factor of 1.5, thus leading to a $\lceil \log_{1.5}(n/2) \rceil$-level circuit for reducing n numbers to two. The two numbers thus derived are then added by a fast logarithmic-time adder, leading to an overall logarithmic latency for the multiplier (Fig. 11).

Such a *full-tree multiplier* is rather complex and its speed may not be needed for all applications. In such cases, more economical *partial-tree multipliers* might be implemented. For example, if about half of the partial products are accommodated by the tree part, then two passes through the tree can be used to form the two numbers representing the desired product, with the results of the first pass fed back to the inputs and combined with the second set of partial products (Fig. 11). A partial-tree multiplier can be viewed as a (very) high-radix multiplier. For example, if 12 partial products are combined in each pass, then a radix-2^{12} multiplication is effectively performed.

An *array multiplier* uses the same two-stage computation scheme of a tree multiplier, with the difference that the tree of carry-save adders is one-sided (has the maximum possible depth) and the final adder is of ripple-carry type (quite slow). An example 4×4 array multiplier is depicted in Fig. 12. Cells marked HA and FA cells are half- and full-adders defined in Figs. 4 and 5, respectively, and MA cells are modified full-adders, one of whose inputs is internally formed as the logical AND of x_i and y_j.

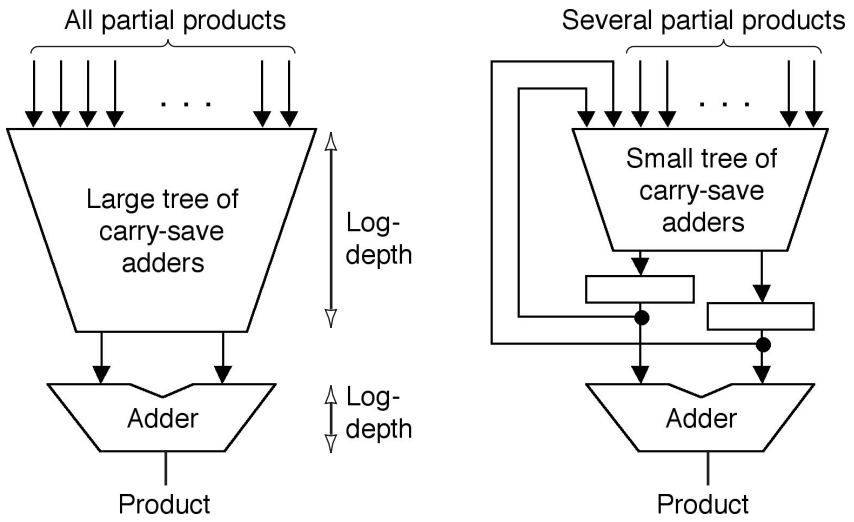


Figure 11 Schematic diagrams for full- and partial-tree multipliers.

The reader may well ask why such a slow tree-type multiplier is of any interest at all. The answer is that array multipliers are quite suitable for VLSI realization, given their highly regular design and efficient wiring pattern. They can also be readily pipelined by inserting latches between some of the rows of cells, thus allowing several multiplications to be performed on the same hardware structure.

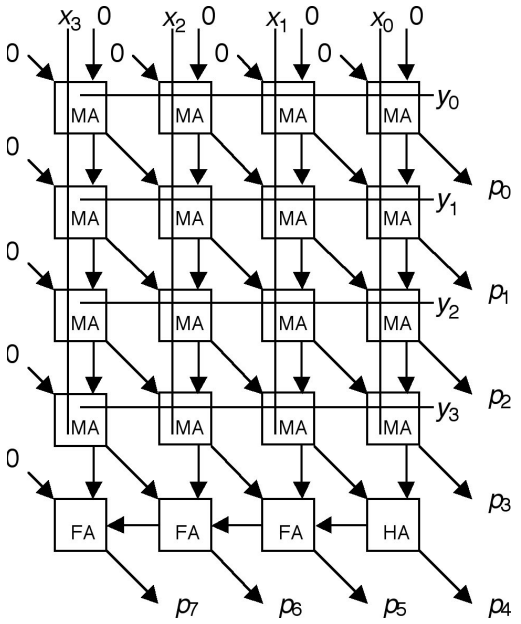


Figure 12 Array multiplier for 4-bit unsigned operands.

X. DIVISION

Like multipliers, the simplest machine dividers are designed to follow a variant of the pencil-and-paper division algorithm depicted in Fig. 13, where each row of dots in the subtracted bit-matrix is either all 0s (if the corresponding $q_i = 0$) or the same as y (if $q_i = 1$). When we perform a $2k/k$ division manually, we form the subtracted terms one at a time by "guessing" the value of the next *quotient digit*, subtract the appropriate term (0 or a suitably shifted version of y) from the *partial remainder* (initialized to the value of the dividend x), and proceed until all k bits of q have been determined. At this time, the partial remainder becomes the final remainder z.

For hardware or software implementation, a recurrence equation describing the process above is used:

$$z^{(j+1)} = 2\ z^{(j)} - q_{k-j}\ y\ 2^k \text{ with}$$
$$\text{|shift|}$$
$$\text{|— subtract —|}$$
$$z^{(0)} = x \text{ and } z^{(k)} = 2^k z \quad (6)$$

Because by the time we are done, the left shifts will have caused the partial remainder to be multiplied by 2^k, the true remainder is obtained by multiplying the final partial remainder by 2^{-k} (shifting it to the right by k bits). Figure 14 depicts a possible hardware realization of the foregoing *shift-subtract division* algorithm. As in multiplication, the shifting of the partial remainder need not be done in a separate step but can be incorporated in the wires connecting the adder output to the partial remainder register.

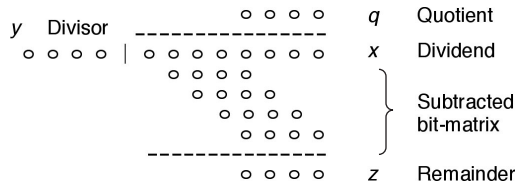


Figure 13 Division of an 8-bit number by a 4-bit number in dot notation.

A comparison of Figs. 10 and 14 reveals that multipliers and dividers are quite similar and can be implemented with shared hardware within an arithmetic/logic unit (ALU) that performs different operations based on an externally supplied *function code*.

As in the case of multipliers, *high-radix dividers* speed up the division process by producing several bits of the quotient in each cycle. Whereas there is no counterpart to fast tree multipliers for performing division, *array dividers* do exist and are structurally quite similar to the array multiplier of Fig. 12. It is also possible to perform division by using a sequence of multiplications instead of additions. Even though multiplications are slower and costlier to implement than additions, advantage over additive schemes may be gained because far fewer multiplications are needed to perform division. Details can be found in any book on computer arithmetic.

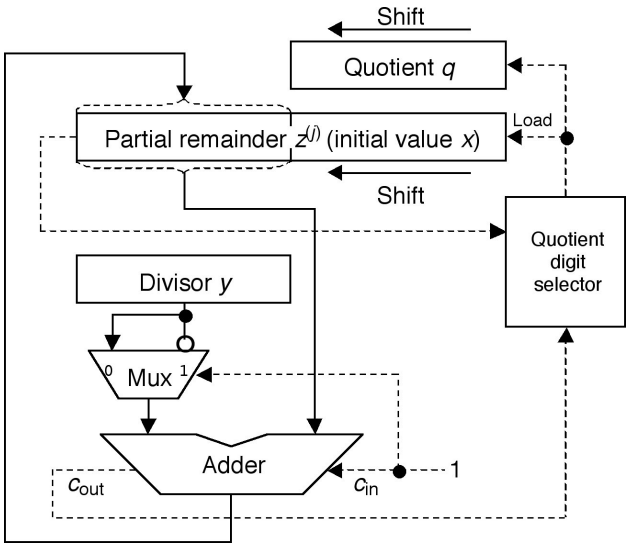


Figure 14 Hardware divider based on the shift-subtract algorithm.

XI. REAL NUMBERS

Integers in a prescribed range can be represented exactly for automatic processing, but most real numbers must be approximated within the machine's finite word width. Some real numbers can be represented as, or approximated by, $(k + l)$-bit fixed-point numbers (see Section V). A problem with fixed-point representations is that they are not very good for dealing with very large and extremely small numbers at the same time. Consider the two (8 + 8)-bit fixed-point numbers shown below:

$x = (0000\ 0000\ .\ 0000\ 1001)_{\text{two}}$ Small number

$y = (1001\ 0000\ .\ 0000\ 0000)_{\text{two}}$ Large number

The relative representation error due to truncation or rounding of digits beyond the −8th position is quite significant for x, but it is much less severe for y. On the other hand, neither y^2 nor y/x is representable in this number format.

The most common representation of numbers for computations dealing with values in a wide range is the *floating-point format*. Old computers used many different floating-point number formats, and some specialized machines still do, but for the most part, the IEEE floating-point standard format (ANSI/IEEE Standard 754-1985) has been adopted by the computer industry. We thus formulate our discussion of floating-point numbers and arithmetic exclusively in terms of this standard format. Other formats will differ in their parameters and representation details, but the basic trade-offs and algorithms remain the same.

A floating-point number has three components: sign ±, exponent e, and significand s, together representing the value $\pm 2^e s$. The *exponent* is a signed integer represented in biased format (a fixed bias is added to it to make it into an unsigned number). The *significand* is a fixed-point number in the range [1, 2). Because the binary representation of the significand always starts with "1.", this fixed 1 is hidden and only the fractional part of the significand is explicitly represented.

Figure 15 shows the details of short (32-bit) and long (64-bit) floating-point formats. The short format has adequate range and precision for most common applications (magnitudes ranging from 1.2×10^{-38} to 3.4×10^{38}). The long format is used for highly precise computations or those involving extreme variations in magnitude (from about 2.2×10^{-308} to 1.8×10^{308}). Of course in both of these formats, as explained thus far, zero has no proper representation.

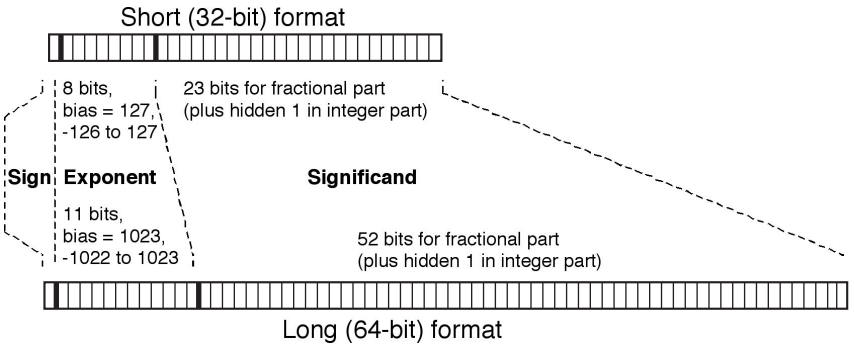


Figure 15 The ANSI/IEEE standard floating-point formats.

To remedy this problem, and to be able to represent certain other special values, the smallest and largest exponent codes (all 0s and all 1s in the biased exponent field) are not used for ordinary numbers. An all-0s word (0s in sign, exponent, and significand fields) represents +0; similarly, −0 and ±∞ have special representations, as does any nonsensical or indeterminate value, known as "not a number" (NaN). Certain other details of this standard are beyond the scope of this article.

When an arithmetic operation produces a result that is not exactly representable in the format being used, the result must be rounded to some representable value. The ANSI/IEEE standard prescribes four rounding options. The default rounding mode is "round to nearest even": choose the closest representable value and, in case of a tie, choose the value with its least-significant bit 0. There are also three *directed rounding modes:* "round toward +∞" (choose the next higher value), "round toward −∞" (choose the next lower value), and "round toward 0" (choose the closest value that is less than the value at hand in magnitude). With the round-to-nearest option, the maximum rounding error is 0.5 *ulp,* while with directed rounding schemes, the error can be up to 1 *ulp.*

XII. FLOATING-POINT ARITHMETIC

The floating-point operations of multiplication and division are not much different from their fixed-point counterparts. For multiplication, exponents of the two operands are added and their significands are multiplied:

$$(\pm 2^{e1} s1) \times (\pm 2^{e2} s2) = \pm 2^{e1+e2}(s1 \times s2)$$

Thus, a hardware floating-point multiplier consists of a significand multiplier and an exponent adder that together compute $2^e s$, with $e = e1 + e2$ and $s = s1 \times s2$. The result's sign is easily obtained from the signs of the operands. This is not all, however. With $s1$ and $s2$ in [1, 2), their product will lie in [1, 4) and may thus be outside the permitted range. If the product of the two significands is in [2, 4), dividing it by 2 via a 1-bit right shift will put it back into the desired range. When this *normalization* is needed, the exponent e must be incremented by 1 to compensate for the division of s by 2.

Floating-point division is similar and is governed by the equation:

$$(\pm 2^{e1} s1)/(\pm 2^{e2} s2) = \pm 2^{e1-e2}(s1/s2)$$

Again, given that the ratio $s1/s2$ of the two significands lies in (0.5, 2), normalization may be required for results that are less than 1. This normalization consists of multiplying the significand by 2 via a 1-bit left shift and decrementing the resulting exponent by 1 to compensate for the doubling of the significand. Of course, throughout the operation and ensuing adjustments, for both multiplication and division, the hardware must check for exceptions such as *overflow* (exponent too large) and *underflow* (exponent too small).

We next discuss floating-point addition. Subtraction can be converted to addition by changing the sign of the subtrahend. To perform addition, the exponents of the two operands must be equalized, if needed. Consider the addition of $\pm 2^{e1} s1$ and $\pm 2^{e2} s2$, with $e1 > e2$. By making both exponents equal to $e1$, the addition becomes:

$$(\pm 2^{e1} s1) + (\pm 2^{e1}(s2/2^{e1-e2})) = \pm 2^{e1}(s1 \pm s2/2^{e1-e2})$$

We thus see that $s2$ must be right-shifted by $e1 - e2$ bits before being added to $s1$. This *alignment shift* is also called *preshift* (so named to distinguish it from the *postshift* needed for normalizing a floating-point result). Figure 16 shows a complete example of

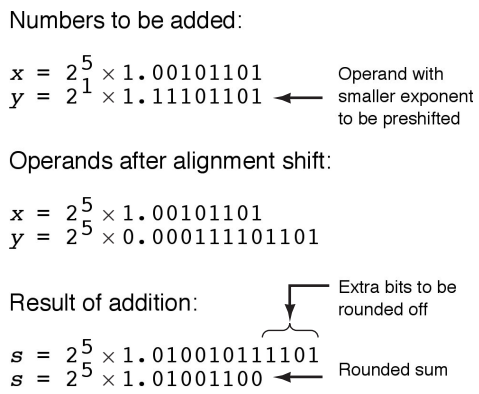


Figure 16 Alignment shift and rounding in floating-point addition.

floating-point addition, including preshift, addition of aligned significands, and final rounding. In this example, no postshift is needed because the result is already normalized. In general, though, the result may need a 1-bit right shift, when it is in [2, 4), or a multibit left shift when the addition of operands with different signs leads to *cancellation* or *loss of significance* and one or more leading 0s appear in the result.

A simplified block diagram for a hardware floating-point adder is shown in Fig. 17. It is seen that once the operands are unpacked, their exponents and significands are processed in two separate tracks whose functions are coordinated by the block labeled "Control & sign logic." For example, based on the difference of the two exponents, this unit decides which operand must be preshifted. To economize on hardware, usually only one preshifter is provided, say for the left operand of the adder. If the other operand needs to be preshifted, the operands are physically swapped. Also, in adding operands with unlike signs, the operand that is not preshifted is complemented. This time-saving strategy may lead to the computation of $y - x$ when in fact $x - y$ is the desired result. The control unit corrects this problem by forcing the complementation of the result if needed. Finally, normalization and rounding are preformed and the exponent is adjusted accordingly.

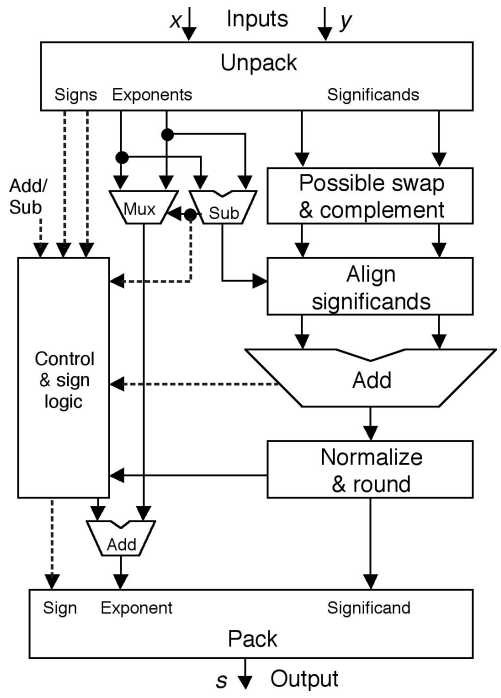


Figure 17 Simplified schematic of a floating-point adder.

XIII. FUNCTION EVALUATION

In many numeric calculations, there is a need to evaluate functions such as square-root, logarithm, sine, or tangent. One approach to function evaluation is the use of an *approximating function* that is easier to evaluate than the original function. Polynomial approximations, derived from Taylor-series and other expansions, allow function evaluation by means of addition, multiplication, and division. Here are a few examples:

$$\ln x = 2(z + z^3/3 + z^5/5 + z^7/7 + \ldots)$$

$$\text{where } z = (x - 1)/(x + 1)$$

$$e^x = 1 + x/1! + x^2/2! + x^3/3! + x^4/4! + \ldots$$

$$\cos x = 1 - x^2/2! + x^4/4! - x^6/6! + x^8/8! - \ldots$$

$$\tan^{-1} x = x - x^3/3 + x^5/5 - x^7/7 + x^9/9 - \ldots$$

A second approach is *convergence computation:* begin with a suitable approximation and proceed to refine the value with iterative evaluation. For example, if $q^{(0)}$ is an approximation to the square root of x, the following recurrence can be used to refine the value, using one addition, one division, and a 1-bit right shift per iteration:

$$q^{(i+1)} = 0.5(q^{(i)} + x/q^{(i)})$$

The initial approximation can be obtained via table lookup based on a few high-order bits of x or simply be taken to be a constant. As an example, suppose that we want to use this method to extract the square root of a floating-point number. To do this, we must halve the exponent and find the square root of the significand. If the exponent is odd, we can subtract 1 from it and double the significand, before applying

this method. As a result, the adjusted significand will be in [1, 4). We may thus take 1.5 as our initial approximation. The better the initial approximation, the fewer the number of iterations needed to achieve a certain level of accuracy. A special case of convergence computation is when each iteration leads to the determination of one digit of the result, beginning with the most significant digit. Such *digit recurrence* schemes are similar in nature to shift-subtract division discussed in Section X.

There are many other methods for function evaluation. As our third and final example, we consider a method based on lookup tables which is becoming increasingly popular, given that tables can be implemented efficiently and compactly in VLSI technology. Within a reasonably narrow interval $[x^{(i)}, x^{(i+1)})$, a function $f(x)$ can be approximated by the linear function $a + b(x - x^{(i)})$. This *interpolation scheme* leads to the hardware implementation depicted in Fig. 18. The range of x values is divided into 2^h intervals based on the h high-order bits of x, which define the value x_H. For each of these intervals $[x_H, x_H + 2^{-h})$, the corresponding a and b values of the approximating linear function $a + b(x - x_H) = a + bx_L$ are stored in two tables. Function evaluation with this method thus involves two table accesses, one multiplication, and one addition.

XIV. PRECISION AND ERRORS

Machine arithmetic is inexact for two different reasons. First, many numbers do not have exact binary representations within a finite word format. This is referred to as *representation error*. Second, even for values that are exactly representable, floating-point arithmetic produces inexact results. For example, the exactly computed product of two short floating-point numbers will have a 48-bit significand that must be rounded to fit in 23 bits (plus the hidden 1). This is characterized as *computation error*.

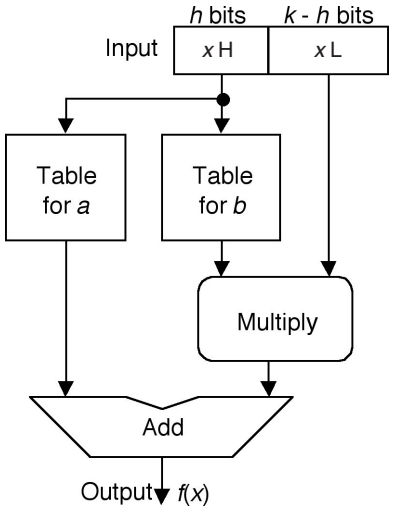


Figure 18 Function evaluation by table lookup and linear interpolation.

It is important for both the designers of arithmetic circuits and for the users of machine arithmetic to be mindful of these errors and to learn methods for estimating and controlling them. There are documented instances of arithmetic errors leading to disasters in computer-controlled critical systems. Even a small per-operation error of 0.5 *ulp,* when accumulated over many millions, perhaps billions, of operations needed in some applications, can lead to highly inaccurate or totally incorrect results.

We limit our discussion of errors, their sources, and countermeasures to a few examples from floating-point arithmetic. See D. Goldberg's article in the bibliography for more detailed discussion and other examples.

One way to avoid excessive error accumulation is to carry extra precision in the course of a computation. Even inexpensive calculators use extra digits that are invisible to the user but help ensure greater accuracy for the results. Without these *guard digits,* the computation of 1/3 will produce 0.333 333 333 3, assuming a 10-digit calculator. Multiplying this value by 3 will yield 0.999 999 999 9, instead of the expected 1. In a calculator with two guard digits, the value of 1/3 is evaluated and stored as 0.333 333 333 333, but still displayed as 0.333 333 333 3. If we now multiply the stored value by 3, and use rounding to derive the result to be displayed, the expected value 1 will be obtained.

Use of guard digits improves the accuracy of floating-point arithmetic but does not totally eliminate some incorrect and highly surprising results. For example, floating-point addition is not associative in that the algebraically equivalent computations $(x + y) + z$ and $x + (y + z)$ may yield very different results. Similarly, many other laws of algebra do not hold for floating-point arithmetic, causing difficulties in result predictability and certification. An optimizing compiler that switches the order of evaluation for the sake of computation speed-up may inadvertently change the result obtained.

One of the sources of difficulties is loss of precision that occurs when subtraction is performed with operands of comparable magnitudes. Such a subtraction produces a result that is close to 0, making the effect of previous roundings performed on the operands quite significant in relative terms. Such an event is referred to as *catastrophic cancellation*. For ex-

ample, when the algebraically correct equation

$$A = [s(s - a)(s - b)(s - c)]^{1/2}$$

with $s = (a + b + c)/2$, is used to calculate the area of a needlelike triangle (a triangle for which one side a is approximately equal to the sum $b + c$ of the other two sides), a large error can be produced due to the catastrophic cancellation in computing $s - a$. A user or programmer who is aware of this problem can use an alternate formula that is not prone to producing such large errors.

Because of the anomalies and surprises associated with floating-point arithmetic, there is some interest in *certifiable arithmetic*. An example is offered by *interval arithmetic* whereby each number is represented by a pair of values, a lower bound and an upper bound. We represent x by the interval $[x_l, x_u]$ if we are certain that $x_l \leq x \leq x_u$. Given interval representations of x and y, arithmetic operations can be defined in such a way as to ensure containment of the result in the interval that is produced as output. For example:

$$[x_l, x_u] + [y_l, y_u] = [x_l + y_l, x_u + y_u]$$

In this way, we always have a guaranteed error bound and will know when a result is too imprecise to be trusted.

The ultimate in result certification is *exact arithmetic,* which may be feasible in some applications through the use of *rational numbers* or other forms of exact representation. For example, if each value is represented by a sign and a pair of integers for the numerator and denominator, then numbers such as 1/3 will have exact representations and an expression such as $(1/3) \times 3$ will always produce the exact result. However, besides limited applicability, exact rational arithmetic also implies significant hardware and/or time overhead.

XV. SPEED AND RELIABILITY

We would, of course, like to design arithmetic circuits to be as fast as possible. The adder or multiplier must indeed be very fast if a machine is to execute one billion arithmetic operations per second. We now have this level of performance on some desktop computers, with 10^3 times greater performance in the largest available supercomputers, and 10^6 times more being worked on in research laboratories. Therefore, methods for speeding up arithmetic algorithms and their circuit implementations form an important part of the field of computer arithmetic.

With modern VLSI technologies, design for high speed is a challenging undertaking. The crossing of the gigahertz milestone in microprocessor clock rates signals the fact that many hardware operations are occurring with subnanosecond latencies. Because an electronic signal can travel only a few centimeters in one nanosecond, the roles of interconnects and package boundary crossings are becoming increasingly important. Given that clock distribution accounts for a significant portion of long wires and power dissipation on a modern VLSI chip, there is significant incentive to do away with the clocked or synchronous design style and adopt a fully asynchronous approach.

Speed is but one of several parameters that a designer must consider. In recent years, compactness and power economy have emerged as important attributes of an implementation. Design for compactness requires careful attention to implementation and packaging technologies and their various constraints. One important aspect to consider is the pin limitation at the chip and other levels of the hardware packaging hierarchy. Power economy is somewhat related to compactness, but it also depends on signal transition activity and circuit speed (faster circuit technologies use more power). Signal transition activity can be reduced via judicious choice of encoding schemes and careful algorithm design. Circuit speed can be reduced, while still keeping the same level of performance through various architectural schemes.

A particularly useful method of designing high-throughput arithmetic circuits without a need for ultrahigh-speed circuit elements is *pipelining*. We explain the use of pipelining and the various trade-offs involved in its implementation through an example. Consider the array multiplier of Fig. 12. The critical (longest) signal path through this circuit goes through four MA, one HA, and three FA blocks. This path begins at the upper left corner of the diagram, proceeds diagonally to the lower right, and then horizontally to the lower left corner. Assuming, for simplicity, that all three block types have the same unit-time latency, one multiplication can be performed every eight time units.

By cutting the design of Fig. 12 in two parts, right below the last row of MA blocks, and inserting temporary storage platforms (latches) to hold intermediate signal values at that point, we can double the throughput of our multiplier. Once the intermediate signals from one multiplication are stored in the latches, a second multiplication can begin in the upper part of the circuit, while the lower part completes the first multiplication. Of course, if we insert more latches, the throughput will increase even further. The improvement is not indefinite, though, because the insertion of more latches introduces increasing overheads in cost and time.

When results of an arithmetic unit are critical to the safety of humans or success of a mission, some

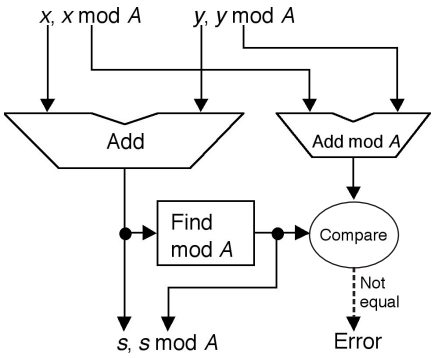


Figure 19 Arithmetic unit with residue checking.

form of checking must be performed. Design methods for *fault-tolerant arithmetic* form active research areas. Here, we just mention one simple method based on *residue checking*. For more information on this and other types of checking, see T. R. N. Rao's *Error Codes for Arithmetic Processors,* Academic Press, 1974.

Suppose that each value is represented by appending to it the residue modulo A, where A is a suitably chosen check constant. Then addition can be checked in the manner shown in Fig. 19. If the mod-A sum of the two check tags does not match the mod-A value of the computed sum s, then the presence of an error is signaled. Special attention must be paid to the design of the various components in Fig. 19 to ensure that matching of the two residues implies fault-free operation with very high probability.

XVI. UNCONVENTIONAL NUMBER SYSTEMS

Thus far, our discussion of computer arithmetic was based mostly on standard representations that are widely used in digital systems; namely, 2's-complement binary and ANSI/IEEE floating-point format. Other number representation systems are either invisible to the computer user (used to encode intermediate values for speeding up arithmetic operations) or are applied in the design of application-specific systems. In this section, we briefly review two examples of number systems in each of these categories. These are only meant to give the reader a sense that other options exist. See B. Parhami's textbook in the bibliography for more detailed discussion and other examples.

The *carry-save* (or *stored-carry*) representation for binary numbers is extensively used in the design of fast multipliers and other arithmetic circuits. A carry-save number is essentially a pair of binary numbers whose sum is the value being represented. Given such a carry-save value, it can be added to an ordinary binary number, yielding a carry-save result at very high speed, because the addition does not involve any carry propagation. Figure 20 shows the addition of a binary number x to a carry-save number composed of y and z, yielding the carry-save number composed of s and c. Comparing Fig. 20 to Fig. 6 reveals the origins of the name carry-save; note that here, unlike in Fig. 6, carries are not connected to the next FA block but are saved along with the sum bits to form a carry-save number.

Carry-save numbers can be viewed as radix-2 numbers that use the digit set {0, 1, 2}. From this observation, it is easy to generalize to other redundant representations such as *signed-digit numbers*. For example, radix-8 numbers with the digit set $[-5, 5]$ can be added without carry propagation chains; the carry only affects the next higher position and nothing else. Details are beyond the scope of this article.

The logarithmic number system (LNS) is an example of number representation for application-specific systems. In LNS, a value x is represented by its sign and the logarithm of its absolute value in some base. For example, if we use base-2 logarithms, with 4 whole and 4 fractional bits, the numbers 8 and 11 are represented as 0011.0000 and 0011.0111, respectively. The key advantage of LNS is that it allows us to multiply or divide numbers through addition or subtraction of their logarithms. For example, the product and ratio of 11 and 8 are found as follows:

$$\log_2 11 + \log_2 8 = (0011.0111)_{\text{two}} + (0011.0000)_{\text{two}} = (0110.0111)_{\text{two}}$$

$$\log_2 11 - \log_2 8 = (0011.0111)_{\text{two}} + (0011.0000)_{\text{two}} = (0000.0111)_{\text{two}}$$

Addition and subtraction, on the other hand, become somewhat more complicated but they are still manageable with help from lookup tables. Details are beyond the scope of this article. Suffice it to say that practical applications of this representation have thus

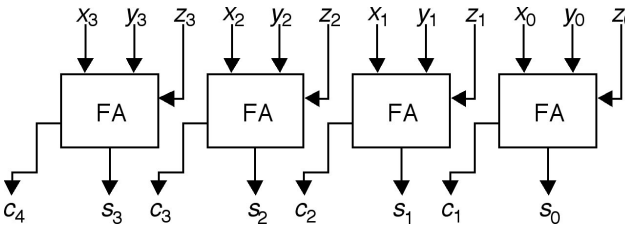


Figure 20 Three binary numbers reduced to two numbers by a carry-save adder.

far been limited to short word widths; however, available methods are improving and an LNS with range equivalent to the short ANSI/IEEE floating-point format has recently been implemented as part of a European microprocessor project.

The residue number system (RNS) has a long history dating back to the ancient Chinese. Use of this representation method in computer arithmetic was proposed in the late 1950s. Despite this long history, applications have remained limited due to the fact that RNS makes some key arithmetic operations, such as division and magnitude comparison, quite difficult. Its main advantages are simple addition and multiplication, thus making applications in which these two operations are predominant more suitable for RNS implementation. In RNS, each number is represented by a set of residues with respect to a set of pairwise relatively prime *moduli*. For example, given the moduli set {3, 5, 7}, the numbers 11 and 8 are represented by the set of their residues with respect to the moduli, thus leading to the codes (2, 1, 4) and (2, 3, 1), respectively. The number of distinct natural numbers that are represented is $3 \times 5 \times 7 = 105$. This set of values can be used to represent the natural numbers 0 to 104 or signed values -52 to $+52$.

In our example RNS with the moduli set {3, 5, 7}, adding or multiplying the numbers 11 and 8 is done by separately operating on the three residues, with each operation performed modulo the corresponding modulus. We thus get:

$$(2, 1, 4) + (2, 3, 1) = (4 \bmod 3, 4 \bmod 5, 5 \bmod 7) = (1, 4, 5)$$

$$(2, 1, 4) \times (2, 3, 1) = (4 \bmod 3, 3 \bmod 5, 4 \bmod 7) = (1, 3, 4)$$

Despite lack of widespread applications, both LNS and RNS remain important from a theoretical standpoint. Additionally, there are indications that with advances in arithmetic algorithms and VLSI technology, these two methods may find greater use in future. For information on applications of RNS arithmetic in signal processing, see *Residue Number System Arithmetic,* edited by M. A. Soderstrand, W. K. Jenkins, G. A. Jullien, and F. J. Taylor.

XVII. RESEARCH DIRECTIONS

Computer arithmetic has played a central role in the development of digital computers. A. W. Burkes, H. H. Goldstine, and J. von Neumann, in their now classic 1946 report entitled "Preliminary Discussion of the Logical Design of an Electronic Computing Instrument," set the stage for many ingenious schemes to perform fast arithmetic on early digital computers, at a time when even the simplest circuits where bulky and expensive. After more than half a century, research and development is still continuing unabated, for even though many of the theoretical underpinnings of the field are now well understood, new application challenges must be faced and old solution schemes must be adapted to emerging technological constraints and opportunities.

Examples of active research issues in computer arithmetic at the turn of the twenty-first century include reducing power consumption, efficient handling of low-precision multimedia data, subnanosecond operations, area-efficient implementations, configurable processing elements, function evaluation with no error other than that dictated by the mandatory rounding, certifiable arithmetic, compatibility/portability of numerical software, applying novel computational paradigms, and fundamental theoretical limits. New results in the field appear in *Proceedings of the Symposia on Computer Arithmetic,* currently held in odd-numbered years, and in archival technical journals such as *IEEE Transactions on Computers,* which occasionally devotes entire special issues to the topic. Web resources can be accessed via the author's home page at http://www.ece.ucsb.edu.

SEE ALSO THE FOLLOWING ARTICLES

Cybernetics • Evolutionary Algorithms • Information Theory

BIBLIOGRAPHY

Flynn, M. J., and Oberman, S. S. (2001). *Advanced Computer Arithmetic Design.* New York: Wiley.

Goldberg, D. (March 1991). What every computer scientist should know about floating-point arithmetic. *ACM Computing Surveys,* Vol. 23, No. 1, 5–48.

Knuth, D. E. (1997). *The art of computer programming—Vol. 2: Seminumerical algorithms,* 3rd edition. Reading, MA: Addison-Wesley.

Parhami, B. (2000). *Computer arithmetic: Algorithms and hardware designs.* New York: Oxford University Press.

Swartzlander, E. E., Jr. (1990). *Computer Arithmetic,* Vols. I & II, Los Alamitos, CA: IEEE Computer Society Press.

Object-Oriented Databases

Vladimir I. Zadorozhny
University of Pittsburgh

GLOSSARY

clustering A placement of objects that are accessed together on pages of disk storage to minimize the number of disk accesses.

complex object An object built up from other objects applying constructors, such as set, list and tuple.

encapsulation A principle of object organization under which an object has a visible external interface and a hidden internal implementation.

function materialization An approach to speed up the evaluation of path expressions that includes operation invocations by precomputing and storing the results of those operations.

index A data structure that speeds up the search for objects with certain attribute values.

intraobject synchronization An approach to object access synchronization that considers values of object attributes as shared data, whereas object methods are modeled as transactions accessing that data.

locking A method of object access synchronization where transactions that get access to database objects place locks on those objects.

multigranularity locking A method to lock a hierarchy of lockable database units (granules), such that placing a lock at a coarse granularity (e.g., object class) implicitly locks corresponding entities of finer granularities (an instance of the class).

object An abstract concept representing an entity of the real world in object-oriented database. Every object is associated with a unique object identifier, a set of attributes, and a set of operations on the object.

object access synchronization The organization of access from different transactions to shared database objects ensuring certain correctness criteria.

object class An object type together with an extent: the collection of objects of that type stored in object-oriented databases.

object-oriented database management system (OODBMS) A database management system that provides facility in storing, retrieving, and manipulating collections of objects according to an object-oriented paradigm.

object-oriented database query language A language that allows users to specify queries over collections of database objects without having the complexity of a general-purpose object-oriented programming language.

object-oriented database (OODB) standard A specification of requirements that should be satisfied by OODB applications to enable the application portability over different OODBMS products.

object type A specification of attributes and operations associated with objects having that type.

persistent object An object that continues to exist after the program that created it has terminated.

persistent object-oriented programming language An object-oriented programming language extended to deal with persistent objects.

I. INTRODUCTION

Relational database systems dominated the information management sector during the last two decades of the 20th century. They demonstrated high efficiency in traditional business applications. Meanwhile, it was obvious that pure relational databases did not satisfy the requirements of new application areas, such as computer-aided design (CAD) or multimedia. Relational databases represent data in a "flat" tabular format, as a set of fixed-length records with atomic fields. The fields have simple data types (e.g., strings, integers). However, CAD or multimedia databases need to store and handle much more complex data, such as composite modules of engineering design, images, and video data. The concept of the object-oriented database management system (OODBMS) was developed mainly as a response to these new application requirements, where data items are naturally represented as complex objects.

The basic data modeling approach of the OODBMS was borrowed from object-oriented programming languages (OOPLs). Extending OOPLs to deal with persistent objects, i.e., objects that continue to exist after the program that created them has terminated, was another research direction that strongly contributed to the concept of the object-oriented database (OODB). In fact, the OODBMS is often explained as an OOPL environment extended with the functionality of a database management system. The OODBMS provides facility in storing, retrieving, and manipulating data items as objects. An OODB is a collection of logically interrelated objects managed by an OODBMS. Since an OODBMS integrates database technology with the object-oriented paradigm, it is defined in terms of its mandatory *object-oriented* and *database* features. The object-oriented features come from OOPLs such as Smalltalk and C++ and include complex objects, object identity, object encapsulation, types or classes, type or class hierarchies and inheritance, overloading and late binding, extensibility, and computational completeness. The database features of OODBMS are persistency and secondary storage management, query language, concurrency, and recovery. They are not generally found in programming languages and are essential features of any database management system.

II. OBJECT-ORIENTED FEATURES OF OBJECT-ORIENTED DATABASE MANAGEMENT SYSTEMS

A. Objects

An OODBMS represents entities of the real world as objects. Every object is associated with a set of attributes and a set of methods implementing certain operations on the object. The values of the object's attributes form the *object state*. The methods act on the object's state and define the *object's behavior*. For example, attributes associated with a person object may include `name` and `address` of the string type, as well as `month_of_birth` and `year_of_birth` of the integer type. A possible state of the person object could be "John von Neumann, Washington D.C., 12, 1903." The behavior may include a method implementing the `get_age` operation to evaluate the age of a person from the current date and values of `month_of_birth` and `year_of_birth` attributes.

B. Object Identity

Every object in an OODB has an *identity,* which is independent of its state. It means that two different objects may have equal states, and that the values of an object's attributes can be changed without affecting its object identity. Object identity is implemented by assigning to each object a system-dependent, unique *object identifier* (OID).

C. Complex Objects

Complex objects can be built up from simpler ones by applying constructors. The typical constructors are *set, list,* and *tuple*. Constructors build collections of objects, ordered collections of objects and aggregated objects. In particular, objects (both simple and complex ones) can be values of other objects' attributes. Simple objects are objects of simple data types, such as strings or integers. Coming back to the person example, a person object could also have a `spouse` attribute, referring to another person object, as well as `parents` and `children` attributes, both referring to sets of person objects.

D. Encapsulation

The principle of encapsulation is that an object has a visible external *interface,* consisting of a set of operations, but hidden internal *implementation*. The implementation part includes both attributes representing the state of the object and methods implementing each operation. Under total encapsulation, applications can perform only operations specified in the interface of the object. That is not convenient for applications sending queries to an OODB. The query predicates are expressed in terms of object attributes.

However, extending the object interface with routing operations to access each attribute (i.e., `get_name`, `set_name` for the `name` attribute of the person object) would be impractical. Commonly, an OODBMS does not support total encapsulation and includes at least a part of the attribute specifications in the object interface. (See Kim, 1990 or Atkinson *et al.,* 1990 for further discussion of encapsulation issues in OODBMSs.)

E. Object Types and Classes

The common features of objects are summarized in object *types* and *classes.* The concepts of object type and class are related. The most appropriate in the context of an OODBMS is the interpretation of the object type as a specification of the object attributes and operations. This corresponds to the concept of an abstract data type. Attribute specification consists of the attribute name and type (e.g., `name:string` or `spouse:Person`). Operations are specified by their names, types of arguments, and types of results. The first argument of an operation is always the object itself. Thus, the `get_age` operation has the argument type `Person` and the integer result type. It will be specified as follows: `get_age:Person -> integer`. An object class is a type with an *extent:* a collection of objects of the given type stored in an OODB. Many OODBMSs support advanced object meta-models where types and classes themselves are represented as objects.

F. Type/Class Inheritance

Types and classes can form *inheritance hierarchies.* Inheritance avoids redundant storage of data that can be inferred and provides a means for code reusability. When two types *T1* and *T2* have common attributes and operations, those attributes and operations can be moved to a single type *T.* The type *T* is declared as a supertype of *T1* and *T2,* while types *T1* and *T2* are declared as subtypes of *T.* The subtypes automatically gain, or *inherit,* the common features from the supertype without having to respecify them. Subtypes can also extend the supertype with new attribute and operation specifications. For example, `Student` and `Instructor` types can inherit all attributes and operations of the `Person` type. In addition, a `Student` type can also specify an attribute `major` referring to `Department` objects, and an attribute `takes` referring to a set of `Course` objects, as well as `register` and `unregister` operations with a second argument of `Course` type (the first argument is the object itself). The type `Instructor` can extend `Person` with the attributes `works_in` of `Department` type, `teaches` referring to a set of `Course` objects, and two operations: `add_new_course` and `drop_course` with the second argument of `Course` type. Figure 1 depicts this type inheritance hierarchy.

The type inheritance hierarchy is isomorphic to the class/subclass hierarchy, where a class extent is a subset of the extent of the corresponding superclass.

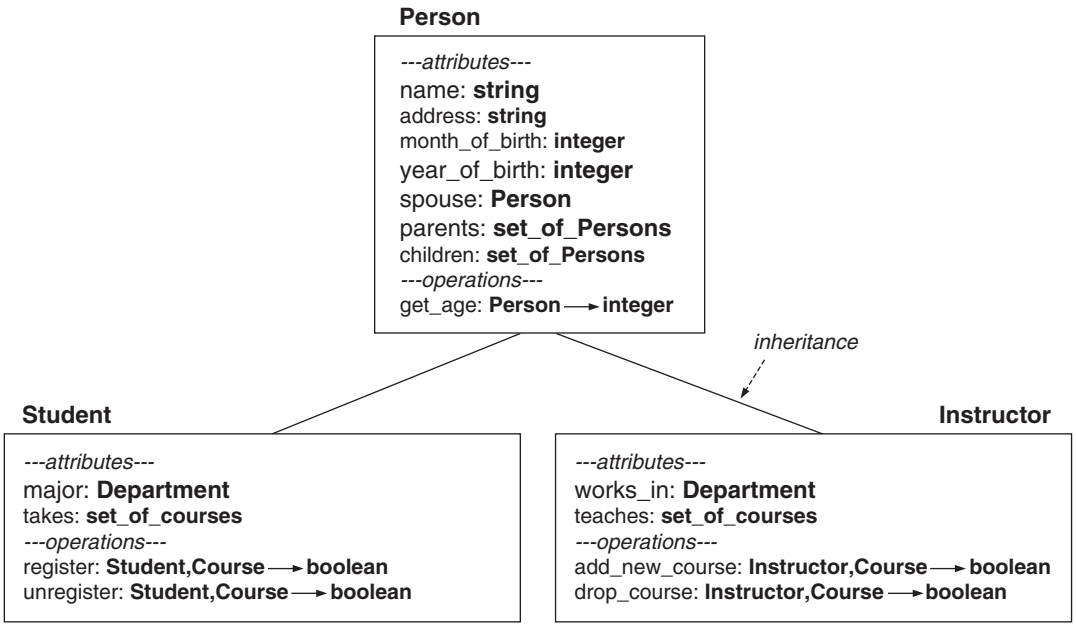


Figure 1 Example of a type inheritance hierarchy.

A type/class can inherit from multiple supertypes/superclasses, but in this case the OODBMS must provide a way to resolve conflicts with same-named attributes or operations that inherited from different supertypes/superclasses.

G. Overloading and Late Binding

A subtype can *override* an attribute or operation of its supertype by defining its own local attribute or operation with the same name. This is called *overloading* the name. For example, the `Student` type can override the `address` attribute so that it refers to university address instead of home address. The *late binding* mechanism allows the system to select appropriate bindings for the overloaded names at run time by examining the most specific type of the object under consideration.

H. Extensibility and Computational Completeness

OODBMS *extensibility* makes it possible to add *user-defined types/classes* to a set of predefined system types/classes, such as `Date or Time`. In particular, `Person, Student,` and `Instructor` are examples of user-defined types. User-defined types or classes have the same status and usage as predefined ones.

Another feature, which is often considered as a characteristic of an OODBMS, is the computational completeness of its data manipulation language (DML). This means that the OODBMS DML should be able to express any computable function. It should be noted that SQL, supporting basic data retrieval functionality, is not computationally complete.

V. DATABASE FEATURES OF OBJECT-ORIENTED DATABASE MANAGEMENT SYSTEMS

A. Persistency and Secondary Storage Management

The data manipulated in an OODBMS should be *persistent*. The database objects must survive the execution of an application process that creates or updates them, so that subsequent applications can use those objects as well. Persistent objects are saved to a persistent storage media (i.e., disk) and loaded back to main memory for application processing. An OODBMS must provide *efficient mechanisms for persistent storage management* to support very large databases. Allocation of persistent storage and transfer of objects to and from main memory should be transparent to the user application.

B. Query Language

An OODBMS should allow the user to specify queries to the database in a high-level and application-independent query language. The query optimizer and execution engine of the OODBMS should ensure efficient query execution. Object queries and query languages are considered in Section IV.

C. Concurrency and Recovery

A database is in a consistent state if none of the database's integrity constraints are violated. An OODBMS should support *concurrent interactions* of multiple users with a database, preserving database consistency. Commonly, such interactions are organized in the form of multiple user *transactions* over a shared database. Transactions are executions of database operation sequences that satisfy the properties of *atomicity, consistency, isolation,* and *durability* (ACID properties). Atomicity assumes that either all or none of the transaction's operations must be completed. Consistency means that the transaction always leaves the database in a consistent state, although during the execution of a transaction the database can be temporarily in an intermediate inconsistent state. Isolation requires that each transaction not reveal intermediate results of its execution to other transactions. So, if the transaction is interrupted by a system failure (e.g., disk, processor, or software failure), the OODBMS should be able to *recover* to a consistent database state. This may include erasing the results produced by partially executed transactions. The durability property ensures that a system failure cannot erase the results of a transaction that has been committed. We elaborate on transaction processing in OODBMSs in Section V.

IV. QUERY LANGUAGES FOR OBJECT-ORIENTED DATABASES

Different kinds of languages can be used to specify queries to object databases. One way is to add object-oriented features to relational query languages such as SQL. Such object-oriented extensions of SQL are used in *object-oriented relational database management systems*

(OORDBMS). As for relational database systems, ORDBMSs represent data in the form of relations and tuples. However, in the object-relational database, attributes of tuples may have complex object types. OODBMSs deal with objects in a more uniform way: collections of any objects (not just tuples) are considered.

A. Persistent Object-Oriented Programming Language

The most straightforward way to provide a query language for an OODBMS is to extend an OOPL to deal with persistent data. In such a *persistent OOPL* (POOPL) an object continues to exist even after the program that created it has terminated. There are several ways to implement object persistence. In the case of *persistence by class,* a whole class is declared persistent. A more flexible approach is to associate persistency with individual objects. An object can be either declared as persistent at the time of its creation *(persistence by creation)* or marked as persistent after it was created *(persistence by marking).* Under the *persistence by reference* approach, objects are persistent if they are referred to directly, or indirectly, from other persistent objects. In this case some objects should be explicitly declared as root persistent objects. Persistence by reference simplifies creating complex persistent data structures. However, the implementation of persistence by reference makes it more difficult for the database system to detect the persistent object automatically. A POOPL allows programmers to access and manipulate a database without using a special database language such as SQL. In past years, considerable efforts have been made to develop persistent versions of the most popular OOPLs such as C++ or Smalltalk. The following example specifies a class of persons in the extended C++ that has been proposed by the Object Database Management Group (ODMG).

```
class Person: public
  PersistentObject {
public:
     String name;
     short date_of_birth;
     String address;
     Ref<Person> spouse inverse
       Person::spouse;
     Set<Ref<Person>> parents inverse
       Person::children;
     Set<Ref<Person>> children inverse
       Person::parents;
     long get_age();
}
```

The ODMG defines a class library to support object persistence. The `Person` class is a subclass of a library class `Persistent_Object,` which makes possible for the `Person`'s objects to be persistent. Type `Ref<Person>` denotes a persistent pointer to an object of the `Person` class. The `inverse` keyword specifies a relationship in the opposite direction, e.g., each person object in the set of children must contain pointers to their parents. Programmers can create and manipulate objects of the `Person` class using common C++ syntax.

One of the main disadvantages of this approach is that imperative and complex POOPLs do not support declarative queries. The power of a general-purpose OOPL also makes it harder to check programming errors and to avoid violations of database integrity constraints. As a result, both query specification and query processing become more difficult.

B. Object-Oriented Database Language

Another approach is to develop a special database object-oriented query language. This language may allow users to specify queries over collections of objects (not just relations of tuples) without having the complexity of a general-purpose OOPL. The ODMG proposed such a language called *object query language* (OQL). OQL has an SQL-like syntax, but instead of relations it deals with classes of objects. The ODMG also specified an object definition language (ODL), which is used to define object classes. For example, the ODL specification that defines a class of persons could be as follows:

```
class Person
(extent persons)
attribute string name;
attribute short date_of_birth;
attribute string address;
relationship Person spouse inverse
Person::spouse;
relationship set<Person> parents in-
verse Person::children;
relationship set<Person> children
inverse Person::parents;
long get_age();
}
```

The complex attributes referring to other objects or to collections of objects are specified as *relationships.* A relationship is associated with a corresponding inverse relationship.

The following OQL query finds married couples such that at least one spouse is more then 30 years old:

```
select P.name, S.name
from Person P, P.spouse as S
where S.get_age() > 30
```

An important issue is what to consider as the result of a query to an object database. Intuitively, it should be a collection of objects satisfying the query condition. If the query selects objects from only one class (e.g., `Person`), the result should include instances of this class. It may also include instances of any subclass of the given class (i.e., `Student` and `Instructor` for the `Person` class). Thus, the query result will be a heterogeneous collection of objects belonging to different classes. Manipulating such collections may involve certain technical difficulties. To avoid this problem, an OODBMS can return a collection of object identifiers as a query result and have the application to request values of object attributes and invoke object operations explicitly. If the query requires a projection over some object attributes (e.g., `select P.name from Person P`), the result will also include values of the projected attributes of the objects satisfying the query condition.

Now queries on several classes are considered. In relational databases, where data items are stored in a set of flat relations, this corresponds to queries over several relations. The relations are combined on the basis of the values of their common attributes using a join operation. In an OODB, object attributes specified in classes can refer to other object classes that define an *implicit join* of two classes over the attributes. For example, the `Instructor` class defines a `work_in` attribute referring to `Department` class, and any selection on `Instructor` will implicitly involve joining of `Instructor` and `Department` classes with the `work_in` attribute of `Instructor` objects being equal to object identifiers of corresponding `Department` objects. Such an implicit join statically determines the classes to be joined, and it is quite restrictive as far as queries over multiple classes are concerned. The implicit join does not allow the user to formulate queries over the classes that are not related via class-composition structure. An *explicit join* allows classes on any *compatible* pair of attributes to join. Two attributes are compatible if they refer either to the same class or to classes related via class inheritance hierarchy. For example, `Student` and `Instructor` classes can explicitly join on `Student's major` attribute and `Instructor's works_in` attribute (both referring to the class `Department`). The result of the explicit join is a set of objects formed by concatenating objects from both classes. The concatenated objects can be assigned fresh object identifiers and can be stored in a new class. (See Kemper and Moerkotte, 1994 for more details on explicit join of classes.)

V. TRANSACTIONS IN OBJECT-ORIENTED DATABASE MANAGEMENT SYSTEMS

The OODBMS transaction manager is responsible for synchronizing access from different transactions to shared database objects, ensuring certain correctness criteria such as the *serializability* of transactions. Serializability requires that the effect of the concurrent execution of several transactions be the same as that of executing them sequentially. We consider several factors making transaction management in OODBMS harder than in conventional database systems:

- Synchronization of access to a complex object also requires access to its components, which themselves can be complex objects.
- Type/class inheritance hierarchy should be taken into account: transactions accessing instances of a particular class may also access instances of its subclasses.
- Method executions should be considered: a transaction can invoke object operations that invoke operations on other objects, and so on. This may result in chain of method executions that requires access to many other objects.

The concurrency control algorithms used by transaction managers are commonly based on *locking protocols:* a transaction places a *shared* or *exclusive lock* on an object before accessing it. The kind of lock required depends on the access mode. Reading an object needs a shared lock, while modifying an object requires an exclusive lock. Lock compatibility rules are used to avoid *read-write, write-read,* and *write-write* conflicts. To minimize the number of locks while accessing the database, a *multigranularity locking protocol* has been proposed. Multigranularity locking deals with a hierarchy of lockable database units (granules) such as database, object class, and individual object. A transaction that puts a lock at a coarse granularity (e.g., object class) implicitly locks corresponding entities of finer granularities (all instances of the class).

To synchronize access to complex objects, it is possible to lock a complex object and all the classes of its component objects using multigranularity locking. Whenever a transaction accesses a class, the class itself and all its subclasses are appropriately locked. This approach is quite restrictive, since it may result in many

locks on objects that are not related to the transaction, particularly when the accessed class is close to the root of the inheritance hierarchy. Another way considers a component object as a lockable unit; this requires some extension of the basic multigranularity locking protocol. The inheritance hierarchy can also be managed on the basis of multigranularity locking. Multiple inheritance results in another problem with basic multigranularity locking in OODBMSs. Two transactions can place conflicting locks (e.g., *read* and *write*) on different superclasses of the same subclass that may thus be implicitly locked in incompatible modes. Several extensions to the basic multigranularity locking have been proposed to avoid the above problems.

Another approach to object access synchronization is based on *intraobject synchronization protocols*. Under this approach the object state is considered as shared data, whereas methods are modeled as transactions getting access to the data. It does not require the same synchronization mechanism for every object. Objects themselves are responsible for the intraobject synchronization. The transaction manager deals with interobject synchronization.

The need to access complex objects with nested structure was one of the reasons for the development of *nested transaction models*. A nested transaction includes other transactions as its subtransactions. The subtransactions themselves can be nested also. Complex OODBMS applications may also require long duration transactions over the database (e.g., activities of complex enterprises). Such transactions are called *workflows*. The execution of workflows can take several days, and the ACID properties are not very appropriate for them. Thus, a workflow may violate ACID properties and permit its partial results to be visible to other activities.

VI. PHYSICAL DESIGN OF OBJECT-ORIENTED DATABASES

The performance of any database management system (DBMS) is strongly influenced by physical database design. This section considers some common issues of physical database design, such as clustering and indexing in the context of the OODB. Techniques that are specific to OODBMSs, including method materialization and pointer swizzling, are elaborated.

A. Clustering

A disk is a slow storage device. To be processed, an object is fetched from disk to main memory. An OODBMS stores data on disk in *page* units. The whole page (or set of pages) where the object resides is transferred into a region of main memory called the *buffer pool*. The size of the buffer pool is limited. If too many pages are requested, some of them must be transferred back to disk and fetched to main memory again as soon as they are needed. A *page fault* occurs when the page with the object needed by a database application is not in the buffer pool and disk access is required.

Because input/output operations on disk are costly, the number of page faults should be minimized by the appropriate placement of objects on pages. Finding such placement is called *clustering*. Intuitively, those objects that are accessed together should be placed on the same page. To find such clusters of related objects, consider both objects structures and application access patterns. For example, it makes sense to put all objects aggregated in the same complex object in one cluster. In such a way the number of pages and, correspondingly, the number of disk accesses required to retrieve the complex object can be reduced. Similarly, objects related via an inheritance hierarchy can be placed into a cluster.

To define the access patterns of applications to objects, the application semantics should be analyzed. For example, if the application computes the average salary of employees within a company, it makes sense to include company and employee objects in one cluster. It is also possible to gather statistics on the frequency of accesses to objects in the application at run time. Obviously, there is no unique best clustering for all applications. Periodical *reclustering* (i.e., transferring objects from their initially assigned page to other pages) typically results in performance improvement.

B. Indexing

A database index works in a similar way as an index for a book. To find a book page where a certain concept is described, look in the index where the corresponding index record has the page number. In an OODB, indices are used to speed up the search for objects with certain attribute values. Unfortunately, conventional indexing techniques of relational databases cannot be applied directly to an OODB. Recall that relational databases store only records with atomic attributes, but object attributes may refer to other complex objects. There should be a special way to deal with values of such complex attributes in the index.

Commonly, values of a complex attribute are represented by means of *path expressions*. A path expression is a sequence of attribute names that are parts of

the object structure. For example, the path expression `p.spouse.name`, where `p` is a `Person` object, refers to the name attribute in the `spouse` component of a person. In general, the computation of path expressions requires access to multiple objects and may be quite costly. That may be critical when path expressions are used in query specifications (e.g., find all persons `p`, such that `p.spouse.name = "John Lennon"`). An index from spouse names to persons would significantly speed up the computation of the path expression. The index could be maintained either only for end points of the path (i.e., persons and names of their spouses), for some path segments (i.e., persons and their spouses), or for the whole path (i.e., persons, their spouses, and spouses' names). A general technique to represent indices for path expressions consists of using *access support relations* (ACR). An ACR is a table with a field for each path component. In the case when a path step corresponds to a collection-valued attribute, multiple tuples with the same value of the path component are included in the ACR. For example, the path `p.children` would result in a tuple for each child of the person.

Care should be taken to maintain the path indices. Changes to objects referred to by any path segment can affect the index, so all path components (not just ends of paths) must be considered when updating the index. For example, if we maintain an index for the path `company.manager.spouse.name`, where `manager` refers to a person object within a company object, changes to `manager` would result in the modification of `spouse` and `name` components.

C. Function Materialization

The maintenance of indexes in an OODBMS is complicated by the presence of methods. A component of a path expression can be invocation of an operation. For example, if the `Company` type defines `get_manager` operation that returns a manager object for a given department, then the following path expression will specify a name of sales manager: `company.get_manager("Sales").name`. One approach to speed up the evaluation of such expressions is to precompute and store the results of such operations and to maintain an index on those results. This is called *operation* or *function materialization*. If the operation has arguments, the arguments should be stored together with the result. The results of a materialized operation can be logically represented as a *generalized materialization relation* (GMR). The attributes of the GMR store references to the argument objects, references to result objects, and some validation information indicating whether or not the stored results are currently valid (Fig. 2). Since methods implementing the operations typically access arbitrarily many objects, any change of those objects (e.g., company changed its sales manager) can invalidate the materialized result and thus affect the index. In this case the system can either immediately re-compute the invalidated operation result *(immediate rematerialization)* or only set corresponding validation attributes of the GMR so that the result of the invalidated operation is recomputed at some other time before it is needed *(lazy rematerialization)*. The detection of invalidated

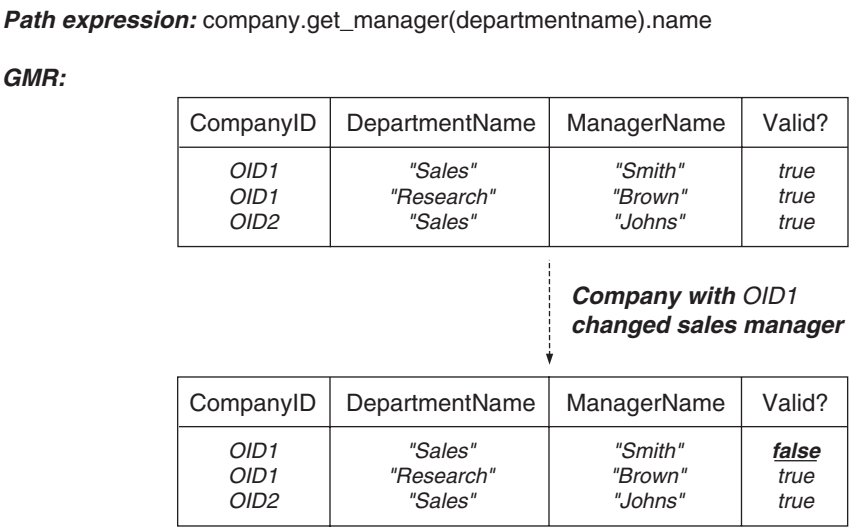


Figure 2 A generalized materialization relation.

operations induces some overhead on object modification.

D. Pointer Swizzling

As we know from Section II.B, OODBMSs reference objects use unique OIDs. As we also remember from Section VI.A, during the application execution a persistent object may be either in main memory or on disk. OIDs allow the system to localize the object. Typically, an OODBMS maintains a table mapping OIDs into current addresses of memory-resident objects. Complex objects refer to their component objects via OIDs, and if the component object is in memory, access to it requires a look up in the table of memory-resident objects. For computation-intensive applications this may result in significant runtime overhead. Pointer swizzling converts memory-resident objects, replacing referenced OIDs of the memory-resident component objects by their main memory addresses. This makes references to memory-resident objects very fast. However, any time an object is moved back to disk, in-memory references to it should be replaced by its OID. Figure 3 depicts the effect of pointer swizzling for `student` and `department` in-memory objects.

VII. STANDARDS FOR OBJECT-ORIENTED DATABASES

Any technology works better if there is an agreement on corresponding standards. Standards make it easier to use the technology and strongly contribute to technology transfer from research labs to industry and to its final success. For example, because of a standard for electrical systems, different kinds of electrical equipment can be used in a plug-and-play manner in many countries all over the world. When you travel to

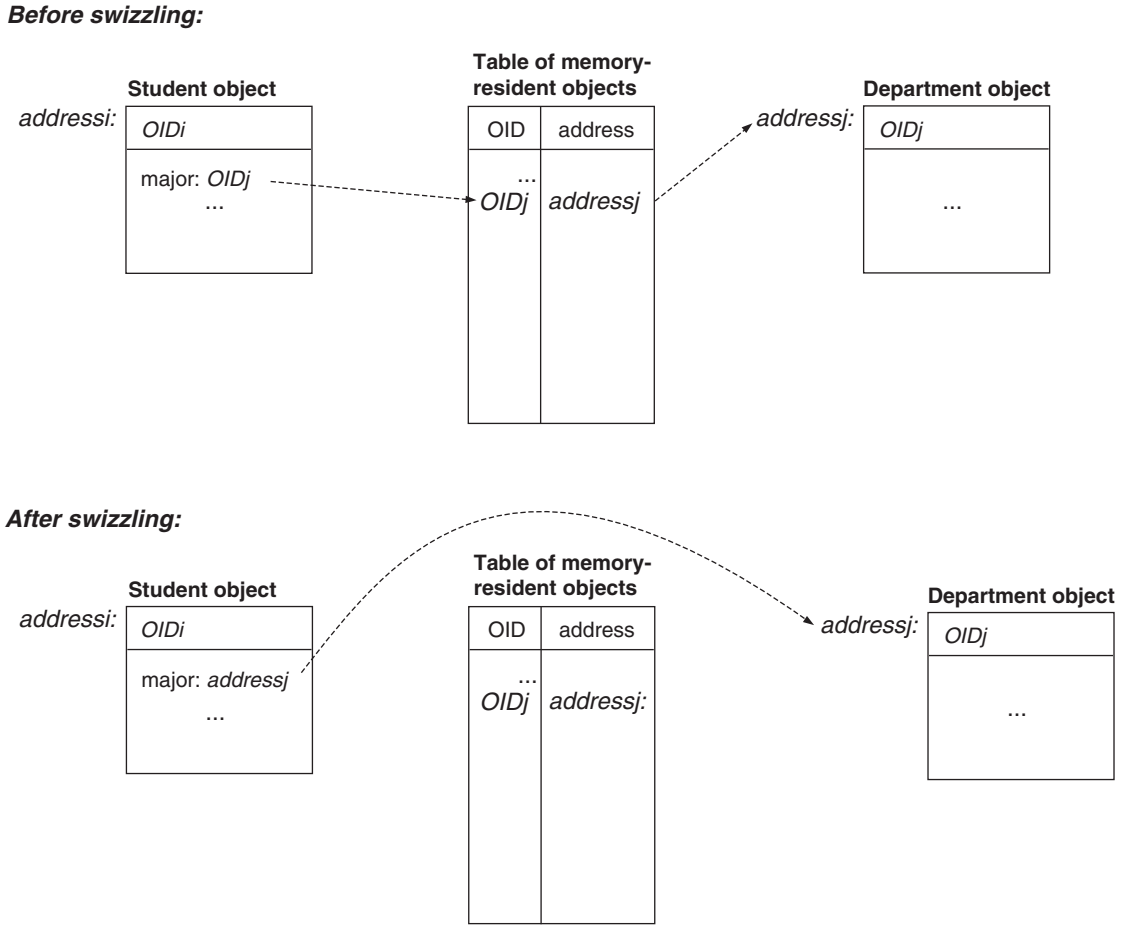


Figure 3 Pointer swizzling for student and department objects.

another country, it is quite disappointing to find out that the local electrical system is not compatible, say, with your shaver.

In OODBMSs, as well as in any other software technology, standards allow us to port applications across different OODBMS products. At this time, there is no single OODBMS standard accepted by all DBMS vendors, although several groups are working in this area. We consider three well-known standards that affect OODBMSs:

- Common Object Services, proposed by the Object Management Group (OMG)
- ODMG 3.0, proposed by the ODMG
- SQL3, proposed by the ANSI X3H2 group

A. Object Management Group Common Object Services

The OMG is a consortium of leading companies in the area of information technology, which is moving forward in establishing a standard architecture for distributed object systems—*common object request broker architecture* (CORBA). The OMG specifies a core object model that includes a small number of basic concepts such as objects, operations, types, and subtyping. A specific application domain can provide extensions of the core object model. The OMG also specifies a set of general-purpose common object services, which are fundamental for developing CORBA-based applications. It includes persistent state service (PSS) (formerly, persistent object service) which provides a uniform interface to manipulate persistent objects. The main task of the OMG PSS is storing/restoring objects on different *persistent storages*. The implementation details of the storage media (i.e., data format) should not influence the client application; that is, the implementation of the storage should be transparent for the client. PSS can also cooperate with other OMG services such as transaction, concurrency, etc.

B. Object Database Management Group

The ODMG includes OODBMS vendors whose primary goal is to achieve source code portability for OODB applications across database systems. Since ODMG members represent a major part of the current ODBMS industry, they want their proposal to be a de facto standard.

The first ODMG specification was released in 1993. Since that time several enhancements to the initial release, commonly known as ODMG-93, have been provided. The major part of ODMG describes the object model; the object specification language; the object query language; and bindings of ODMG implementations to the C++, Smalltalk, and Java programming languages. The ODMG standard was developed under the influence of earlier work of the OMG on the CORBA. The OMG object model has been used as a basis for the ODMG model. In addition, the ODMG model includes database features, such as transactions, keys, and class extents. The OQL is also based on the query portion of SQL-92, an SQL standard developed by the ANSI X3H2 group.

C. SQL3

SQL3 does not use any of the standards from the OMG or from object-oriented programming. SQL3 is founded on previous SQL standards, such as SQL-92, and supports backward compatibility to previous SQL versions. It also adds features to the SQL specification that support object-oriented data management. Among those features are support for abstract data types and language extensions that make SQL computationally complete. Objects in SQL3 are stored in attributes of particular relations. As a result, objects can be accessed only through those relations, not directly. That makes SQL3 more related to object-relational databases.

VIII. PROSPECT OF OBJECT-ORIENTED DATABASE SYSTEMS

The evolution of database technology is characterized by a number of paradigm shifts. The most notable of these is associated with the emergence of relational databases. E.F. Codd devised the idea of relational databases in 1969. Relational databases were adopted in the late 1970s and early 1980s and started to dominate the market of database systems.

The concept of OODBs was proposed in the mid-1980s. New OODB products emerged quite soon after that. Relational database vendors responded to the OODB products by creating hybrid object-relational database systems that incorporated relational and object-oriented features. An important advantage of the object-relational approach is that it is based on the well-established relational paradigm. In

addition to that, many users still find relational databases to be adequate for their needs. As a result, at the end of the 1990s sales of the OODB products were relatively low compared to the size of the database market. The size of most object-oriented database vendors was also quite small.

It is too early to come to a definite conclusion about the prospect of OODB systems. Comparing the evolution of relational databases with the progress in OODB systems, it is reasonable to conclude that OODB technology still has some time to mature. It may be a matter of time until the object-oriented vendors take a more powerful position in the database market. Among the advantages of the object-oriented approach that its supporters often mention is the fact that an OODB typically outperforms a relational database for common queries. Another benefit of OODBs is a reduction of application development time when the application is implemented in an OOPL such as Java or C++. This is because an OODB stores objects directly, and there is no need to write code for translating between objects and relational tables. To summarize, it is still not clear whether OODBs will eventually dominate in the market of database systems.

SEE ALSO THE FOLLOWING ARTICLES

Data Modeling: Object-Oriented Data Model • Distributed Databases • Hyper-Media Databases • Object-Oriented Programming • Relational Database Systems • Structured Query Language (SQL) • Temporal Data Model and Query Language Concepts

BIBLIOGRAPHY

Atkinson, M., Bancilhon, F., DeWitt, D., Dittrich, K., Maier, D., and Zdonik, S. (1990). *The object oriented database manifesto.* Amsterdam/New York: Elsevier.

Bertino, E., and Martino, L. (1993). *Object-oriented database systems. Concepts and architectures.* Reading, MA: Addison-Wesley.

Cattell, R., and Barry, D., Eds. (2000). *The object data standard: ODMG 3.0.* San Mateo, CA: Morgan Kaufmann.

Date, C., and Darwen, H. (2000). *Foundation for the future database systems. The third manifesto.* Reading, MA: Addison-Wesley.

Dogac, A., Ozsu, T., Biliris, A., and Sellis, T., Eds. (1992). *Advances in object-oriented database systems.* Berlin/New York: Springer-Verlag.

Kemper, A., and Moerkotte, G. (1994). *Object-oriented database management.* New York: Prentice Hall.

Kim, W., (1990). *Introduction to object-oriented databases.* Cambridge, MA: MIT Press.

Zdonik, S., and Maier, D., Eds. (1990). *Readings in object-oriented database systems.* San Mateo, CA: Morgan Kaufmann.

Object-Oriented Programming

Raymond Greenlaw and Y. Daniel Liang
Armstrong Atlantic State University

GLOSSARY

abstract class When you are designing classes, a superclass should contain common features that are shared by subclasses. Sometimes the superclass is so abstract that it cannot have any specific instances. These classes are called abstract classes.

abstraction A technique in software development that hides detailed implementation. Class abstraction hides the implementation of the class from the client.

abstract method A method signature without implementation. Its implementation is provided by its subclasses.

accessor method The get and set methods for retrieving and setting private fields in an object.

aggregation A special form of association that represents an ownership relationship between two classes.

association A general binary relationship that describes an activity between two classes.

class An encapsulated collection of data and methods that operate on data. A class may be instantiated to create an object that is an instance of the class.

class method A method that can be invoked without creating an instance of the class. To define class methods, put the modifier static in the method declaration.

class variable A data member declared using the static modifier. A class variable is shared by all instances of that class. Class variables are used to communicate between different objects of the same class and to handle global states among these objects.

class's contract Refers to the collection of methods and fields that are accessible from outside a class, together with the description of how these members are expected to behave.

composition A form of relationship that represents exclusive ownership of the class by the aggregated class.

constructor A special method for initializing objects when creating objects using the new operator. The constructor has exactly the same name as its defining class. Constructors can be overloaded, making it easier to construct objects with different initial data values.

dynamic binding An object of a subclass can be used by any code designed to work with an object of its superclass. For example, if a method expects a parameter of the `GeometricObject` type, you can invoke it with a `Circle` object. A `Circle` object can be used as both a `Circle` object and a `GeometricObject` object. This feature is known as polymorphism (from a Greek word meaning "many forms"). A method may be defined in a superclass, but is overridden in a subclass. Which implementation of the method is used on a particular call will be determined dynamically by the JVM at runtime. This capability is known as *dynamic binding*.

encapsulation Combining of methods and data into a single data structure. In Java, this is known as a class.

inheritance In object-oriented programming, the use of the extends keyword to derive new classes from existing classes.

instance An object of a class.

instance method A nonstatic method in a class. Instance methods belong to instances and can only be invoked by them.
instance variable A nonstatic data member of a class. An instance variable belongs to an instance of the class.
instantiation The process of creating an object of a class.
interface A class-like construct, used to model weak inheritance relationship. In Java, it can be used to achieve multiple inheritance.
object-oriented programming (OOP) An approach to programming that involves organizing objects and their behavior into classes of reusable components.
private A modifier for members of a class. A private member can only be referenced inside the class.
public A modifier for classes, data, and methods that can be accessed by all programs.
sequence diagram A UML diagram that describes interactions among objects by depicting the time-ordering of method invocations.
statechart A UML diagram that describes the flow of control of an object.
static method Same as class method.
static variable Same as class variable.
subclass A class that inherits from or extends a class.
superclass A class inherited from a subclass.
unified modeling language (UML) A graphical notation for describing classes and their relationships.

I. INTRODUCTION

Computer programs are instructions to computers. You tell a computer what to do through a program. Without programs, a general-purpose computer such as a personal computer (PC) is an empty machine. Computers do not understand human languages very well. So, you need to use computer languages to communicate with computers. There are over one hundred different programming languages. A number of the most popular languages are:

- COBOL (COmmon Business Oriented Language)
- FORTRAN (FORmula TRANslation)
- BASIC (Beginner All-purpose Symbolic Instructional Code)
- Pascal (Named for Blaise Pascal)
- Ada (Named for Ada Lovelace)
- C (Whose developer designed B first)
- Visual Basic (BASIC-like visual language developed by Microsoft)
- Delphi (Pascal-like visual language developed by Borland)
- C++ (object-oriented language, based on C)
- Java (object-oriented language based on C++)

Each language mentioned here was designed with a specific purpose in mind. COBOL was designed for business applications and now is used primarily for business data processing. FORTRAN was designed for mathematical computations and is used mainly for numeric computations. BASIC, as its name suggests, was designed to be easy to learn and use. Ada was developed at the direction of the Department of Defense, and is mainly used in defense projects. C is popular among system software developers for projects like writing compilers and operating systems. Visual Basic and Delphi are for rapid application development. C++ is the C language with object-oriented features. Java is partially modeled on C++, but greatly simplified and improved.

All of these languages except C++ and Java are known as *procedural programming language*. Software systems developed using procedural programming language are based on the paradigm of procedures. In procedural programming, data and operations on the data are separated; this methodology requires sending data to procedures and functions. Object-oriented programming (OOP) places data and the operations that pertain to the data within a single entity called an *object*. This approach solves many of the problems inherent in procedural programming. The OOP approach organizes programs in a way that mirrors the real world, in which all objects are associated with both attributes and activities. Object-oriented programming is centered on creating objects, manipulating objects, and making objects work together. An OOP can be viewed as a collection of cooperating objects.

The benefits of OOP are summarized by Harmon and Booch. Here we mention a couple of the key advantages of OOP. The central issue in software development is how to reuse code. Rather than reinvent the wheel for each complex program, a programmer would like to be able to utilize as much of an existing code base as possible. Object-oriented programming provides great flexibility, modularity, clarity, and reusability through method abstraction, class abstraction, and class inheritance. It leads to faster software development, increased quality, easier maintenance, and flexible modifiability. These features make OOP the paradigm of choice for many developers.

Object-oriented programming languages have their roots in SIMULA 67. SIMULA 67 was the first OOP language to provide objects, classes, inheritance, and dynamic binding. SmallTalk was the next major contributor, followed by C++, Eiffel, and Java. All OOP languages support abstract data types using classes, class inheritance, and dynamic binding. Some procedural languages support abstract data types, but not inheritance and polymorphism, as in Ada 83, Moduls-2, and Microsoft Visual Basic. These last three languages are known as *object-based,* rather than object-oriented.

C++ and Java are the most popular OOP languages. Java has gained tremendous momentum in recent years. For this reason, we will use Java's syntax to demonstrate principles of OOP. For references on Java programming, please see Liang and Arnold and colleagues.

II. OBJECTS AND CLASSES

Object-oriented programming is all about objects. An object represents an entity in the real world that can be distinctly identified. For example, a student, a desk, a circle, and even a mortgage loan can all be viewed as objects. An object has a state and behavior. The state of an object consists of a set of fields, or attributes, coupled with their current values, which describe the properties of an object. The behavior of an object is defined by a set of *methods.* Figure 1 shows a diagram of an object with its data fields and methods. Shortly, you will see an example of a method.

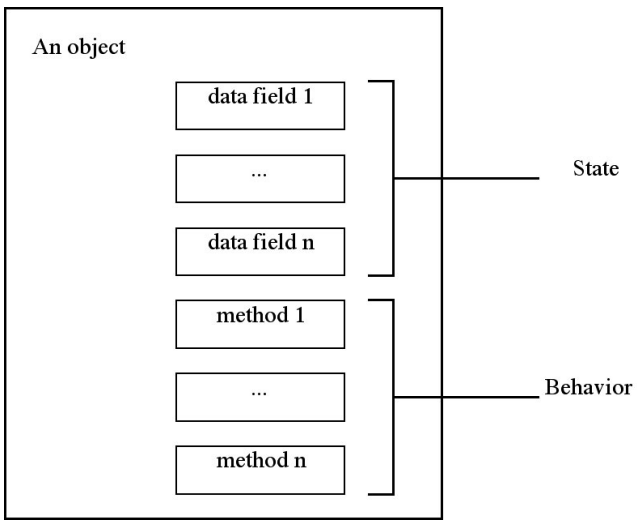


Figure 1 An object contains both state and behavior. The state defines the object and the behavior defines what the object does.

A `Person` object, for example, might have data fields of name, address, and age, which are several properties that in part characterize a person. The behavior of the person object that we would like to have is for its address and age to be obtainable and modifiable. A sample `Person` object is shown in Fig. 2. In the figure, values for each of the data fields are provided. Additionally, a number of methods are shown for manipulating the data.

An object can be perceived as a black box. You manipulate the object through its public methods. For example, to find out who a particular `Person` object corresponds to, you can invoke the object's `whoAmI` method. As expected, this method returns the value specified in the name field. The object's properties are usually defined as private to prevent direct modifications. Modifying data properties directly often causes programming errors that are difficult to debug.

Classes are constructs that define objects of the same type. In a Java class, data are used to describe properties, and methods are used to define behaviors. A class for an object contains a collection of methods and a group of data definitions. Listing 1 defines a class for `Person`. Java keywords are highlighted in listings.

Listing 1: Class definition for `Person`.

```
public class Person
{
  /**The name of this person*/
  private String name;

  /**The age of this person*/
  private int age;

  /**The address of this person*/
  private String address;

  /**Get the name of this person*/
  public String getName()
  {
    return name;
  }

  /**Set the name of this person*/
  public void setName(String name)
  {
    this.name = name;
  }

  /**Get the address of this person*/
  public String getAddress()
  {
```

```
    return address;
  }

  /**Set the address of this person*/
  public void setAddress(String
      address)
  {
    this.address = address;
  }

  /**Get the age of this person*/
  public int getAge()
  {
    return age;
  }

  /**Set the age of this person*/
  public void setAge(int age)
  {
    this.age = age;
  }

  /**Return the identity of this
      person*/
  public String whoAmI()
  {
    return "I am " + name;
  }
}
```

A class is a blueprint that defines what an object's data and methods will be. An object is simply an instance of a class. You can create many instances of a class (see Fig. 3). Creating an instance of a class is referred to as *instantiation*. The terms object and instance are often used interchangeably. The relationship between classes and objects is analogous to the relationship between an apple pie recipe and apple pies. You can make as many apple pies as you want from a single recipe. Different recipes result in different pies. However, the pies are all rather similar to each other.

This article uses the graphical notations adopted in the unified modeling language (UML) to illustrate classes and objects. UML has become the standard for object-oriented modeling. The UML notation we use is largely self-explanatory. For more information on UML, see Booch and colleagues and www.rational.com/uml.

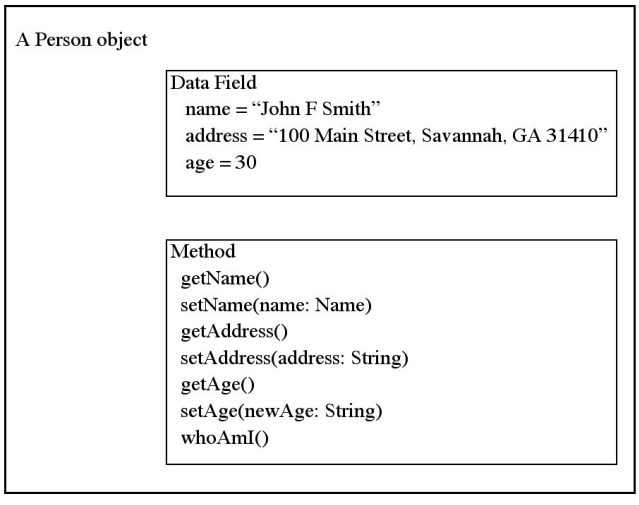


Figure 2 A Person object contains data fields of name, address, and age plus various methods for manipulating the object.

A. Creating Objects and Object Reference Variables

Objects are created through *constructors*. A constructor has exactly the same name as its defining class. Constructors are a special kind of method. Like methods, constructors can be overloaded, making it easier to develop objects with different initial data values. Constructors play the role of initializing objects. A constructor with no parameters is referred to as a *default constructor*. If a class does not define any constructors explicitly (the case in Listing 1), a default constructor is assumed implicitly. If a class defines constructors explicitly, a default constructor will not exist unless it is defined explicitly.

Listing 2 expands the `Person` class defined in Listing 1 with two constructors.

Listing 2: Class definition for `Person` with constructors.

```
public class Person
{
  /**Default constructor*/
  public Person()
  {
  }

  /**Construct a person with specified
      name, address, and age*/
  public Person(String name, String
      address, int age)
  {
    this.name = name;
    this.address = address;
    this.age = age;
  }
  /**Everything else the same as in
      Listing 1, hence omitted.*/
}
```

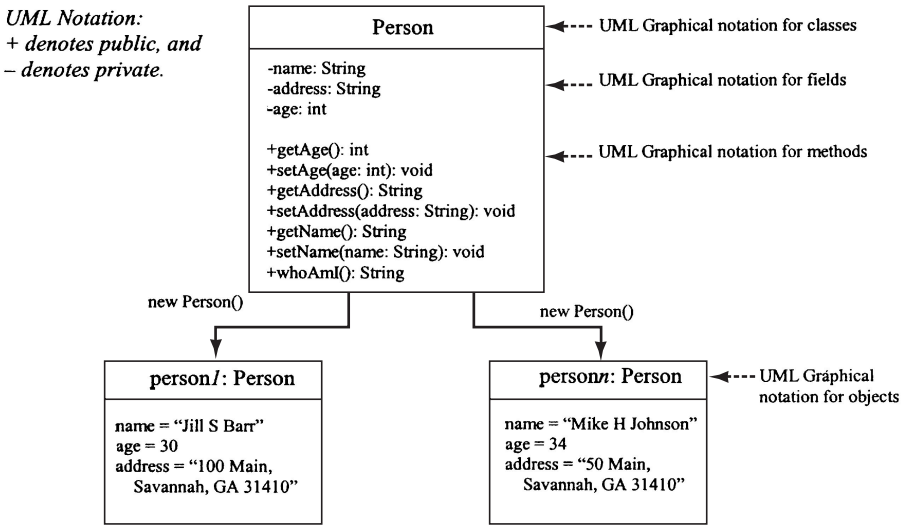


Figure 3 A class can have many objects.

Objects are created using the new operator on a constructor as follows:

```
new ClassName(arguments);
```

For example, `new Person()` creates an object of the `Person` class. Newly created objects are allocated in memory. Objects are accessed via object reference variables to the objects. Such reference variables are declared using the following syntax:

```
ClassName objectReference;
```

The types of reference variables are known as *reference types*. The following statement declares the variable `p` to be of the `Person` type:

```
Person p;
```

The variable `p` can reference a `Person` object. The following statement creates an object and assigns its reference to `p`.

```
p = new Person();
```

You can combine the declaration of an object reference variable, creation of an object, and assignment of the object reference to the variable together in one statement using the following syntax:

```
ClassName objectReference =
  new Class Name();
```

Below is a concrete example of this syntax:

```
Person p = new Person();
```

The variable `p` now holds a reference to a `Person` object.

When a variable of a reference type is declared, the variable holds a special value, *null*, which means that the variable does not reference any object. Once an object is created, its reference can be assigned to a variable. For example, the statement

```
p = new Person();
```

creates a `Person` object by allocating the memory space for the object and assigns its memory reference to the variable `p`. When you assign one variable into another, the other variable is set to the same value.

After an object is created, its data can be accessed and its methods can be invoked using the following dot notation:

`objectReference.data`—References an object's data

`objectReference.method(arguments)`—Invokes an object's method

For example, `p.setName("Sam J Fox")` invokes the setName method of object `p`. Methods are invoked as operations on objects. If the address property were defined as public, the variable address could be accessed using `p.address`.

B. Modularity

Modularity is a fundamental principle of programming. It is intended to control the complexity of a software system through the use of the *divide and conquer* approach. A complex system can be decomposed into a set of loosely coupled, but cohesive modules.

The concept of modularity was developed for programming using procedural programming languages in which the modules take the form of procedures and functions. The same concept applies to OOP in which modules take the form of classes. Decomposition of a software system into smaller modules in an object-oriented system means designing classes to model the system.

A class should be cohesive and describe a single entity or a set of similar operations. You can use a class for students, for example, but do not combine students and staff in the same class since students and staff will have different sets of operations. A single entity with too many responsibilities can be broken into several classes to separate responsibilities.

C. Class Abstraction

One of the key characteristics of OOP is *class abstraction*. Class abstraction means to separate class implementation from the use of the class. The creator of the class provides a description of the class and lets the user know how the class can be used. The collection of methods and fields that are accessible from outside a class, together with the description of how these members are expected to behave, serves as the class's *contract*. The user of the class does not need to know how the class is implemented. The details of implementation are encapsulated and hidden from the user. For example, you can create a Circle object and find the area of the circle without knowing how the area is computed. There are many real-life examples that illustrate the concept of class abstraction.

Consider building a computer system, for instance. Your PC is made up of many components, such as a CPU, CD-ROM, floppy disk, motherboard, fan, and so on. Each component can be viewed as an object that has properties and methods. To get the components to work together, all you need to know is how each component is used and how it interacts with the others. You do not need to know how it works internally. The internal implementation is encapsulated and hidden from you. You can build a computer without even knowing how a single component is implemented.

The computer system analogy just presented precisely mirrors the object-oriented approach. Each component can be viewed as an object of the class for the component. For example, you might have a class that models all kinds of fans for use on a computer with properties like fan size, speed, and so on, and methods like start and stop. A specific fan is an instance of this class with specific property values.

Consider paying a mortgage, for another example. A specific mortgage can be viewed as an object of a `Mortgage` class. Interest rate, loan amount, and loan period are its data properties, and computing monthly payment and total payment are its methods. When you buy a house, a mortgage object is created by instantiating the class with your specific mortgage interest rate, loan amount, and loan period. You can then use the mortgage methods to easily find the monthly payment and total payment of your loan. As a user of the `Mortgage` class, you do not need to know how these methods are implemented.

D. Instance Variables (Methods) and Static Variables (Methods)

The data field name in the `Person` class is referred to as an *instance variable* because it is dependent on a specific instance. For the same reason, the method `getName` in the `Person` class is referred to as an *instance method,* because you can only invoke it on a specific instance.

If you want all the instances of a class to share data, you can use *static variables,* also known as *class variables*. Static variables store values for the variables in a common memory location. Because of this common location, all objects of the same class are affected if one object changes the value of a static variable. Similarly, static methods are not tied to a specific instance. Static methods can be directly invoked using a class name without a specific instance of the class.

To declare a static variable (respectively, method), insert the modifier *static* in the variable (respectively, method) declaration.

As example, suppose that you want to track the number of objects of the `Person` class created; you can define the static variable `numOfObjects` as follows:

```
private static int numOfObjects;
```

You may provide a static method to retrieve `numOfObjects` as follows:

```
public static int getNumOfObjects()
{
  return numOfObjects;
}
```

Listing 2 expands the `Person` class defined in Listing2 by adding a new static property `numOfObjects` and its associated accessor methods.

Listing 2: Class definition for Person with a static property and accessor methods.

```
public class Person
{
  private static int numOfObjects;
  /**Default constructor*/
  public Person()
  {
    numOfObjects++;
  }

  /**Construct a person with specified
      name, address, and age*/
  public Person(String name, String
      address, int age)
  {
  numOfObjects++;
    this.name = name;
    this.address = address;
    this.age = age;
  }

  /**Return number of objects*/
  public static int getNumOfObjects()
  {
    return numOfObjects;
  }

  /**Everything else the same as in
      Listing 1, hence omitted.*/
}
```

Static variables can be accessed from instance or static methods in the class. Instance variables can be accessed from instance methods in a class, but not from static methods in the class, since static methods are invoked regardless of specific instances.

III. CLASS INHERITANCE

With OOP, you can derive new classes from existing classes. This is called *inheritance*. In OOP terms, a class C1 derived from another class C2 is called a *subclass,* and C2 is called a *superclass*. Sometimes a superclass is referred to as a *parent class* or a *base class,* and a subclass is referred to as a *child class,* an *extended class,* or a *derived class*. You can reuse or change the methods defined in the superclasses in the subclass, as well as creating new data and new methods in subclasses. Subclasses usually have more functionality than their superclasses. Inheritance is a natural way to model the *is-a* relationship, which we discuss later on in this article. Inheritance provides for code and structural reuse so is a very important programming tool.

A. Overriding Methods

A subclass inherits methods from a superclass. Sometimes, it is necessary for the subclass to modify the methods defined in the superclass. This is referred to as *method overriding*. Overriding is to provide an implementation for the method that is appropriate for the subclass. For example, you can create a subclass named `Student` that extends the `Person` class. The `Student` class, as shown in Listing 3, overrides the `whoAmI` method defined in the `Person` class.

Listing 3: Class `Student`

```
public class Student extends Person
{
  /**The major of this student*/
  private String major;

  /**The score of this student*/
  private int score;

  /**Default constructor*/
  public Student()
  {
    super();
  }

  /**Construct a student with specified
      name, address, age, and major*/
  public Student(String name, String
      address,
    int age, String major)
  {
    super(name, address, age);
    this.major = major;
  }

  /**Override whoAmI*/
  public String whoAmI()
  {
    return super.whoAmI() + " in " +
        major;
  }
}
```

The *super* keyword refers to the superclass of the Student class, i.e., Person. The **`super`**`()` call invokes the default constructor of the `Person` class, and super(name, address, age) invokes the constructor `Person`(name, address, age). The call `super.whoAmI()` invokes the `whoAmI` method in the `Person` class.

B. Abstract Classes

In the inheritance hierarchy, classes become more specific and concrete with each new subclass. If you move from a subclass back up to a superclass, the classes become more general and less specific. Class design should ensure that a superclass contains common features of its subclasses. Sometimes a superclass is so abstract that it cannot have any useful specific instances. Such a class is referred to as an *abstract class*.

Consider geometric objects. Suppose you want to design some classes to model geometric objects including circles, cylinders, and rectangles. Geometric objects have many common properties and behaviors. Geometrical objects can be drawn in a certain color and be filled or left unfilled. Color and filled are examples of common properties. Common behaviors include the facts that their areas and perimeters can be computed. Thus, a general class `GeometricObject` can be used to model all the geometric objects mentioned earlier. This class contains the properties color and filled, and the methods `findArea` and `findPerimeter`. A cylinder is a special type of circle, and thus it shares common properties and behaviors with a circle. It therefore makes sense to define the `Cylinder` class so it extends the `Circle` class. Figure 4 illustrates the relationship of the classes for geometric objects in UML.

The methods `findArea` and `findPerimeter` cannot be implemented in the `GeometricObject` class because their implementation is dependent on the specific type of geometric object. Such methods are referred to as *abstract methods*. Abstract methods are implemented in concrete subclasses.

C. Polymorphism and Dynamic Binding

An object of a subclass can be used by any code designed to work with an object of its superclass. For example, if a method expects a parameter of the `Circle` type, you can invoke it with a `Cylinder` object. A `Cylinder` object can be used as both a `Cylinder` object and a `Circle` object. This feature is known as Polymorphism (from the Greek word meaning "many forms"). A method may be defined in a superclass, but is overridden in a subclass. The implementation of the method that is used on a particular call is determined dynamically at runtime. This capability is known as *dynamic binding* in OOP.

Dynamic binding works as follows. Suppose an object ○ is an instance of classes `A1, A2, . . ., An-1`, and `An`, where `A1` is a subclass of `A2, A2` is a subclass of `A3, . . .`, and `An-1` is a subclass of `An`. That is, `An` is the most general class and `A1` is the most specific class. If ○ invokes a method `p`, the runtime system searches the implementation for this method `p` in `A1, A2, . . ., An-1` and `An`, in this order, until it is found. Once an implementation is found, the search stops and the first found implementation is invoked.

To make things concrete, here is an example of dynamic binding at work:

```
public class TestPolymorphism
{
  /**Main method*/
  public static void main(String[]
      args)
  {
```

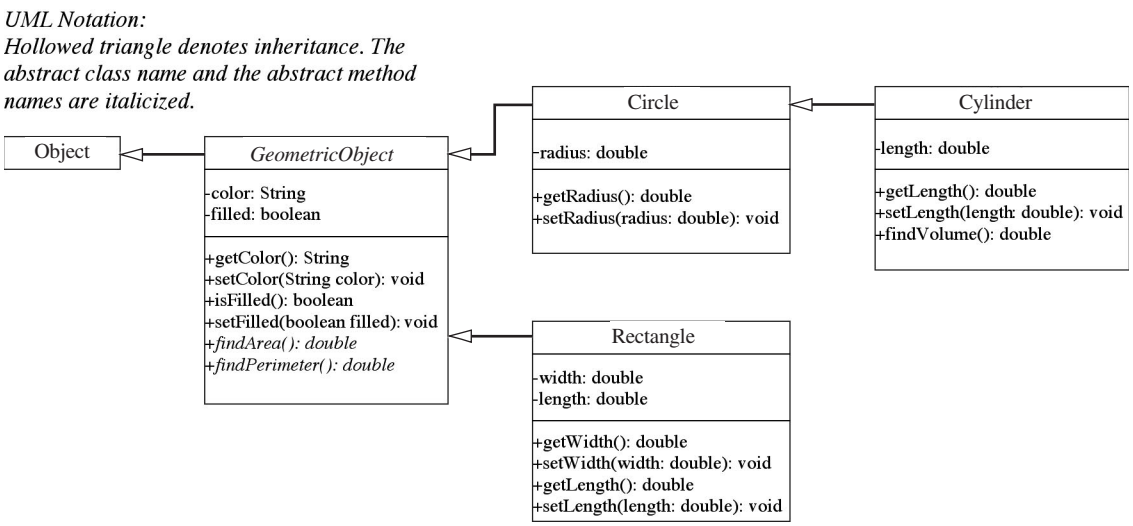


Figure 4 The `GeometricObject` class models the common features of some geometric objects.

```
    /**Declare and initialize two
        geometric objects*/
    GeometricObject geoObject1 = new
        Circle(5);
    GeometricObject geoObject2 = new
        Rectangle(5, 3);

    System.out.println("The two
        objects have the same area? " +
        equalArea(geoObject1,
        geoObject2));
  }

  /**A method for comparing the
      areas of two geometric objects*/
  static boolean equalArea
      (GeometricObject object1,
      GeometricObject object2)
  {
    return object1.findArea() ==
        object2.findArea();
  }
}
```

In this example, the `equalArea` method is invoked with arguments `geoObject1` and `geoObject2`, both of which are of the `GeometricObject` type. The object `geoObject1` is an instance of the `Circle` class, and `geoObject2` is an instance of the `Rectangle` class. When computing their individual areas, their respective `findArea` methods are dynamically determined by the system at runtime. This process is dynamic binding.

Polymorphism allows methods to be more easily extended and used generically for a wide range of object arguments. If a method's parameter type is a superclass (for example, `GeometricObject`), you may pass an object to this method of any of the parameter type's subclasses (for example, in this case, `Circle` or `Rectangle`). When an object (for example, a `Circle` object or a `Rectangle` object) is used in the method, the particular implementation of the method of the object (for example, `findArea`) that is invoked is determined dynamically.

D. Casting Objects and Checking Object Types

In the previous section, the statement

```
GeometricObject geoObject1 = new
   Circle(5);
```

assigns a `Circle` object to a variable of the `GeometricObject` type. This is known as *implicit casting* where an object of the subclass is assigned to a variable of its superclass. Sometimes, you need to cast an object from a superclass type into a subclass in order to explore properties and functions in the subclass. This has to be done through explicit casting. Here is an example of casting `geoObject1` to a `Circle` object using the syntax with the target object type enclosed inside the parentheses.

```
Circle circle = (Circle)geoObject1;
```

For the casting to be successful, you must make sure that the object to be cast is an instance of the subclass. If the superclass object is not an instance of the subclass, an exception would occur. It is good practice, therefore, to ensure that the object is an instance of its subclass type before attempting a casting. This can be accomplished by using the instanceof operator in Java as follows:

```
if (geoObject1 instanceof Circle)
  Circle circle = (Circle)geoObject1;
```

E. Single Inheritance versus Multiple Inheritance

Multiple inheritance means that a subclass can inherit from two or more superclasses. C++ allows multiple inheritance, but Java allows only single inheritance, that is, a subclass can inherit only one superclass.

Multiple inheritance is useful when a subclass needs to combine multiple contracts and inherit some, or all, of the implementation of those contracts. For example, the `AmericanStudent` class needs to inherit from both the Student class and the American class. But multiple inheritance imposes additional difficulties. For example, if both superclasses have an identical method with different implementations, which implementation should the subclass use? One possible solution to this dilemma would be to force the subclass to choose an implementation. Another possibility would be to reimplement the method. Either solution would complicate class inheritance.

Java uses the single inheritance model to simplify class inheritance. To circumvent the single inheritance restriction, Java introduces a new construct, called *interface*.

F. Java Interfaces

Sometimes it is necessary to derive a subclass from several classes, thus inheriting their data and methods. Java, however, does not allow multiple inheritance. If

you use the extends keyword to define a subclass, it allows only one parent class. With interfaces, you can obtain the effect of multiple inheritance.

An *interface* is a class-like construct that contains only constants and abstract methods. In many ways, an interface is similar to an abstract class, but an abstract class can contain constants and abstract methods as well as variables and concrete methods.

The syntax to declare an interface is as follows:

```
modifier interface InterfaceName
{
  /**Constant declarations*/
  /**Method signatures*/
}
```

So, there are a series of constant declarations with a group of method signatures.

An interface is treated like a special class in Java. As with an abstract class, you cannot create an instance for the interface using the new operator, but in most cases you can use an interface more or less the same way you use an abstract class. For example, you can use an interface as a data type for a variable, as the result of casting, and so on.

Suppose you want to design a generic method to find the larger of two objects. The objects could be, for example, students, circles, or cylinders. Because compare methods are different for various types of objects, you need to define a generic compare method to determine the order of the two objects. Then you can tailor the method to compare students, circles, or cylinders. For example, you can use student ID as the key for comparing students, radius as the key for comparing circles, and volume as the key for comparing cylinders. You can use an interface to define a generic `compareTo` method, as follows:

```
/**Interface for comparing objects*/
public interface Comparable
{
  public int compareTo(Object o);
}
```

The `compareTo` method determines the order of this object with the specified object `o` , and returns a negative integer, zero, or a positive integer if this object is less than, equal to, or greater than, respectively, the specified object `o`.

A generic `max` method for returning the larger of two objects can be declared in a class named `Max`, as follows:

```
public class Max
{
  /**Return the larger between two
      objects*/
  public static Comparable max
     (Comparable o1, Comparable o2)
  {
    if (o1.compareTo(o2) > 0)
      return o1;
    else
      return o2;
  }
}
```

The `Max` class contains the static method named max. To use the `max` method to find a maximum between two objects, one needs to implement the `Comparable` interface for the class of these objects. For example, you can define a `Student` class that implements the `Comparable` interface as follows:

```
public class Student extends Person
  implements Comparable
{
  /**The major of this student*/
  private String major;

  /**The score of this student*/
  private double score;

  /**Default constructor*/
  public Student()
  {
    super();
  }

  /**Construct a student with
      specified name, address, age,
      major, and score*/
  public Student(String name, String
     address,
    int age, String major, double
       score)
  {
    super(name, address, age);
    this.major = major;
    this.score = score;
  }

  /**Get the major of this student*/
  public String getMajor()
  {
    return major;
  }

  /**Set the major of this student*/
  public void setMajor(String major)
```

```
  {
    this.major = major;
  }

  /**Get the score of this student*/
  public double getScore()
  {
    return score;
  }

  /**Set the score of this student*/
  public void setScore(double score)
  {
    this.score = score;
  }

  /**Implement the compareTo method*/
  public int compareTo(Object o)
  {
    if (score - ((Student)o).score
        > 0)
      return 1;
    else if (score -
        ((Student)o).score == 0)
      return 0;
    else
      return -1;
  }

  /**Override whoAmI*/
  public String whoAmI()
  {
    return super.whoAmI() + " in "
        + major;
  }
}
```

Having extended the `Person` class, you can now create two instances of the `Student` class and use the max method in the `Max` class to find a student with a better score. This is done as follows:

```
Student s1 = new Student("John", "10
    Main Street", 25, "Computer
    Science", 90.5);
Student s1 = new Student("Jennifer",
    "11 Main Street", 25, "Computer
    Science", 96.5);
Student maxS = (Student)Max.max(s1,
    s2);
```

IV. OOP METHODOLOGIES

While there are many object-oriented methodologies for designing classes, UML has become the industry standard for class analysis and design notation. The utilization of this notation leads to a design methodology.

To build an object-oriented system, the following steps are usually involved:

1. Identify classes for the system.
2. Describe attributes and methods in each class.
3. Establish relationships among classes.
4. Create classes.

The first step is to identify classes for the system. There are many strategies for identifying classes in a system. One strategy that works well is to study how the system works and select a number of typical cases, or scenarios that can be modeled as classes.

The second step is to describe attributes and methods in each of the classes you have identified. You will need to decide whether or not to provide accessor methods for these attributes.

The third step is to establish relationships among the classes. The relationships usually emerge pretty clearly from the analysis of the previous two steps. The first three steps are closely intertwined. When you identify classes, you also should think about the relationships among classes.

The fourth step is to write the code for the classes. There are object-oriented development tools such as `Rational Rose` and `Together` that make this process more efficient. These tools can be used to draw class diagrams graphically and generate class source code from the diagram automatically. Once you have carried out steps one through four, you may find it necessary to revisit each of the steps in order to tune your design.

A. Association

Previous sections have demonstrated how to use UML to describe classes. This section focuses on modeling relationships among classes. The relationships can be classified into three types: association, aggregation, and inheritance.

Association represents a general binary relationship that describes an activity between two classes. For example, a student taking a course is an association between the `Student` class and the `Course` class, and a faculty member teaching a course is an association between the `Faculty` class and the `Course` class. These associations can be represented in UML graphical notations, as shown in Fig. 5.

An association is illustrated using a solid line between the two classes with an optional label that

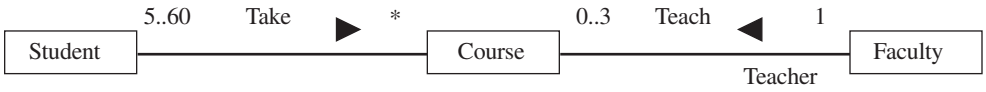


Figure 5 A student may take any number of the courses (denoted by *). A faculty member may teach at most three courses. A course may have 5 to 60 students. A course is taught by only one faculty member.

describes the relationship. In Fig. 5, the labels are *Take* and *Teach*. Each relationship may have an optional small black triangle that indicates the direction of the relationship. In Fig. 5, the direction indicates that a student takes a course, as opposed to a course taking a student.

Each class involved in the relationship may have a *role name* that describes the role played by the class in the relationship. In our example, *teacher* is the role name for the `Faculty` class.

Each class involved in the association may specify a multiplicity. A multiplicity could be a number or an interval that specifies how many objects of the class are involved in the relationship. The character * means an unlimited number of objects, and an interval l..u means the number of objects should be between l and u, inclusive. In our example, each student may take any number of courses, and each course must have at least 5 students and at most 60 students. Each course is taught by only one faculty member, and a faculty member may teach 0–3 courses.

Association may exist between objects of the same class. For example, a person may have a supervisor. This is illustrated in Fig. 6.

An association is usually represented as a data field in the class. For example, in the association "a course is taught by a faculty member," a faculty member may be a data field in the `Course` class, as follows:

```
public class Course
{
  private Faculty faculty;

  /**Constructors*/

  /**Methods*/
}
```

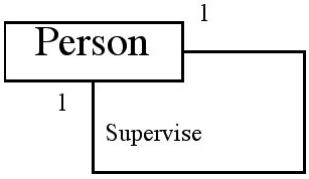


Figure 6 A person may have a supervisor.

In the association "a person has a superviser," a person may be a data field in the `Person` class, as follows:

```
public class Person
{
  private Person supervisor;

  /**Constructors*/

  /**Methods*/
}
```

The types of relationships that programmers find they want to express among objects are often the types that can be qualified under association.

B. Aggregation

Aggregation is a special form of association that represents an owner relationship between two classes. Aggregation models relationships such as *has-a, part-of, owns,* and *employed-by*. An object may be owned by several other aggregated objects. If an object is owned exclusively by an aggregated object, the relationship between the object and its aggregated object is referred to as *composition*. For example, a publisher that owns a magazine is a composition between the `Publisher` class and the `Magazine` class, whereas a publisher that has consultants is an aggregation between the `Publisher` class and the `Consultant` class, since a consultant may work for several publishers. In UML, a filled diamond is attached to the `Publisher` class of the association to denote the composition relationship, and an empty diamond is attached to the aggregated class of the association to denote the aggregation relationship, as shown in Fig. 7.

The publishing example presented here illustrates the importance of being able to express relationships of the type that can be modeled using aggregation.

C. Inheritance

Inheritance models the *is-a* relationship between two classes. A strong is-a relationship describes a direct inheritance relationship between two classes. A weak is-a relationship describes that a class has certain properties.

Figure 7 A magazine is owned by a publisher, and a consultant may work for several publishers.

A strong is-a relationship can be represented using class inheritance. For example, a student is a person, and a faculty member is a person. You can define a class for `Student` that inherits from the `Person` class, and a class for `Faculty` that inherits from the `Person` class, as follows.

```
public class Student extends Person
{
  /**Constructors*/

  /**Methods*/
}

public class Faculty extends Person
{
  /**Constructors*/

  /**Methods*/
}
```

A weak is-a relationship can be represented using interfaces. For example, the weak is-a relationship "students are comparable based on their scores" can be represented by implementing the `Comparable` interface, as follows:

```
public class Student extends Person
implements Comparable
{
  /**Constructors*/

  /**Methods*/

  /**Implement the compareTo method*/
  public int compareTo(Object object)
  {
    // Write comparison code here
  }
}
```

It is clear that there are many programming situations where inheritance relationships among classes are important.

D. Modeling Dynamic Behavior

The UML diagrams presented so far describe the properties and methods of a class or the static relationships among classes. This section introduces the sequence diagrams and statechart diagrams that model dynamic behaviors among the objects. Such modeling is important for complicated OOP as it gives the programmer a systematic way of tackling dynamic behaviors.

1. Sequence Diagrams

Sequence diagrams describe interactions among objects by depicting the time ordering of method invocations. A sequence diagram consists of the following elements, as shown in Fig. 8:

- *Class role* represents the roles the object plays. The objects at the top of the diagram represent class roles.
- *Lifeline* represents the existence of an object over a period of time. A vertical dotted line extending from the object is used to denote a lifeline.
- *Activation* represents the time during which an object is performing an operation. Thin rectangles placed on lifelines are used to denote activations.
- *Method invocation* represents communication between objects. Horizontal arrows labeled with method calls are used to denote method invocations.

Suppose you have two classes named `PrintableObject` and `Print`. An object of the `Print` class is to print a `Printable` object to a printer. The `Print` class has a method `printAnObject(Object o)`, which is invoked to print an object. After printing is finished, the `Print` object notifies the `Printable Object` by invoking the object's finished method. The interactions between the objects just described are illustrated in Fig. 9.

2. Statechart Diagrams

Statechart diagrams describe the flow of control of an object. A statechart diagram contains the following elements, as shown in Fig. 10:

- *State* represents a situation during the life of an object in which it satisfies some condition,

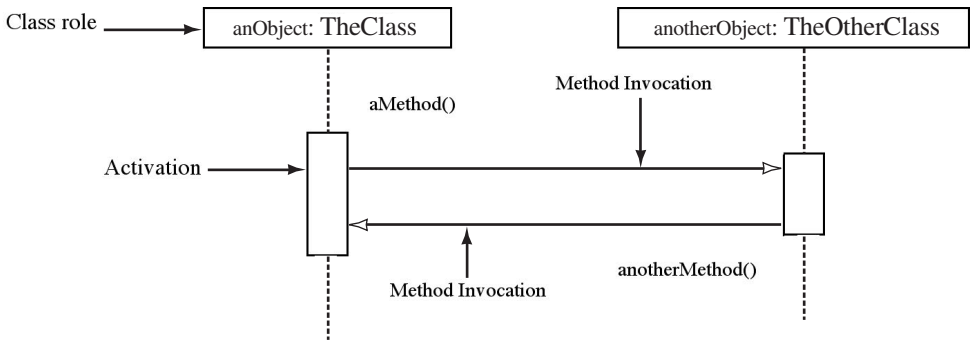


Figure 8 Sequence diagrams describe interactions between objects.

performs some action, or waits for some event to occur. Each state has a name. Rectangles with rounded corners are used to represent states. The small filled circle with an arrow leaving it points to the initial state.

- *Transition* represents the relationship between two states, indicating that an object will perform some action to transfer from one state to the other. A solid arrow with appropriate method invocation is used to denote a transition.

The statechart diagram in Fig. 10 can be used to describe the flow of control of a Java *applet,* as shown in Fig. 11. The init method is invoked when the applet is first loaded and again if it is reloaded. The init method transfers the applet from the start to the Initialized state. The start method is invoked after the init method. It is also called whenever the applet becomes active again after the web page containing the applet is revisited. The start method transfers the applet to the Started state. The stop method is invoked when the user leaves the corresponding web page. The stop method transfers the applet to the Stopped state. The destroy method is invoked when the web browser exits normally. The destroy method transfers the applet to the Destroyed state.

Compact statechart diagrams pack a great deal of information into an easily understandable picture. They are very useful for helping programmers reason about OOP.

V. SUMMARY

This article provided an overview of OOP including applying class abstraction, class inheritance, and polymorphism in the OOP language Java.

The key to OOP is to model an application in terms of cooperative objects. Carefully designed classes are critical when a project is being developed. There are many levels of abstraction in system design. Method abstraction is applied in the procedural development approach. Methods are a means to group statements. Classes extend abstraction to a higher level and provide a means of grouping methods. Classes do more than just group methods, however, they also contain

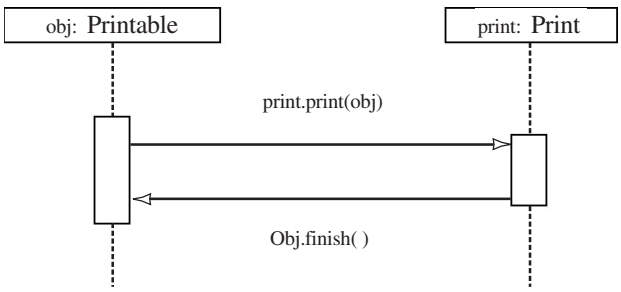


Figure 9 The Printable object invokes the print method of a Print object to print the object, and the Print object invokes the Printable object's finished method to notify it that the print job is finished.

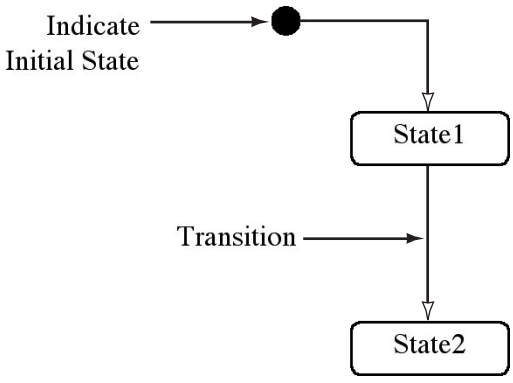


Figure 10 Statechart diagrams describe the flow of control of an object.

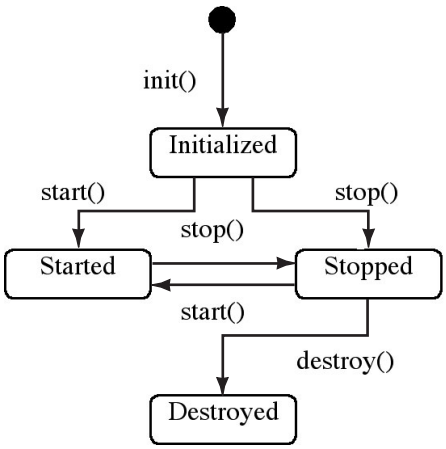


Figure 11 The flow of control of an applet can be conveniently described using a statechart diagram.

data fields. Methods and data fields together describe the properties and behaviors of classes. In the development of an object-oriented system, class abstraction is applied to decompose the problem into a set of related classes, and method abstraction is applied to design individual classes.

The power of classes is further extended by inheritance. Inheritance enables a subclass to extend the contract and the implementation of an existing superclass without knowing the details of the superclass. The methods of the superclass can be overridden in the subclass. Polymorphism allows methods to be more easily extended and used generically for a wide range of object arguments. If a method's parameter type is a superclass, you may pass an object of any of the type's subclass. When an object is used in a method, the implementation of the method of the object that is invoked is determined dynamically.

Because of the many advantages they provide, OOP languages have gained a great deal of popularity in recent years. They are now being taught as the first programming language in many computer science curriculums.

SEE ALSO THE FOLLOWING ARTICLES

Artificial Intelligence Programming • Automata Theory • Cohesion, Coupling, and Abstraction • Data Modeling: Object-Oriented Data Model • Goal Programming • Programming Languages Classification • Pseudocode

BIBLIOGRAPHY

Arnold, K., Gosling, J., and Holmes, D. (2000). *The Java programming language,* Third Edition. Reading, MA: Addison-Wesley.

Booch, G. (1994). *Object-oriented analysis and design with applications,* 2nd Ed. Redwood City, CA: Benjamin-Cummings.

Booch, G., Jacobson, I., and Rumbaugh, J. (1998). *The unified modeling language user guide.* Reading, MA: Addison-Wesley.

Dahl, O.-J., and Nygaar, K. (1967). SIMULA 67 Common Base Proposal. Norwegian Computing Center Document, Oslo.

Goldberg, A., and Robson D. (1983). *Smalltalk-80: The language and its implementation.* Reading, MA: Addison-Wesley.

Harmon, P. (1993). *Objects in action: Commercial applications of object-oriented technologies.* Reading, MA: Addison-Wesley.

Joy, B., Gosling, J., and Steele, G. (2000). *The Java language specification, Second Edition,* Reading, MA: Addison-Wesley.

Liang, Y. D. (2000). *Introduction to Java programming,* Third Edition. Englewood Cliffs, NJ: Prentice Hall.

Meyer, B. (1991). *Eiffel: The language.* Englewood Cliffs, NJ: Prentice Hall.

Stroustrup, B. (1986). *The C++ Programming Language.* Reading, MA: Addison-Wesley.

On-Line Analytical Processing

Robert J. Thierauf
Xavier University

GLOSSARY

business intelligence Gives a better understanding of the area under study by employing OLAP functionality, data warehousing, and data mining.
consolidation The ability to combine data into ever larger aggregations.
critical success factors Factors that are critical in making or breaking the organization.
data mining A knowledge discovery process that extracts previously unknown, actionable information from very large databases.
data warehousing A very large storage facility that serves as a delivery mechanism to provide data for obtaining desired information and knowledge about a company's operations.
drill down The ability to seek increasingly lower levels of information on a particular topic.
multidimensional analysis The capability to look at different dimensions of the same data to better understand a company's operations.
multidimensional data warehouse A subject-oriented multidimensional database that is designed specifically for analysis and support activities of decision makers.
OLAP and the Web Application of on-line analytical processing can be extended to the Web to get a better handle on a company's operations.
OLAP software Multidimensional analytical tools for accessing, storing, and manipulating decision support information and knowledge.
on-line analytical processing An approach to aid decision makers in *getting at the whys* of the problem(s) under study by using multidimensional analysis.
problem finding The process of going out into the future and determining potential problems and identifying future opportunities that are related to these problems.
slice and dice The ability to look at the database from different viewpoints.
Rubik's cube of data Capability to twist and twirl data to work through different scenarios to get at the *whys* of the situation

I. INTRODUCTION TO ON-LINE ANALYTICAL PROCESSING (OLAP)

For all sizes of organizations, a newer type of computer-based information system, that is, an on-line analytical processing (OLAP) system, is expanding the computer's role in everyday operations. An important goal of an OLAP system is to provide information and knowledge based on insight and understanding that decision makers (from the highest level to the lowest level) need. In effect, an organization-wide OLAP system challenges decision makers to evaluate the status quo. An OLAP system allows decision makers to tailor their information and knowledge requirements by discriminating according to user-defined criteria. An OLAP system is capable of making comparisons, analyzing trends, and presenting historical and current

data. It is interactive and needs users' input to search and summarize the pertinent data of interest. More recently data mining (also known as knowledge discovery) has been used to obtain information and knowledge from data.

Unlike a traditional information system (IS) presentation, it can distinguish between vital and seldom-used data. An on-line analytical processing system can track and evaluate key critical success factors for decision makers, which is valuable in assessing whether or not the organization is meeting its corporate objectives and goals. In general, an OLAP system can shape an information and knowledge base that decision makers can use to make better informed decisions.

In the past, information systems basically used the computer as a means of providing information to solve recurring operational problems. A better approach today is OLAP systems, which position decision makers at the center of the decision-making process. By increasing the capabilities of decision makers and by eliminating impediments to their rational functioning, an OLAP system improves the chances that an organization will achieve its goals of increased sales, profits, and so forth, by placing information and knowledge in the hands of decision makers at the proper time and place, and by providing flexibility in their choice and sequence of information analysis and in the ultimate presentation of results. From this perspective, OLAP systems provide essential information and knowledge to decision makers so that they can better cope with changing times.

II. OLAP DEFINED

As will be seen later on the essentials of OLAP, multidimensional analysis looks at data from different perspectives, as defined by the end user. In order to undertake the desired analysis, the end user acts as the front end. In the middle is an OLAP server where data is retrieved from back-end databases and staged in an OLAP-multidimensional database for retrieval by front-end systems. At the back end are databases and data warehouses that are multidimensional, relational, or specialized relational. In a typical situation, OLAP uses a multidimensional database that reads and aggregates large groups of diverse data to analyze relationships and look for patterns, trends, and exceptions.

Based on this overview, OLAP technology gives end users easy access to large volumes of numerical data on-line. Usually, data is transferred from on-line transactional processing systems to a multidimensional database for data summarization. On-line analytical processing functions between the back-end databases and data warehouses and the front end or end user. The multidimensional engine that drives OLAP technology enables the system to analyze large volumes of data quickly. Retrieval times are aided by the use of efficient data storage algorithms that preaggregate data, saving it in fewer places but with more intelligent directional markers.

A most important feature of OLAP is its drill-through capability from the highest level to the lowest level of detail. As such, OLAP is a user-friendly, quick, and flexible client/server solution that lets users drill down into data housed in databases and data warehouses for analysis. To get at the detail level, there is need for a seamless integration of these data whether they are transaction-processing databases for everyday or data warehouses for historical data. Thus, there is a need for an open architecture which allows the user to obtain the desired data for analysis from any database or data warehouse in an organization.

Essentially, OLAP can be defined as an open-systems architecture that provides the user with rapid retrieval of data from company databases and/or data warehouses for the purpose of summarization and multidimensional analysis. Typically, the end result of the analysis is information and knowledge per user-defined dimensions for getting at the *whys* of the problem(s) under study. Where deemed necessary, a drill down and/or dice capability is utilized to get at the appropriate level of detail in the analysis.

III. FOCUS OF OLAP—CHALLENGING A COMPANY'S STATUS QUO

Fueling the interest in OLAP is going beyond the traditional approach to finding out what went wrong. In the past, a typical relational database-management system (RDBMS) query told the decision maker *what* had happened, but not *why*. The focus, then, has shifted to finding out why. In a few words, "what" is not satisfying, that is, it only provides the decision maker with an opportunity to solve today's superficial problems; whereas "why" helps the decision maker to get to the root of the problems and prevent future occurrences.

To put the OLAP concept into perspective, a sales manager, for example, wants to know why sales are down in a specific region for a specific product, and he would like to know if this would change in the future. The sales manager wants to know it *now*, without being subjected to information technology (IT) backlog found in most IS departments. Needless to say, this is where OLAP comes into play. To get at OLAP

information and knowledge that challenges a company's status quo by asking why, there is need to go beyond extracting data from company databases for traditional two-dimensional analysis and provide the ability for managers at all levels to employ multidimensional analysis in finalizing their decisions. Or, to state it another way, the information and knowledge useful to a typical manager should center around information organized to answer questions that challenge an organization's strategy today and tomorrow.

A. Gaining Competitive Advantage

From past surveys of American competitiveness, the consensus is that the competitiveness of the United States has deteriorated and that diminished competitiveness will hurt America's economic performance for the foreseeable future. The net result is that the current situation represents a threat to the country's standard of living and economic power. In light of these comments, there is need for organizations "to shift gears" by employing an appropriate information technology to gain a competitive advantage. It is the job of IS management, working with top management, to bring about the needed change in the *competitive landscape* by employing the proper approach to organization-wide computing activities.

Essentially, an important thrust of OLAP systems, whether they are on an individual or a group-oriented basis, is to gain competitive advantage. The technology of personal computers and data communications is changing the parameters of competition in every industry whether it is manufacturing- or service-oriented. Formerly, information technology was directed toward being a "storekeeper" of data and information. In today's fast-changing world, information technology has to facilitate change so that the organization remains competitive. The importance of OLAP technology as a competitive weapon in furthering the organization's objectives, goals, and strategies is no less applicable for the small- to medium-size company than for the large-size company.

B. Leveraging a Company's Critical Success Factors

Although the typical manager undertakes a number of roles, the accent today is on relating decision making to the traditional *problem-solving* processes plus the utilization of the newer *problem-finding* process to expand the manager's view of how problems can be solved and how new opportunities can be identified for implementation. In the process of making a wide range of decisions, there is need for the manager to take into account the organization's critical success factors (CSFs), that is, those factors that are critical in making or breaking the organization. The process to identify CSFs was originally defined by John Rockart of MIT (in the late 1970s). It is sufficient to say that there are a specific and limited number of areas in which satisfactory results will dramatically affect the competitive performance of an organization. Typically, these areas are the ones to be measured and evaluated. As the old saying goes, "if you can't measure, you can't manage." Similarly, "what you measure wrong, you manage wrong." Establishing the wrong measures will lead to far worse results than establishing no measures at all.

Depending on the strategic direction of an industry and the specific organization, there are different ways of managing change and reshaping a business. The CSF process identifies what these specific needs are. Also, an OLAP system can be used to monitor these CSFs. In turn, OLAP can be used to analyze these CSFs such that their thorough analysis leads to new and different ways of exploring opportunities for a company. This approach can result in leveraging a company's CSFs for its betterment. Thus, by using a CSF process, an OLAP system can provide access to selected views of information and knowledge that allow company managers and their staffs to direct their present and future operations in an effective manner.

C. Improving an Organization's Productivity

Related to gaining competitive advantage and leveraging a company's critical success factors is the area of improving management productivity. Discussions have generally ignored the productivity of managers at the top. Attention has focused on assembly-line robots or the implementation of the paperless office, whereby the productivity of individual workers at the lower levels of an organization is increased. However, if the management of an organization decides to launch a new product, which the customer will not buy, it is irrelevant whether the workers actually assembling the product are performing their jobs efficiently. Having the right product at the right time, however, has a much greater impact on the organizational productivity than gaining an incremental improvement in labor or clerical productivity.

Because a manager makes decisions and not products per se, his or her productivity is measured by the

quality and timeliness of those decisions. Accepting the fact that management decisions are at least as important to organization productivity as the automation of lower level work leads to the conclusion that managerial productivity is worthy of a great deal of time, attention, and money. From this view, there is need for managers to analyze data in order to see important trends and relationships. Generally, this can be accomplished by the utilization of multidimensional analysis within an OLAP processing mode. For these multidimensional representations to accomplish anything, they must communicate relationships–contrasts, comparisons, and trends. These are important elements of which decisions are made. The right kind of multidimensional analysis via computerized graphics overcomes the inability of tabular data to represent relationships. In view of these factors, multidimensional analysis is an important means to improve managerial productivity and, at the same time, support decision making within an OLAP environment.

IV. THE ESSENTIALS OF ON-LINE ANALYTICAL PROCESSING

The essentials of OLAP cover a wide range of areas that will be seen below. They include: (1)accent on multidimensional analysis, (2)utilization of OLAP servers, (3)accent on multidimensional databases and tie-in with other databases, (4)employment of OLAP software, and (5)bringing OLAP to the Web. All of these essentials are covered below.

Initially, it should be noted that the decision-support process should be *open, scalable,* and *integrated* for a typical company. The *first* term (open) refers to the use of standards of interoperability of company databases and data warehouses. In this way, current technology investment and infrastructure can be leveraged to a company's full advantage. The *second* term (scalable) refers to the ability of the system to expand and handle additional work. The *third* term (integrated) means the OLAP solution should be comprehensive, that is, one place for query, analysis, and reporting.

A. Accent on Multidimensional Analysis

A most important characteristic of OLAP is multidimensional analysis, that is, analysis that goes beyond the traditional two-dimensional analysis. Essentially, multidimensional analysis represents an important method for leveraging the contents of an organization's production data and other data stored in company databases and data warehouses because it allows users to look at different dimensions of the same data. According to recent research, users need OLAP if they spend more than 20% of their time analyzing data and the data is compared across more than two dimensions (such as business units, products, industries, market segments, distribution channels, and so forth). As such, OLAP makes it easier to do analyses that cross department and even corporate boundaries.

Another way of viewing OLAP is getting a typical company out of the custom-report-writing business and into the data-cube-server building business. An OLAP data structure can be thought of as a *Rubik's Cube* of data that users can twist and twirl in different ways to work through "what-if" and "what happened" scenarios to get at the *whys* of the situation. Because the Rubik's Cube is a good way of visualizing a typical multidimensional database, its structure is composed of cubes of data and, in turn, cubes within cubes of data. Each side of the cube is considered a dimension representing a category, such as product, region, sales, and time. In addition, each cell within the multidimensional structure has aggregated data relating elements along each of the dimensions. For example, a single cell may contain the total sales for a given product in a region for a quarter. To better visualize a multidimensional cube, reference can be made to Fig. 1, where a typical relational-flat table has been restated in the form of a multidimensional cube. In the (a) part, only the flat table for the first product is shown. Comparable tables need to be stated for the other four products. As can be seen, a multidimensional cube is a more efficient way to store data as well as retrieve data than from a relational-flat table, that is, a file.

Typically, multidimensional databases provide users with consolidation, drill-down, and slice-and-dice capabilities. *Consolidation* allows users to combine data into ever-larger aggregations, such as rolling sales office data into district data into regional data. *Drill down,* which is the opposite of consolidation, allows users to seek increasingly detailed information on a particular topic. In addition, *slice-and-dice* provides the capability to look at the database from different viewpoints, such as showing sales of all products within a single region and sales of a single product across all regions. Going one step further, multidimensional databases are capable of storing data in compressed form, thereby making it possible to analyze large amounts of data while minimizing physical storage requirements. The systems dynamically separate so-called dense data, where data exists for a large percentage of dimension cells, from sparse data, or dimensions where a significant portion of the cells is

a

Product	Region	Quarter	Sales in Units
P1	R1	Q1	12,405
P1	R1	Q2	13,057
P1	R1	Q3	13,275
P1	R1	Q4	15,125
P1	R2	Q1	14,460
P1	R2	Q2	16,106
P1	R2	Q3	17,251
P1	R2	Q4	18,210
P1	R3	Q1	12,490
P1	R3	Q2	13,976
P1	R3	Q3	14,015
P1	R3	Q4	16,782

Note: to be comparable to (b), flat files must be added for Products 2 through 5

b

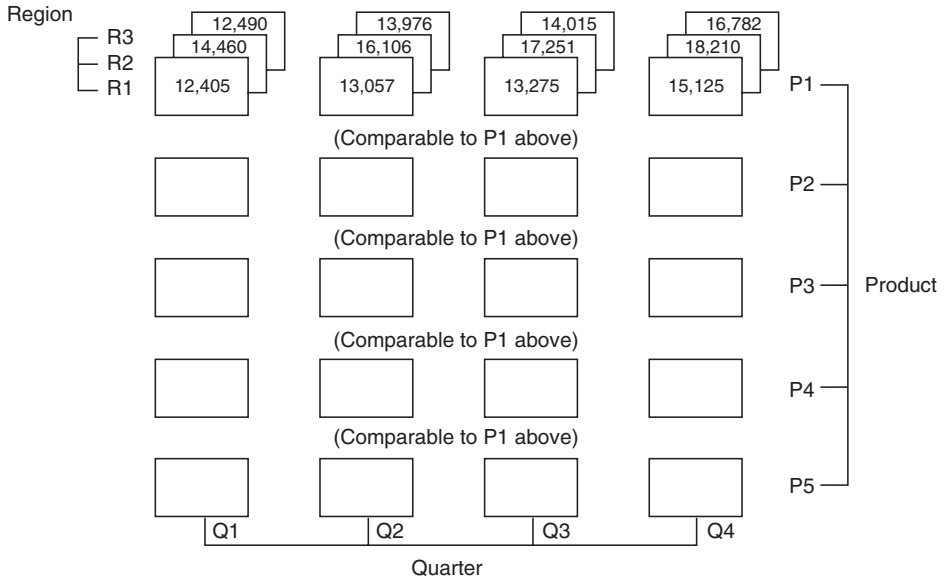


Figure 1 Two data models: (a) relational-flat table (File) for one product only, and (b) multidimensional cube for five products.

empty. Overall, multidimensional databases make it easier for users to obtain their desired information and knowledge within an OLAP mode.

Within an OLAP environment, there is need to perform dictionary definition and maintenance as well as mapping flat files or relational columns to dimensions and measures. Although this may sound like a lot of work, managing one data cube beats writing a number of custom reports. Currently, vendors provide administrative tools to get the data into the cubes in the first place, in the proper form, and on a regular basis. It should be noted that some of these data cubes are "real" while others are "virtual" or "open." That is, in some cases, data are replicated from the warehouse or other data source into the cube; in others, a metadata structure is built full of pointers that runs directly against a standard relational warehouse. In either case, there is need to design and maintain it. Generally, the real cubes are more limited in size but run much faster. Most real cubes can support at least 1 or 2 GB structures, while a few claim up to 100 GB capacity. Cube size grows rapidly as the user adds dimensions because of all the cross-dimensional links. Real cubes have something of

an advantage for financial analysis, which generally has fewer empty spots in the matrix. On the other hand, the virtual ones are able to handle sales and marketing analysis, which, generally, goes against larger, more spottily populated data sets. Virtual cubes have pointers, not data, so sparsity is not so much of a problem.

B. Utilization of OLAP Servers

Although data are found somewhere in a company's scattered databases, this does not mean that the user can get at it or that it can answer the toughest questions. From this view, there is a need for OLAP servers that can get the data out and make it mean something. Important criteria to consider when evaluating OLAP servers are the architecture, data manipulation, ad hoc reporting, links to back-end data, and security. Needless to say, it pays to keep an open mind.

Currently, OLAP servers are multidimensional, but to varying degrees. In addition, OLAP servers traffic in aggregated data. Because an important part of an OLAP system is the capability of drilling down through the cube, the most flexible products let the user drill in any direction, skipping levels as one drills, and drilling along pathways that have not been predefined. Thus, there is a need for seamless integration with transaction-processing databases.

It is also necessary to look at the degree of multiuser support, and the ability to do time-series analysis—both trademarks of the best OLAP servers. In a similar manner, security, especially user authority, is necessary with OLAP. Many of these products let the user read and write to the cube. That can create a problem if the products do not give the user tools to control read-and-write access. The user needs tailoring and flexibility as well as support for multiple hierarchies for the same entities. Overall, OLAP servers represent an approach to answer the *whys* of users' questions in a timely manner.

C. Accent on Multidimensional Databases and Tie-In to Other Databases

Typically, most companies have all the data they need for OLAP in their databases or data warehouses. However, they find it difficult to get at the data since it is necessary to deal with data in multidimensional space. To assist users in getting at the data, OLAP applications typically make use of three different types of database structures: (1) multidimensional, (2) relational, and (3) specialized relational databases, which are noted below.

The *first* type of database is multidimensional. Its essential characteristic centers on high-performance databases, capable of handling more than five dimensions, and typically used by a smaller number of users, as in a single department. This is the mainstay of OLAP. The *second* type is relational, which requires the database to be redesigned to achieve good performance. However, performance declines when more than five dimensions are implemented, and is typically used by a larger number of users across multiple departments. The *third* type of database is the specialized relational which requires a front-end OLAP tool. It is designed for decision support and data warehousing. Some products require that the database be redesigned since they still treat it as relational.

D. Employment of OLAP Software

Typically, OLAP software centers on multidimensional analysis that divides data into the important dimensions used to manage the business. For example, dimensions for marketing applications might include products, market areas, distribution channels, and time periods. In turn, these dimensions can be used to describe each data point in the database, such as selling price, units sold, total revenue, and market share. Dimensions can be further described by attributes, such as location, size, or year. Attributes also can describe hierarchies within a dimension, even overlapping and inconsistent hierarchies. From an end-user's perspective, multidimensional analysis provides for the selection, analysis, summarization, and reporting by dimensions and attributes within dimensions. Current spreadsheet, database, and reporting-tool vendors are offering simplistic multidimensional tools. However, some of these vendors and others are offering more sophisticated tools. That is, all multidimensional engines are not created equal. There is a need to look for engines that can work with any data warehouse, use metadata, and can use any PC development tools for the user interface. Not only should the engine be robust to support an unlimited number of dimensions, custom groupings, and automatic and custom calculations, but also the engine needs to reside on a server large enough to support hundreds of users, large volumes of data, and intensive processing.

Currently, OLAP software vendors include the following: Brio Technology, Business Objects, Comshare Inc., Hyperion Solutions Corporation, Microsoft, Oracle Corporation, Pilot Software, and SAS Institute,

Inc. Although these vendors offer operational OLAP software, a number of other vendors offer software that can be used. Other software packages include: business graphics packages, electronic spreadsheets, financial planning languages, and statistical analysis packages. Some of these software packages can be used as is or in conjunction with other OLAP software to provide users with some type of multidimensional analysis.

The capabilities of OLAP software today extend beyond what was originally intended. For example, ever since Microsoft Corporation released SQL Server 7.0, many companies have jumped at the opportunity to develop products that further leverage the relational database management system's capabilities. OLAP@Work Inc.(now a part of Business Objects) has provided users of Microsoft Excel with tools to leverage its OLAP client (better known as Pivot tables). Now the company has branched out to cover all of Microsoft Office with its OLAP@Work for Office. The software works with both Office 97 and Office 2000 to enhance the capabilities of SQL Server 7.0 so that all of the applications in Office transform into OLAP clients. Previously, a user would have only been able to use Excel Pivot Tables as an OLAP client. OLAP@WORK for Office makes the entire suite of Office components, capable of OLAP reporting and analysis. The product extends beyond the Excel product by embedding OLAP reporting and analysis into applications that come with Office. A user can now do OLAP reporting in any of these applications. With OLAP@Work for Office, the user can use Microsoft Word, Digital Dashboards, PowerPoint, Excel, Internet Explorer, and Outlook as an OLAP client.

E. Bringing OLAP to the Web

Today, there is need to extend the capabilities of OLAP beyond the company's boundaries to the Web. For example, Hyperion Solutions Corporation has partnered with vendors of data mining and data mart solutions. Based on agreements with Engage Technology Inc. and WebTrends Corporation, Hyperion is now bringing OLAP to the Web. Engage Technology Inc. is a provider of on-line marketing solutions and WebTrends Corporation is an organization that specializes in web site statistical reporting. Hyperion Solutions Corporation has combined its e-business Analysis Suite with Engage's Profile Server, a dynamic audience-profiling application that provides services for on-line marketers. The OLAP specialist has integrated its Essbase OLAP Server and Wired for OLAP products with WebTrends' CommerceTrends eBusiness intelligence solution, which is a web-based data warehousing technology.

It is no longer sufficient for on-line companies to simply log web-site clicks or collect information about customers. In today's environment, E-businesses need to combine reports and other statistical information and knowledge with the detailed analytical capabilities provided by an OLAP solution. In the past, E-business initiatives used to be valued by the number of clicks that people were getting. However, companies need to show accountability for their E-business initiatives in financial terms. The bottom line to these agreements is that while in the past users may have been satisfied with or accustomed to simple query or reporting solutions, now they want to drill down and do advanced analysis or do other things that only an OLAP solution can provide. In the case of Hyperion's partnership with Engage, the combination of the two companies' efforts should result in a solution that will help E-businesses get a better handle on customer behavior, assist E-business marketing organizations by better tailoring marketing campaigns for customers, and provide superior overall customer service. Through Hyperion's partnership with WebTrends, the combination of OLAP technologies produces a combined solution that can capture, analyze, and understand the behavior of web site visitors.

V. APPLICATIONS THAT LEND THEMSELVES TO AN OLAP OPERATING MODE

Typically, there are many areas that lend themselves to an OLAP operating mode. Among the major areas are: corporate planning, marketing, manufacturing, finance and accounting, and human resources. In turn, these areas can be broken down by their subparts. In the area of *corporate planning,* a thorough analysis of strategic planning information and knowledge for the short-, medium-, and long-range can produce important input for corporate planners. In the area of *marketing,* on-line analytical processing can be helpful to marketing managers and personnel in market planning, sales forecasting, market research, sales-order processing, sales analysis, advertising, product pricing, and physical distribution. For the area of *manufacturing,* viable applications abound for purchasing, production planning and execution, supply operations, inventories, and total quality management (TQM). The area of *finance and accounting* includes budgeting, financial statement analysis, cost accounting, accounts receivables and payables, source and

application of funds, and even payroll. Lastly, in the area of *human resources,* OLAP can be applied to human resource planning, personnel selection and placement, wage-and-salary administration, and training. Due to space limitations, only the area of sales analysis is presented below.

VI. AN OLAP APPROACH TO SALES ANALYSES

In a business environment characterized by shrinking profit margins, global markets, and faster product cycles, companies are spending millions of dollars annually on their marketing efforts, including product promotions. The demand to produce maximum results from these promotions is phenomenal. Success means that companies must focus constantly on identifying market changes. Their mission is to search the vast amounts of corporate data for answers that enable their company to make intelligent marketing decisions. Sales and marketing managers have need of a system that delivers a wide range of sales information and knowledge to their desks every day, which allows them to rank products, analyze channel performance, or create quickly ad hoc views of sales by account or region. To be useful to the manager, the system must be flexible, timely, and intuitive. Such an approach is possible within an OLAP environment.

In the areas of forecasting and pricing, for example, marketing managers have great impact on the needed input for these areas. However, they should go a step further by maintaining control over these forecasts and prices. The forecasted sales on the corporate database or data warehouse allow marketing managers to compare projected product sales to actual sales activity (updated on a daily basis) for some specific time period. To make comparisons of actual to forecasted sales on a periodic basis (say daily, weekly, or monthly), it is helpful for marketing managers to utilize a multidimensional-analysis approach to sales. It should be noted that there is need to go beyond sales analysis by developing appropriate analyses to evaluate specific products and their performance over time. This will lead to further analysis that can result in the need for new products and services. Also, this analysis could lead to assistance in the development of a revised marketing strategy.

A. Sales Analysis by Geographical Areas to Identify Market Changes

To analyze sales by geographical areas in order to identify market changes, a three-part level of inquiry formats can be utilized, that is, (1) total geographical sales, (2) detailed geographical sales, and (3) exception geographical sales. In the process of doing so, marketing managers can manipulate the data by looking for future trends; performing audits of the sales data; and calculating totals, averages, changes, variances, or ratios. At the highest level of marketing analysis, *total geographical sales,* is the overall sales performance for the current year as well as past years. This level of marketing inquiry can be extended to include an overall performance summary that describes the sales budget and actual month-to-date, last month-to-date, year-to-date, and last year-to-date sales by geographical areas. If a given geographical-sales performance is deficient compared to the budgeted figures, control can then be transferred to successive levels of detailed reports relative to the area of interest. Also, projected total sales can be generated by geographical areas.

At the second level of inquiry, *detailed geographical sales,* allows for optional levels of detail to be obtained. Normally, the inquiries requested of this structure are triggered for further analysis by the total sales inquiry. As with the first level, data are available for the same time periods and by the same sales categories at the detailed level. Also, this inquiry format is programmed to enable the sales managers to format report structures because they might be interested in certain sales ratios or sales-to-expense ratios. An example of detailed geographical sales is found in Fig. 2, where the total sales history for one geographical area of the United States over the last five years is shown by months. Although not shown sales data can be ranked by months and years and pie charts can be used to show the best and worst sales amounts over the five years for one geographical area. Comparable analyses can be developed for the other geographical areas of the United States plus other geographical areas of the world. In all cases, the focus of the analysis is on the highest sales periods for one geographical area versus other geographical areas. Such an analysis might help determine where goods could be shipped from one part of the world to another to smooth out production flows in the many manufacturing facilities of the world.

Lastly, at the third level of inquiry, *exception geographical sales,* is generally more complex and flexible than the first two. The exception inquiry is structured to highlight a certain condition or conditions. A typical inquiry might be: "Which products in a certain geographical area of the United States are above 110% of forecast sales and below 90% of forecast sales for the current month?" or "Which salespersons in a certain geographical area of the United States are below their sales quotas for the current month?" This in-

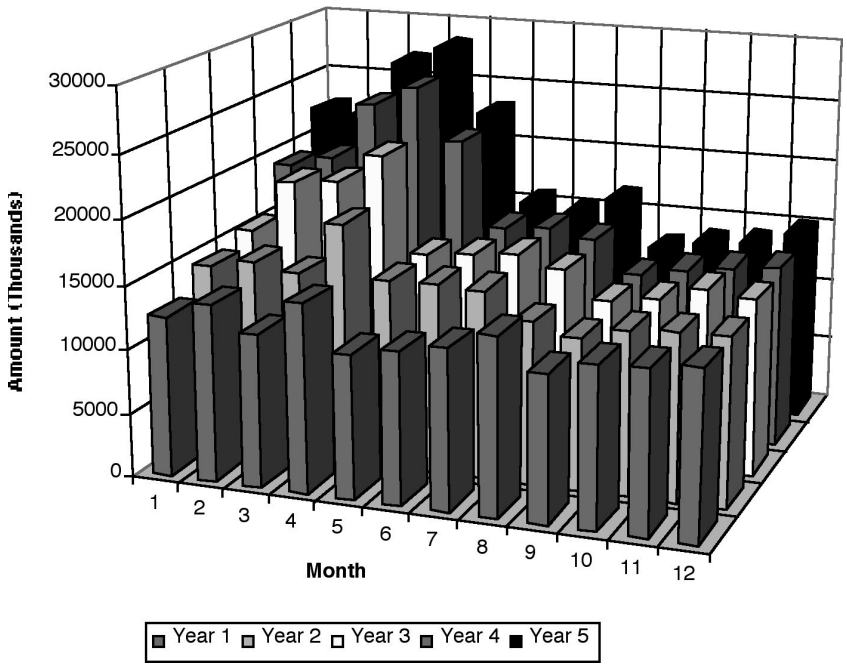


Figure 2 Total sales history by months for one geographical area of the United States over the last five years.

quiry level provides not only information to individual sales managers about the effectiveness or lack of it in terms of the company's marketing effort, but also provides a forum for a group discussion by marketing managers such that each geographical-sales area of the United States can be compared to other areas.

B. Sales Analysis to Evaluate Specific Products

Building upon the example given above using multidimensional analysis, the important feature of drill down can be illustrated for a major retailer. Just as the first level of inquiry (total geographical sales) and the second level of inquiry (detailed geographical sales) were discussed and illustrated for the second level of drill down, the third level (exception geographical sales) can also be an important means of sales analysis. As a starting point for this major retailer, a sales analysis grid can be developed that centers on the following six matrix roles:

1. Flagship—high markup, high volume
2. Cash machine—high markup, medium volume
3. Maintain growth—high markup, low volume
4. Core traffic—low markup, high volume
5. Rehab—low markup, medium volume
6. Under fire—low markup, low volume

It should be noted that where matrix roles have average markup for high, medium, and low volume, they have been placed in one of the above six matrix roles to simplify the evaluation process.

To better understand this evaluation process, reference can be made to Table I where the *first level* of inquiry is by department for one month in terms of sales, percent of total sales, gross margin percent, and role (1 through 6 per the above). A thorough examination of this figure indicates that there are seven product-commodities that need a thorough investigation as to the products and the rationale for being "under fire," i.e., very poor performance. Similarly, there is need to investigate nine product commodities that have need of "rehab," i.e., why they are marginal products that need to be reevaluated. Referring to the under fire product commodities, a drill down to the *second level* of inquiry is found in Table II for commodity 138—Feminine Hygiene. This breakdown indicates that all subcommodities are not problem ones, but only one subcommodity is, i.e., 62014, under fire.

Drilling down even further to the *third level* of inquiry per Table III for subcommodity 62012 indicates that ten out of eleven regions are "core traffic" and one region is "flagship." A similar type breakdown could have been developed for other subcommodities per Table II. Although not shown, it is recommended subcommodity 62014 be investigated further, since it is un-

Table I Total Sales for One Month of One Department That Centers on Matrix Roles for a Major Retailer (First Level of Inquiry)

Commodity Category Roles — All Regions
Department 03 Drug GM
Date Range: XX/XX/2000—XX/XX/2000
Average GM%: 29.53

Commodity	Description	Sales	% of Total	GM%	Role
206	Charcoal and lighter fluid	$14,912,984	1.35	29.19	Under fire
102	Soap—liquid and bar	$24,365,607	2.21	27.46	Under fire
229	Audio/video products	$23,229,980	2.11	20.05	Under fire
138	Feminine hygiene	$23,881,987	2.17	23.44	Under fire
86	Baby foods	$23,328,045	2.11	13.85	Under fire
169	Shaving care products	$21,165,790	1.92	28.32	Under fire
182	Dietary aid products	$18,868,089	1.71	19.84	Under fire
123	Tobacco other	$5,311,705	0.48	28.34	Rehab
243	Postal	$130,669	0.01	26.18	Rehab
170	Smoking cessations	$4,581,469	0.42	23	Rehab
141	No comm description	$189,049	0.02	17.78	Rehab
193	Fragrances	$356	0	28.09	Rehab
242	Personal care appliances	$578,424	0.05	29.04	Rehab
247	Underwear/outerwear	$1,379,374	0.13	26.21	Rehab
245	Continuities	$4,725,672	0.43	26.73	Rehab
241	Christmas seasonal	$466,340	0.04	−25.29	Rehab
224	Insecticides	$5,845,248	0.53	37.64	Maintain grow
221	Hosiery/socks	$8,583,438	0.78	41.65	Maintain grow
177	First aid products	$12,720,628	1.15	40.97	Maintain grow
220	Batteries	$12,964,210	1.18	36.93	Maintain grow
165	Hair care accessories	$4,248,572	0.39	56.3	Maintain grow
185	Suntan	$5,505,770	0.5	30.68	Maintain grow
161	Baby HBC	$13,652,962	1.24	32.21	Maintain grow
174	Sinus and allergy	$10,044,230	0.91	38.69	Maintain grow
228	Coffee filters, mugs, thermos	$3,921,967	0.36	49.38	Maintain grow
223	Sewing	$521,978	0.05	54.46	Maintain grow
210	Brooms and mops	$12,278,916	1.11	48.05	Maintain grow
82	Marshmallows	$7	0	42.86	Maintain grow
176	Laxatives	$9,646,648	0.87	37.7	Maintain grow
179	Foot care products	$4,650,937	0.42	41.42	Maintain grow
227	Portable electric appliances	$381,201	0.03	30.96	Maintain grow
215	Watches/calculators/lobby	$6,750,093	0.61	40.95	Maintain grow
237	Infant care products	$4,425,101	0.4	43.62	Maintain grow
178	Eye and ear care products	$9,534,843	0.86	33.01	Maintain grow
159	Coupon	$7,618,340	0.69	33.28	Maintain grow
225	Kitchen gadgets	$6,946,052	0.63	57.12	Maintain grow
233	Toys and games	$7,180,827	0.65	34.28	Maintain grow
218	Ironing and chemicals	$4,622,275	0.42	45.39	Maintain grow
219	Electrical supplies	$11,823,161	1.07	52.66	Maintain grow

(continues)

Table I *(continued)*

Commodity	Description	Sales	% of Total	GM%	Role
240	Fall and winter seasonal	$500,661	0.05	31.47	Maintain grow
208	Glassware and dinnerware	$792,427	0.07	37.73	Maintain grow
231	Contact paper	$141,543	0.01	58.92	Maintain grow
167	Ethnic personal care	$1,907,814	0.17	32.02	Maintain grow
209	Cookware and bakeware	$3,593,431	0.33	45.03	Maintain grow
181	Home health care	$3,032,301	0.27	43.39	Maintain grow
207	Home freezing and canning supply	$1,386,518	0.13	37.22	Maintain grow
122	Cigars	$2,518,982	0.23	30.01	Maintain grow
191	Bath	$48	0	41.67	Maintain grow
216	Home furnishings	$1,708,775	0.15	39.59	Maintain grow
183	Contraceptives	$1,990,928	0.18	46.08	Maintain grow
217	Hardware supplies	$3,244,203	0.29	52.36	Maintain grow
230	Automotive products	$2,853,300	0.26	36.27	Maintain grow
211	Plastic housewares	$4,060,945	0.37	41.55	Maintain grow
226	Disposible foilware	$3,919,971	0.36	70.8	Maintain grow
137	No comm description	$163	0	36.81	Maintain grow
232	Shoe care	$1,188,382	0.11	58.8	Maintain grow
140	No comm description	$1,884	0	36.41	Maintain grow
135	No comm description	$1,421	0	38.78	Maintain grow
194	Cosmetic accessories	$11,023	0	51.92	Maintain grow
186	Miscellaneous HBC	$67,578	0.01	34.98	Maintain grow
126	Fireworks	$405,098	0.04	32.06	Maintain grow
239	Lawn and garden shop	$1,903,832	0.17	34.43	Maintain grow
222	Domestic goods	$1,289,578	0.12	35.78	Maintain grow
202	Candy—packaged	$82,038,439	7.44	30.67	Flagship
246	Magazine	$45,467,899	4.12	34.69	Flagship
172	Analgesics	$31,632,098	2.87	33.6	Flagship
201	Candy—Checklane	$26,958,429	2.44	36.64	Flagship
235	Greeting cards/wrap/party sply	$57,688,489	5.23	49.35	Flagship
136	Diapers and disposables	$57,631,626	5.22	13.56	Core traffic
171	Oral hygiene products	$40,671,307	3.69	29.31	Core traffic
121	Cigarettes	$121,662,574	11.03	12.96	Core traffic
166	Hair care products	$40,825,510	3.7	29.51	Core traffic
94	Infant formula	$60,759,331	5.51	8.54	Core traffic
175	Antacids	$14,111,487	1.28	32.62	Cash machine
238	Spring/summer seasonal	$18,060,425	1.64	36.37	Cash machine
234	Stationary and school supplies	$14,047,479	1.27	51.75	Cash machine
214	Film and camera products	$24,346,935	2.21	34.6	Cash machine
163	Deodorants	$18,134,680	1.64	29.61	Cash machine
236	Bookstore	$19,076,388	1.73	30.44	Cash machine
173	Cold and flu	$20,830,209	1.89	37.78	Cash machine
180	Vitamins	$24,075,146	2.18	39.71	Cash machine
162	Hand/body/facial products	$17,496,887	1.59	33.35	Cash machine

Table II Total Sales for One Month—Commodity 138 Feminine Hygiene Products for a Major Retailer (Second Level of Inquiry)

Commodity Category Roles — All Regions
Department 03 Drug GM
Commodity: 138 Feminine hygiene
Date Range: XX/XX/2000—XX/XX/2000
Average GM%: 23.44

Sub Commodity	Description	Sales	% of Total	GM%	Role
	Feminine hygiene	$23,881,987	2.17	23.44	Under fire
62012	Fem. hygn. napkins	$12,085,078	50.6	21.02	Core traffic
62014	Fem. hygn. tampons	$8,160,391	34.17	22.69	Under fire
62010	Fem. yeast treatments	$2,217,207	9.28	32.72	Maintain grow
62016	Fem. hygn. douches	$784,196	3.28	33.28	Maintain grow
62024	Misc. feminine hygiene	$293,391	1.23	32.9	Maintain grow
62018	Fem. hygn. deodorants	$280,729	1.18	36.82	Maintain grow
62017	Fem. hygn. medicated douches	$60,995	0.26	32.62	Maintain grow

der fire. The reader should be aware that the inability to raise gross margins does not mean that the product should be taken off the market. Otherwise the company's customers typically will go elsewhere to buy the product plus many others if they are not available. The net result could be a large number of lost customers. From another perspective, depending on the product, it may be far better for a retailer to accept lower gross margin percentages for sales volume leaders in order to increase overall profits for its operations.

Table III Total Sales for One Month Feminine Hygiene Products Subcommodity 62012 for a Major Retailer (Third Level of Inquiry)

Commodity Category Roles — All Regions
Department 03 Drug GM
Commodity: 138 Fem. hygn. napkins
Date Range: XX/XX/2000—XX/XX/2000
Average GM%: 23.44

Region	Subcommodity	Description	Sales	% of Total	GM%	Role
		Fem. hygn. napkins	$12,085,078	50.6	21.02	Core traffic
1	62012	Fem. hygn. napkins	$1,842,624	46.96	19.16	Core traffic
2	62012	Fem. hygn. napkins	$1,305,721	51.57	15.26	Core traffic
3	62012	Fem. hygn. napkins	$1,552,080	49.68	24.35	Core traffic
4	62012	Fem. hygn. napkins	$1,152,561	56.41	22.66	Core traffic
5	62012	Fem. hygn. napkins	$1,154,221	54.58	17.21	Core traffic
6	62012	Fem. hygn. napkins	$876,356	49.16	19.48	Core traffic
7	62012	Fem. hygn. napkins	$660,540	49.75	21.05	Core traffic
8	62012	Fem. hygn. napkins	$808,502	49.08	24.67	Core traffic
9	62012	Fem. hygn. napkins	$806,658	49.06	22.24	Core traffic
10	62012	Fem. hygn. napkins	$1,230,893	52.46	24.63	Flagship
11	62012	Fem. hygn. napkins	$694,922	49.74	22.87	Core traffic

Although the above analyses have yielded some interesting results, Table I could have ranked by dollar sales or by gross margin percent. Also, although not shown, the data could have been ranked by sales units. However, it was more convenient for the manager to look at specific category roles and drill down further for getting a handle on those products that are in need of improvement for bottom line results. Additionally, this major retailer could have made comparisons at the highest level to other large retail types, such as mass merchandisers, or drug chains. In turn, appropriate drill downs could be made to "those versus us" analyses. It should be noted that this major retailer cannot get data from competing retailers legally although such analyses would prove to be quite interesting to the company's management. From another perspective, a slice-and-dice approach could have been used to better understand the data from different viewpoints.

C. Appropriate Analyses to Develop an Overall Marketing Strategy

In the development of an overall marketing strategy, multidimensional analysis can be helpful in getting a handle initially on who the customer really is. Marketing managers need not only bring insights and expertise to determine who the customer is, but also must recognize that the customer is the most important entity in the distribution channel. As such, marketing managers need to consider the present needs and future wants of their customers in every step of the distribution channel. They need to spend a typical day in the life of their small-, medium-, and large-size customers from the standpoint of meeting their needs today and their wants tomorrow. For example, even though the company ships all of its orders 95% on the average, the customer views it differently since the order must be 100% complete before they can introduce the item into production. This is where OLAP comes into play since multidimensional analysis can be used to fully understand the customer's viewpoint. In the example, multidimensional analysis of three important variables (percent of shipments received on time, percent of shipments received late, and days late for remaining items) can give marketing managers a better perception of its operations through the eyes of a typical customer. The results may not be very positive. However, OLAP is capable of presenting the true picture to marketing managers—good, bad, or indifferent.

From another perspective, OLAP can assist marketing managers in developing an overall marketing strategy by concentrating on customer satisfaction. It is a generally known fact that retaining existing customers is usually much more profitable than focusing on acquiring new customers. Even though many companies state that customers are important, often company employees do not rate customer satisfaction as a top priority of their company. Many times, there is a large gap between what a company says it does and what its customers perceive it as doing. Hence, multidimensional analysis can be used to get a handle on customer satisfaction or lack thereof.

Still other ways to employ OLAP can be found in the marketing efforts of well-known companies. For example, the Procter & Gamble Company has undertaken a global initiative to make its marketing efforts more efficient. Procter & Gamble's effort to find more efficient ways of marketing its wide array of products can already be seen in the Cincinnati-based company's United States operations. In the United States, Procter & Gamble is building closer ties to a reduced number of outside companies that do everything from art to package design to printing materials for promotional campaigns. In essence, Procter & Gamble is looking at its whole system of developing and executing its marketing programs with its suppliers to find ways to improve flexibility, save time, and get the results it seeks at a lower cost.

VII. LATEST VIEW OF ON-LINE ANALYTICAL PROCESSING

In the preceding discussion, OLAP was seen as a means that allows decision makers to interpret information and knowledge found in large volumes of data by employing advanced data storage, aggregation, and presentation techniques to get at the *whys* of a company's operations. Going a step further, current work is focusing on predicting more accurately important marketing and sales trends that will impact the company's future. For example, *why* are certain products and services selling so well while others are not? There is need to make important associations and comparisons with competing products and services as well as other significant marketing efforts. Similarly *why* do certain company products and services fair especially well abroad and do only average in this country? Possibly partnering with others could be a viable alternative to the current lackluster performance. Answers to these marketing questions as well as other nonmarketing ones will allow a company's decision makers the capability to meet and, possibly, beat competition.

To reach this current level of important analyses, companies are turning to web-based analytical tools,

sometimes referred to as business intelligence tools, that include the functionality to slice and dice data and information, navigate easily through the data warehouse, employ data-mining capabilities, as well as utilize appropriate security measures. In addition to traditional financial information, the key area of usage for web-based business intelligence tools is studying markets, sales, logistics, and trends. Business intelligence tools, for example, can help glean such information as which third-party logistics providers they have spent sufficient money with in order to see if they can renegotiate a contract or to determine how often they are delivered on time. Equally important, companies are opening up aspects of their business to their suppliers and customers so they have access to information and knowledge about their relationships. It is recognized that this new paradigm for analyzing information and knowledge is here today because so much is coming through the Web on a continuous basis. Thus, the current web-based business intelligence approach is now considered as a logical offspring of OLAP.

ACKNOWLEDGMENT

Portions of this article are taken with permission from *On-Line Analytical Processing Systems for Business* by Robert J. Thierauf, published in 1997 by Quorum Books, an imprint of Greenwood Publishing Group, Inc., Westport, CT.

SEE ALSO THE FOLLOWING ARTICLES

Data Mining • Data Warehousing and Data Marts • Decision-Making Approaches • End-User Computing Concepts • Internet, Overview • Marketing Information Systems • Productivity • SAS (Statistical Analysis System).

BIBLIOGRAPHY

Amo, W. C. (2000). *SQL server 7 OLAP developer's guide.* Foster City, CA: IDG Books.

Berson, A. (1997). *Data warehousing, data mining and OLAP.* New York: McGraw-Hill.

Freeze, W. S. (2000). *Unlocking OLAP with SQL server and excel 2000.* Foster City, CA: IDG Books.

Peterson, T. (2000). *Microsoft OLAP unleashed.* 2nd Ed. Indianapolis, IN: Sams.

Shumate, J. (2000). *A practical guide to microsoft OLAP server.* Reading, MA: Addison-Wesley.

Thierauf, R. J. (1997). *On-line analytical processing systems for business.* Westport, CT: Quorum Books.

Thomsen, E. (1999). *Microsoft OLAP solutions.* New York: John Wiley.

Thomsen, E. (1997). *OLAP solutions: Building multidimensional information systems.* New York: John Wiley.

Operating Systems

Hossein Bidgoli
California State University, Bakersfield

Andrew Prestage
Kern County Superintendent of Schools, Bakersfield, California

GLOSSARY

buffers Storage space used to temporarily store data while it is waiting to be transferred between two devices or between a device and a program.

dispatcher Manages the allocation of processor time slices required to perform program instructions.

file management A task performed by an operating system that allows the computer to keep track of where files are stored, manage directory structures, maintain file access privileges, and organize disk storage into files, directories, and subdirectories.

input/output (I/O) subsystem A task performed by an operating system that manages access to and use of all of the various peripherals that may be connected to a computer system.

kernel The core component of an operating system—responsible for resource allocation, memory, and task and disk management.

memory cache Makes a copy of data that already exist in another memory location and places the copy into a faster region of memory. Since access to cache memory is much faster than access to standard memory, the data held within the cache can be processed far more quickly.

memory management A task performed by an operating system responsible for moving programs and data back and forth between storage and memory.

multiprocessor system A computer system that contains more than one processor chip, or a computer system that is connected to other systems that contain a processor chip(s) via a local area network or Internet connection.

operating system A set of programs that controls and supervises computer hardware and software.

resource allocation A task performed by an operating system that ensures the efficient and effective control of the allocation of memory, disk, processor, and other resources of the computer among multiple competing (mutually exclusive) applications and user programs. It instructs the CPU, for example, on which job should be assigned to which printer or from which input device the data should be retrieved.

scheduler Determines which user programs will receive execution time from the processor.

shell A component of an operating system responsible for user interface.

single-processor system A computer system that contains (or that has access to the use of) only one processor chip.

time-sharing systems A type of computer processing that processes different jobs by dividing the computer processing time into slices or intervals. Each job utilizes the time interval allocated to it.

user interface A task performed by an operating system that provides commands, instructions, and features for an efficient and effective method for accessing the operating system. Its main responsibility is to make using the operating system a simple and easy to use process.

virtual memory A hard disk space that the computer treats as if it were RAM. When the computer runs out of RAM, it automatically switches to virtual memory to fulfill user requests and memory management tasks.

I. INTRODUCTION

This article defines an operating system and reviews its basic capabilities. Two major components of an operating system are examined including the shell and kernel and single-processor and multiprocessor systems are reviewed. To better understand the architecture of an operating system, various types of software used in a computer environment are reviewed and major tasks performed by a typical operating system are explored including file management, input/output (I/O) operations, memory management, resource allocation, and user interfaces. Several examples of popular operating systems are presented for personal computers (PCs), local area networks (LANs), and mainframes and criteria for selecting an operating system are given.

II. WHAT IS AN OPERATING SYSTEM?

An *operating system* (OS) is a set of programs that controls and supervises computer hardware and software. An OS provides an interface between a computer and a user. It increases computer efficiency by helping users share computer resources and by performing repetitive tasks that otherwise would be performed by the users. Figure 1 shows major resources managed by an OS.

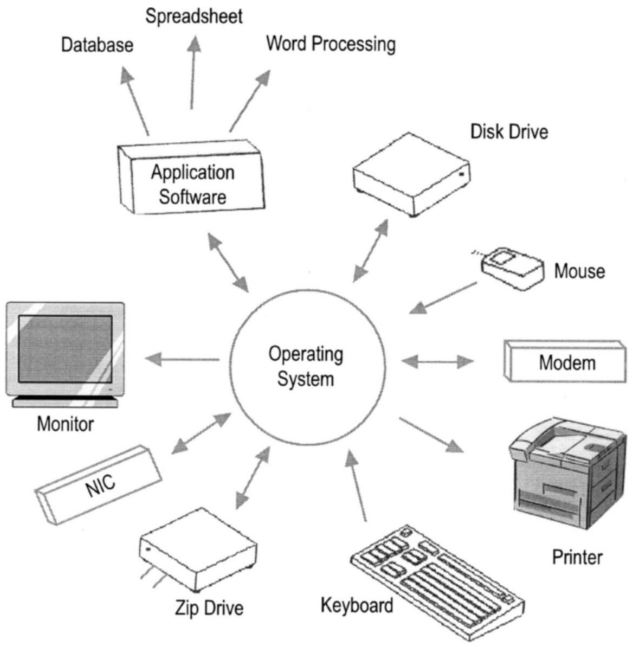


Figure 1 Major resources managed by an operating system.

Table I Popular OS Types

AIX (Advanced version of UNIX, developed by IBM)
DOS (Disk Operating System, developed by Microsoft)
Linux (an open architecture variant of UNIX, developed by Linus Torvalds)
Macintosh (developed by Apple Computer)
MVS (Multiple Virtual Storage, developed by IBM)
OS/2 (Operating System 2, developed by IBM)
OS/400 (Operating System developed by IBM for AS/400 minicomputers)
Unix (developed by Bell Laboratories)
VMS (Virtual Memory System, developed by Digital Equipment Corporation)
Windows 3.1, 95, 98, NT, ME, XP, 2000 (developed by Microsoft)
z/OS (developed by IBM for the z series of large servers, renamed from OS/390, which in turn evolved from the MVS operating system)

There are several types of operating systems available. Some are single tasking (performing one task at a time all the way through to completion) while others are capable of multitasking (performing more than one task at a time). Some are text-based or command-line oriented such, as is the case with DOS, while others are icon-based and offer graphical user interfaces such as Microsoft Windows. In the microcomputer environment the popular types of operating systems include DOS, Windows 3.x, Windows 95, Windows 98, Windows NT, Windows 2000, Windows ME, Windows XP, OS/2, Unix, Linux, Macintosh, and the Apple OS. As soon as a user turns on a PC or a workstation, the operating system takes over.

The majority of operating systems for microcomputers, such as MS-DOS, Apple PRODOS, and Apple Macintosh, are single-tasking operating systems. However, OS/2, developed by IBM, and different versions of the Unix and Microsoft Windows operating systems are capable of multitasking, with the ability to perform multiple operations simultaneously. Table I lists some of the popular operating systems on the market.

III. COMPONENTS OF AN OPERATING SYSTEM

A typical operation system includes two major components: the shell and kernel. In the following paragraphs we explain these two major components.

A. Shell

To work with a computer without the benefit of a friendly user interface would require that users memorize and use complex memory addresses and instructions. The responsibility of an efficient user interface, commonly referred to as a shell, is to make using the operating system a simple and easy to use process. The primary function of a shell is to perform and accommodate communications between the user and the operating system. The most common form of a shell employs a graphical user interface (GUI) such as OS/2 or Windows. Using a GUI shell, a series of icons are displayed on the screen and mouse clicks are used to execute user programs and carry out other user requests. Older shells offer a nongraphical, character-based interface. A character-based interface facilitates communication between the user and the operating system through the use of the keyboard and computer screen. Newer operating systems are programmed to include a variety of shell options, allowing the user to select the shell that most closely matches his or her individual preferences. Regardless of the shell selected, the core functions of the operating system do not change. The shell simply provides an interface between the user and the operating system that makes using the computer easier.

B. Kernel

While the shell defines the interface between the user and the operating system, the kernel defines the operating system's internal or core components. In other words, the shell can be thought of as the user interface between the user and the kernel. The components of the kernel are responsible for carrying out the most basic and fundamental functions of the operating system.

File management is among the major functions of a kernel. The ability to perform file management allows the computer to keep track of where files are stored, manage directory structures, maintain file access privileges, and organize disk storage into files, directories and subdirectories. It allows for addition, deletion, and modification of files and directories. It also allows users to organize files based on their size, date and/or time of creation and modification, name, and so forth. The ability to group files together into directories allows users to more easily manage the files on their storage devices. A directory can contain additional directories. For example, the WP directory can be created to contain all of the word processing files on the computer and can contain additional directories called WORK, PERSONAL, and SCHOOL, allowing the user to organize their various word processing files into logical categories.

Another important function performed by the kernel is the management of device drivers. Device drivers are software programs that tell the operating system how to access and work with various hardware components. For example, in order to use a CD-ROM drive, a CD-ROM device driver must be installed so that the computer can recognize and use the drive. Device drivers are created specifically for particular hardware or peripheral components. By using device drivers to control specific hardware components, the core operating system can be programmed generically without concern for the various hardware components that may ultimately be attached or installed into the computer.

Yet another component of an operating system is the memory manager. To be useful, a computer must be capable of doing more than just storing programs and data. A computer must also allow the user to retrieve programs and data out of secondary storage into main memory so that user processes of such programs and data can be accessed. After the computer is turned on, it must load the operating system into memory. Typically, the size of the operating system is kept small to maximize the amount of memory available to user processes such as applications and data files. Once the operating system is loaded, the user can then initiate processes that will in turn be loaded into memory, along with their associated data files. Moving programs and data back and forth from storage device to memory is the function of memory management. Many memory management schemes have been devised, usually optimized in one way or another to fit the particular hardware characteristics of the computer.

Two other important components of the kernel include the scheduler and the dispatcher. In the simplest terms, the scheduler determines which user programs will receive execution time from the processor while the dispatcher manages the allocation of processor time slices required to perform program instructions.

IV. TYPES OF OPERATING SYSTEMS

Early computer system architecture provided for only one microprocessor. However, as the computing industry evolved and developed more sophisticated processing requirements, the need for multiple processors within a single computer system became evident.

Systems with multiple processors could significantly increase a system's performance by allocating certain tasks to processors that were optimized to perform those tasks. For example, performance advantages can be achieved in engineering and scientific applications by allocating one chip to handle all computational operations, while another chip performs logic, data movement, and application processing operations. In so doing, application processing and calculation processing can occur in parallel, thereby significantly decreasing the amount of time required to accomplish the assigned tasks. Other multiple processor systems are capable of performing dynamic resource allocation by dividing large processing jobs into smaller ones and assigning the work to be performed to separate processors. Although the ability to perform dynamic resource allocation in systems with multiple processors introduces some overhead in the form of algorithms that allocate the processing work, efficient algorithms can perform these allocations during runtime and still achieve significant performance increases over single processor systems. Distributed (network) systems are becoming an increasingly prevalent example of partitioning processing tasks across multiple processors. Systems that are capable of dividing processing tasks and allocating the tasks to processors on remote systems can achieve tremendous performance gains through processing parallelism. The next two sections that follow provide further explanations of the difference between single- and multiprocessor systems.

A. Single-Processor Systems

In the 1940s and 1950s, computers were large, expensive, difficult to operate, and contained only one processor. Programs written to run on these computers were called jobs and were run one at a time using punched cards. Users had to load the cards into a peripheral device called a card reader and manually prepare the computer using switches prior to running a job. When a job was completed, it was unloaded from the card reader and the computer could be prepared to run the next job. For a number of reasons, this was not a very efficient process. For example, only one user could use the computer at a time. Other users had to wait in line for their turn to use the computer. The computer processor would sit idle between turns and while jobs were loaded and unloaded.

Eventually, companies began to use computer operators to help make more effective use of computer resources. The use of computer operators meant that users no longer had to run their own jobs. Users could submit their jobs for processing and pick up the results later. Through training and constant use, computer operators quickly became more proficient than regular users at using the computer. This helped increase the return on investment of the costly computer equipment, and in addition, helped increase system security by reducing the sheer number of people who had to have access to the computer.

In a rudimentary sense, the use of computer operators represented an early batch-processing environment. By submitting their jobs to a computer operator for later processing, jobs with similar setup requirements could be put together and run as a batch. This type of processing was ideal in environments where no decision making was necessary during program execution time. For example, payroll jobs where all of the input data were known and no further input was necessary during execution time were ideal for this type of processing. On the other hand, an airline reservation system was not well suited for this type of environment since it required real-time interaction and decision making on the part of the user during processing. Still, the fact that the computer contained only one processor meant that only one job could be run at a time.

To alleviate this problem, a capability known as time-sharing was developed. Time-sharing systems are operated by dividing the computer processing time into slices or intervals. The need for time-sharing systems resulted from the deployment of dumb terminals to permit multiple user access to the computer. The terminals were referred to as "dumb" since they did not contain an internal processor. Because the terminals did not contain a processor, they were relegated to serving as input/output devices, leaving all of the actual processing requirements to the central computer. Creation of terminals through which users could interact with the central computer led to a capability known as interactive processing. Using time slices, jobs could take turns accessing the processor, providing multiple users with real-time interactive access to the central computer. At the beginning of a time slice, a job would be loaded and processed. When that time slice was over, the job would be set aside and the next job would receive a turn. Although the ability of the processor to process multiple jobs concurrently created the illusion of simultaneous, real-time processing, in reality, only one job was running at any given time. Furthermore, time-sharing created a significant amount of overhead on the processor to be able to rotate multiple jobs, and problems sometimes occurred when multiple users would submit a job at

the same time. Nevertheless, time-sharing represented an improvement in the efficiency of computers. The development of time-sharing increased the efficiency of the computer by allowing more than one person to use the computer at a time and by increasing the efficient use of processor time. Time-sharing systems could perform adequately for up to as many as 30 concurrent users operating in real-time interactive processing mode. Time-sharing technology continues to be in use on single-processor systems today. Modern computers still take advantage of the computer's ability to divide processor time into slices to allow multiple tasks to be processed concurrently. The capability is commonly known as multitasking, creating an illusion that the computer is performing more than one task simultaneously.

B. Multiprocessor Systems

With the advent of the microcomputer in the early 1980s, users began to replace their terminals with more powerful workstations. The newer workstations each contained independent microprocessor and data storage components, allowing users to install applications and store data on the workstation. This capability did not replace the central computer. In fact, users still needed to connect to the central computer to access corporate financial and administrative information systems and their associated data. Users would log in to the central computer just as they always had, and during their log-in session their computer would be operating in terminal mode. However, the ability to load and use other locally stored applications residing on the user's workstation represented a significant step toward distributing power back to the user's control.

The concept of using many small computers to perform many of the tasks previously performed by a single large computer has become very attractive. In this manner, data storage, printing capabilities, and specific applications can be distributed throughout the network on different computers. Each user could now create and maintain complex spreadsheets, databases, and word processing documents. Eventually, users began to share data and information among each other. Simply being connected via terminals to a central computer was not sufficient. As networking technology grew, users began to connect their computers together to form networks. These networks represented a type of multiprocessor environment.

Network operating systems (NOS) such as Novell NetWare and Microsoft NT were developed to allow multiple computers to share resources. Still, the resources available within a network are not evenly distributed. For example, a single computer may be very busy performing a processing task while other computers are sitting idle. A more efficient use of the computers within the network would be to share the processing tasks through a technique known as load balancing, thereby accomplishing tasks more quickly and efficiently. By dividing complex tasks among multiple computers, the workload is more effectively distributed among all of the computers on the network. The process of breaking tasks down into subtasks for distribution among multiple machines is known as scaling. Research in the area of load balancing and task distribution in network environments continues and will almost certainly result in even more efficient processing environments.

However, this simple definition of a network operating system could be misleading because it implies that a NOS resides entirely on a single file, print, or communications server. Since servers provide centralized resources to client workstations that are connected within the local area network (LAN), a more comprehensive definition of a NOS must also take into account the client computers (workstations) that comprise the rest of the LAN. Taken together, the software components on client workstations along with the network software residing on server equipment are all integral components of a network operating system. Thus, a comprehensive definition of a NOS must conceptually take into account all of the separate network communications and operating system software residing on computers throughout a network. Stand-alone computer workstations do not have the necessary communications hardware and software required to be a member of a LAN. Special hardware, such as a network interface card (NIC) and specialized client software must be installed on client computers to enable communications with the server. The NIC provides a physical connection point to the network while the client software defines the rules to be followed when communicating as one of many members of a LAN. Major network operating system vendors have developed special rules, referred to as protocols, to enable member computers to communicate on the LAN. For example, computers that have Novell's IPX/SPX (Internetwork Packet Exchange/Sequenced Packet Exchange) protocol installed can communicate with any Novell servers on the network. The NetBEUI (NetBios Enhanced User Interface) protocol was originally designed by IBM for their LAN manager server and was later extended by Microsoft and Novell for use on their network platforms. More recently, network system vendors such as Microsoft and Novell have developed TCP/IP (Transmission

Control Protocol/Internet Protocol) versions of their client software applications.

Networks are only one example of a multiprocessor system. However, each individual computer still contains only one central processing unit. Taking the concepts of load balancing and scaling to the individual workstation level, many computers sold today contain multiple processors. Computers used in scientific, engineering, computer-assisted design (CAD), and simulation applications are often equipped with dual processors to enable more efficient processing. Complex tasks can be divided between several processors and therefore completed more quickly. File and application servers in network environments are often equipped with dual processors to be able to accommodate the needs of the many users who are connected and requesting services.

V. SOFTWARE CLASSIFICATION

To better understand the architecture of an operating system requires an understanding of the various types of software that can be found on a typical computer (see Fig. 2).

At the most fundamental level, software can be divided into two broad categories: application software and system software. Application software are associated with what users routinely work with on a daily basis to get their work accomplished. Examples of application software include spreadsheet, database, word processing, Internet browsers, and graphics software. System software are commonly associated with the efficient and reliable functioning of the computer system. Examples of system software include disk, file, and memory maintenance applications, and software that makes using the operating system simpler.

System software can be further subdivided into two additional categories: operating system software and utility software. Initially, the utility software market niche developed to "fill the gaps" in the missing or desired capabilities of operating systems, and an entire software industry has developed to meet the demand for utility software applications. Some of the utility software applications were developed to give users capabilities that were not available in the oper-

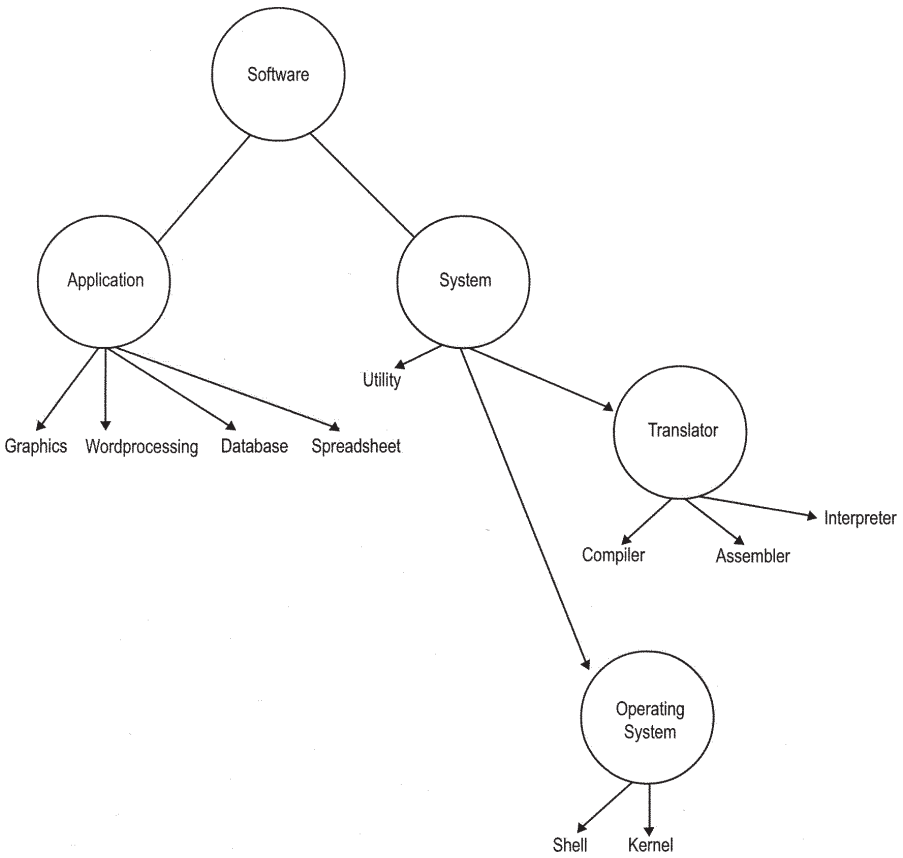


Figure 2 Major software used in a computing environment.

ating system. Other utility software applications were developed to make using the operating system simpler. Examples of utility software include installation programs, specialty software for maintaining efficient control over memory, files, and disk storage areas, and software that makes the operating system easier to use. Virus protection software can also be regarded as utility software in that while it does not directly serve the user's information processing needs, it is protecting the operating system, applications, and data from harm caused by malicious virus applications.

Other types of system software include compilers, interpreters, and assemblers.

A *compiler* is a program that translates a user program from source code into object code. If the source code is written in assembly language, the translator is called an assembler.

An *assembler* is a program or set of programs that takes basic computer instructions and converts them into an object code (a series of ones and zeros) that the CPU can use to perform its basic operations. Some computer programming languages use *interpreters,* which are similar to compilers; however, interpreters are slower and less sophisticated. The major difference between a compiler and an interpreter is that a compiler offers the user a choice between executing or saving the object code version of a program, while an interpreter does not. Another difference is that an interpreter converts each line of a source code program into machine language and executes it statement by statement. An interpreter does not wait for the entire program to be compiled prior to execution. Interpreters are commonly used when an interactive process occurs between a user and a computer. In such a case, a user receives immediate feedback.

VI. HOW DOES AN OPERATING SYSTEM WORK?

First, the computer must be started using a process known as boot strapping or booting the computer. Once the computer has been turned on, control is passed to a special read-only memory (ROM) chip that contains instructions for how to load the operating system. These instructions are burned onto the ROM chip and remain on the chip even when the machine is turned off. The instructions contained on the ROM chip are commonly referred to as the bootstrap. Typically, the bootstrap instructions cause the computer to check the diskette drive for the presence of any operating system files. If no diskette is found, the bootstrap instructions then default to the hard disk. However, if a diskette is inserted into the drive but does not contain the operating system files, an error message is generated and displayed on the screen indicating the operating system files are corrupted or missing. Once the operating system files are found, the bootstrap instructions cause the CPU to begin to transfer the files and instructions from storage device into memory. The CPU is considered volatile since it can only maintain the data and program instructions that are loaded into it while the computer is in use. The contents of the CPU are lost immediately as soon as power to the computer is shut off.

To understand how a computer coordinates the use of application software, utility software, and operating system software requires an understanding of what a process is. Put simply, a process can be described as the act of executing the instructions within a program. At any single point in time the computer can be said to be in a specific process state. Semantically, the process state represents the state of the computer at a single point in time and encompasses any pointers that are keeping track of the current processing position within the program, counters, and the values contained within the CPU registers and memory buffers. It is important to note that a single program can spawn multiple, concurrent processes, each competing for access to the CPU and operating independently to carry out the specific instruction. Processes must enable program execution without interfering with other application processes. For example, if two users were to execute the same server-based application at the same time, separate processes would be required to handle the requests so that the instructions being processed could be kept logically separate and out of each other's way. Processes compete for time slices and are allocated time slices so that they can accomplish their functions. The job of allocating time slices to the many processes that are competing for processor time is a basic function of the operating system. The operating system must ensure that processes have the memory, data, and CPU resources that they need, and that independent processes do not crash into each other.

An important task of the scheduler is to maintain a record of the processes that are currently active, allow new processes to start, and terminate processes that have finished. This work is accomplished through the use of a process table. Each process has an associated record or entry within the process table that keeps the scheduler informed as to what is going on at all times. It is also the job of the dispatcher to ensure that processes are actually executed. To do this, the dispatcher controls the allocation of time slices

(approximately 50 ms in length) to individual processes. The dispatcher ensures that each process gets a turn to be executed by the CPU through a capability known as process switching. To allow the dispatcher to redirect CPU resources to high priority processes, each process has an associated priority ranking. In this manner, high priority processes receive proportionally more CPU time than low priority processes. In a sense, the functions performed by the scheduler and the dispatchers are similar to the activities performed at a typical automotive repair shop. Cars are brought in to be serviced, repairs are completed, and the cars are taken away. A mechanic may be working on several cars at once, and each car may have an associated priority for the completion of its repair work. Working on car A, the mechanic may have to temporarily stop to wait for parts. During this time, repair work on car B can proceed. When the parts for car A arrive, the shop foreman may direct the mechanic to return to work on car A. The mechanic must stop working on car B to return to work on car A. When repairs to car A are complete, repair on car B can resume. The mechanic must quickly remember the work being performed to car B and pick up exactly where he or she left off.

VII. MAJOR TASKS PERFORMED BY A TYPICAL OPERATING SYSTEM

As briefly explained earlier, file management, input/output (I/O) operations, memory management, resource allocations, and user interfaces are among the primary tasks performed by an operating system.

A. File Management

As mentioned previously, file management is among the major functions of the operating system. The computer must be able to keep track of where files are stored, manage directory structures, maintain file access privileges, and organize disk storage into files, directories, and subdirectories. It must also allow for the addition, deletion, and modification of files and directories. By so doing, users can organize files based on their size, date, and/or time of creation and modification, name, and so forth. The ability to group files together into directories allows users to more easily manage the files on their storage devices. A directory can contain additional directories. For example, the WP directory can be created to contain all of the word processing files on the computer and can contain additional directories called WORK, PERSONAL, and SCHOOL, allowing the user to organize their various word processing files into logical categories.

On personal computers, file management is performed by maintaining a record of every file and its location within a single table. On DOS and early Windows operating systems (up until Windows NT), the table that was used to maintain file information was referred to as the File Allocation Table (FAT). The information contained within the FAT provided a reference to what files were available on the disk, and where they could be found. Of course, if the information within FAT became corrupted or otherwise destroyed, the computer would be incapable of "finding" the programs and data files on the storage media, even if they were still actually there!

More recent versions of Windows utilize the New Technology File System (NTFS) for disk storage tasks. Under NTFS, file data are not maintained in FAT, but in a more comprehensive table known as the Master File Table (MFT). Other vendors use alternative systems such as the High Performance File System (HPFS) and many others. Despite the particular file management technique employed, the basic function remains the same: to allow the user to store, retrieve, update, and store again programs and data files onto a media capable of efficient and reliable storage.

B. Input/Output (I/O) Operations

A basic job of an operating system is to provide the user with access to peripheral devices such as scanners, printers, monitors, and a myriad of other devices. The job of managing and communicating with all of the various peripherals that a user may have belongs to the input/output component of the operating system. At the most basic level, the three functions of an operating system are to allow the user to perform input, processing, and output functions. Without the ability to input data, the processing function could not be accomplished. Similarly, without the ability to output data and information to a screen or printer, the results of processing would never become known!

Basic file operations such as creating a file, writing a file, reading a file, repositioning within a file, deleting a file, truncating a file, appending to a file, copying a file, or renaming a file could not be accomplished without an input/output system. From the time when a user issues a command such as a request for the credit limit of a customer from the keyboard until he or she sees the actual information on the monitor, a number of activities take place within the

computer to fulfill the request. These activities are under the control of I/O operations of the operating system. First the request may go into a queue waiting for its turn to be processed by the CPU. Second, the OS and the database management system cause the read/write heads to position over the proper cylinder (called a seek), the proper head is selected for reading, then the proper sector is rotated under the read/write head (called latency) and the actual information is then transferred to the output device, which in our example is the computer monitor. The same principle applies when a print job is requested or when the user moves the contents of one directory to another.

The sheer number and variety of I/O devices available on the market makes standardization of the I/O subsystem extremely difficult. On the other hand, efforts to achieve standardization have resulted in improvements to the I/O subsystem. These improvements include better and more stable software and hardware interfaces, allowing computer manufacturers to bundle more complex I/O devices into the computer systems and operating systems available today. To make this possible, I/O devices are shipped with device drivers, which provide a standard method of communication between the I/O subsystem and the device and which contain information relevant to the management and capabilities of the device.

Computers must be capable of communicating and working with a diverse variety of I/O devices. Commonly used I/O devices are outlined in Table II.

The computer communicates with these various devices via a cable or wireless connection such as infrared transmission. The connections on the computer that are used to achieve communications with I/O devices are called *ports* (such as a serial, parallel, or USB port). The wires that are used to send and receive signals inside the computer are referred to as a *bus*. Finally, many I/O devices come complete with a separate *controller,* which is combination of electronics that can be used to operate a bus, port, or device. The controller typically includes a separate processor, microcode to run on the processor, and separate memory that is specifically devoted to the needs of a particular device.

Table II Commonly Used I/O Devices

Human interface devices	Screen, keyboard, mouse, light pen, scanners, joystick, touch screen, microphone, and speakers
Transmission devices	Network cards, modems, radio-frequency (RF) spectrum, spread spectrum, infrared light, satellite transmission, and microwave
Storage devices	Floppy disks, hard disks, CD-ROM, magnetic tapes and zip disks

An I/O device can get the attention of the CPU through the use of *interrupts*. Within the CPU there is a special wire called the *interrupt request line*. This line is checked to see if there is a new interrupt signal after each instruction has executed. If the CPU detects a signal on the interrupt request line, it passes control to the *interrupt handler routine*. The interrupt handler routine determines the cause of the interrupt, performs the necessary processing, and passes control back to the CPU once processing has been completed. In this manner, any I/O device that is connected to the computer can get the CPU's attention and obtain a share of CPU execution time. Modern systems contain interrupt-processing capabilities that allow the computer to:

- Defer processing of an interrupt request if any critical processing tasks are taking place
- Eliminate the need to poll all I/O devices through the use of sophisticated polling techniques to determine which device generated the interrupt
- Take advantage of multilevel interrupts to enable the operating system to distinguish between low and high priority interrupt requests

One of the most important functions of the I/O subsystem is to provide efficient scheduling. Scheduling refers to the order in which tasks are to take place. A well-designed I/O subsystem will include a scheduling component that greatly increases the efficiency of the system. For example, assume that the read/write head of the disk armature is positioned at the beginning of the disk, and three separate applications have generated a request to read data from the disk. The data requested by the first application are located at the end of the disk, while the data requested by the second application are located back at the beginning of the disk, and the data requested by the third application are in the middle of the disk. Performing the requests in the order received would mean repositioning the read/write head to the end of the disk for the first data read operation, then repositioning back to the beginning of the disk for the second request, and repositioning again to the middle for the third request. Table III illustrates this sequence.

In the sequence of Table III, the read/write head moved across the entire surface of the hard disk 2.5

Table III Example of an Inefficient I/O Subsystem Scheduling Component

	Current read/write head location	Move to	Perform
Start	Beginning of disk	End of disk	Read request 1
	End of disk	Beginning of disk	Read request 2
	Beginning of disk	Middle of disk	Read request 3 **End**

times. Obviously, a more efficient sequence would be for the hard drive to perform disk read 2, followed by 3, and finally by 1. Table IV illustrates this sequence.

In the sequence shown in Table IV, the hard drive has completed all of the requested read operations, but has only traveled across the entire disk surface 1 time. By scheduling the order of read operations to minimize unnecessary read/write head travel, the I/O subsystem has greatly increased the efficiency and performance of the system.

Two other methods that are commonly used to enhance the performance of I/O subsystems are through *buffering* and *caching*. Buffers are used to temporarily store data while it is waiting to be transferred between two devices or between a device and a program. Buffers can improve the performance of systems by eliminating any speed mismatch that may exist between the sender and receiver. For example, network interface cards and modems contain buffers that allow them to accumulate data until the buffer is full whereupon the buffer contents are transferred to memory for processing. This eliminates the need for constant data transfer at the slower modem or network interface card speed into the computer's memory. In addition, buffers allow devices that have different send and receive window sizes to communicate with each other. For example, if a device with a large data window size must communicate with a device that has a small data window, the data may be moved into buffers while it is broken down into the smaller window size required by the receiving device. On the other hand, a *memory cache* makes a copy of data that already exist in another memory location and places the copy into a faster region of memory. Since access to cache memory is much faster than access to standard memory, the data held within the cache can be processed far more quickly.

C. Memory Management

Each PC or workstation has a limited amount of random access memory (RAM). The effective management of RAM has a direct impact on the performance of the system. When the PC or workstation is turned on, a portion of the OS is loaded into RAM, leaving the remaining memory available for other requests. If RAM requirements exceed the amount of available RAM, virtual memory is used. Virtual memory is really hard disk space that the computer treats as if it were RAM. When the computer runs out of RAM, it automatically switches to virtual memory to fulfill user requests and memory management tasks. Switching between RAM and virtual memory is known as "swapping." By swapping between RAM and virtual memory, the OS fulfills its memory management tasks. Naturally, if there is a lot of swapping the responsiveness of the system will be reduced since the OS is using an electromechanical device (the hard disk) instead of RAM (which is purely electronic).

Memory is the electronic holding place where applications and data can be stored while being accessed and manipulated by the central processing unit. Data can be moved in and out of memory much faster than secondary storage simply because the operations involved are purely electronic, as opposed to electro-

Table IV Example of an Efficient I/O Subsystem Scheduling Component

	Current read/write head location	Move to	Perform
Start	Beginning of disk		Read request 2
	End of disk	Middle of disk	Read request 3
	Beginning of disk	End of disk	Read request 1 **End**

mechanical with hard drives or other forms of secondary storage. When the computer is being used, the operating system is loaded from secondary storage into memory. Memory is sometimes also referred to as random access memory (RAM) and comes in the form of chips that are installed physically close to the CPU on the motherboard of the computer. Most desktop computers today are equipped with 128 or 256 megabytes of memory. The more memory installed on the computer, the less the system must "fetch" data or program instructions from the hard drive making the computer faster.

In simple terms, application programs are stored in the form of binary executable files. When a program is executed, an instruction (also called a process) from the program is moved from secondary storage into an input queue. From there, the process is moved into memory to be executed. Once the process has been executed, the instructions that it contains make reference to other instructions and data. After the process has been executed, it is cleared from memory and the memory is once again available for another process to be loaded from the input queue. This process is repeated for all of the processes that are called by the program according to the tasks being performed by the user.

Memory addresses are separated into two broad categories: logical or physical.

A memory address that is generated by the CPU is referred to as a logical address or a virtual address. On the other hand, a memory address that is seen by the memory unit is referred to as a physical address. Taken together, the memory space allotted for all logical addresses is known as the logical address space, while the space allotted to all physical memory is known as the physical address space. The hardware device that is responsible for mapping logical addresses to physical addresses (and vice versa) is known as the memory management unit (MMU). Instructions and processes are loaded from physical memory into logical memory to be accessed by user programs. User programs do not directly see physical memory addresses. Instead, user programs work with logical addresses to execute instructions and processes. The logical addresses used by user programs are converted (mapped) to physical addresses by the MMU.

D. Resource Allocations

In addition to disk storage and memory allocation there are other resources that must be managed. Among these resources the CPU is the most important. Decisions such as, "Which jobs will be processed and in what order?" (first-in first-out, or based on some priority determination scheme) are a part of the resource allocation responsibility of an OS.

An important function of the operating system is to ensure that mutually exclusive applications can share system resources without interfering with each other. This has become increasingly difficult over the past several years as the information processing demands placed upon processors by sophisticated applications have increased significantly. Not only have the information processing needs increased, but so have the computational, data movement, and logic processing requirements. Of course, resource allocation is critically important to ensuring that these independent functions can be carried out for currently processing applications in as efficient a manner as possible.

E. User Interfaces

The method in which the user interacts with a computer and a software application is part of the user interface responsibility of an OS. In the DOS environment this interface is called the command-line user interface (CLUI). Using the keyboard, the user must issue a specific command and then the system carries out the request. In a graphical user interface (GUI) environment such as OS/2 or Windows, a series of icons are displayed on the screen and mouse clicks are used to execute user programs and carry out other user requests.

VIII. EXAMPLES OF POPULAR OPERATING SYSTEMS

Major groups of operating systems currently in use include operating systems for personal computers (PCs), local area networks (LANs), and mainframes.

A. Personal Computers

The disk operating system (DOS) was the first widely installed OS on personal computers. Developed by Microsoft, the first version of DOS was referred to as PC-DOS in reference to its use on personal computers. Microsoft signed an agreement with IBM to develop a version of DOS specifically for the IBM line of computers and named the version MS-DOS. Very few differences existed between the two operating

systems, however, and ultimately both versions were referred to simply as DOS.

These operating systems were nongraphical, character-based, and were generally categorized as "menu-driven" or command line user interface (CLUI) systems. Although they were a relatively simple interface in terms of design, they were not considered user-friendly or easy to use. Early versions of the Windows operating system were in fact not truly operating systems at all. Instead, when a Windows computer was turned on, DOS would be loaded followed by Windows which was loaded "on top" of DOS. For this reason, many users correctly asserted that early versions of the Windows operating system operated as a shell, relying instead on DOS to perform the core kernel functions. Despite the fact that Windows 3.x operated on top of DOS, it still represented a major improvement over earlier releases of Windows. The ease-of-use and point-and-click aspects of Windows 3.x represented a significant improvement over earlier versions of Windows and DOS interfaces. Windows 95 represented another major shift in the ease-of-use of the Windows user interface by radically changing all aspects of the look-and-feel of the operating system. Windows 95 finally replaced DOS as the primary operating system, and no longer resided on top of DOS in order to perform its functions. In fact, the popularity of the Windows 3.x and 95 user interfaces has directly contributed to the revolution that has placed computers in so many households and businesses throughout the world.

The Apple OS was the product of Apple Computer, founded by Steve Jobs and Steve Wozniak. The Apple II, introduced in 1977, was the first computer to produce color graphics, and work on the Apple III began in 1980. However, it was the introduction of the graphical user interface that represented the single most important development in the history of the Apple operating system. Initially developed by the Xerox Corporation, the graphical user interface was brought to full functionality by Apple. The operating system was very easy to use and controlled with the click of a mouse. The new operating system was first deployed on a project called Lisa. The Lisa project was ultimately canceled in favor of a less ambitious product development effort resulting in the creation of the Macintosh. The introduction of IBM's first PC in 1981 brought about a sense of urgency to the Macintosh project. In addition, Microsoft began development of the first version of Windows for the IBM PC, pushing the Macintosh development team even harder. The Macintosh computer was finally introduced in 1984. Windows OS will be further discussed in the next section.

B. Local Area Networks

Popular LAN operating systems are outlined in Table V. In this section we review three of the most popular ones including Microsoft NT Server, Novell NetWare, and Linux.

Windows NT Server is a preemptive multitasking and multithreading network operating system. "Preemptive multitasking" means that the operating system allocates processor time to one or more applications that are running. Alternatively, cooperative multitasking allows each application to control how much CPU time they get and to surrender the processor after a certain amount of time. If an application refuses to yield control of the CPU, all other applications stop running. The term "multithreading" means that programs can start subtasks that are then executed by the operating system in the background. Applications that utilize multithreading can be much more responsive to the user's needs. Windows NT Server provides an application server foundation with tightly integrated file and print sharing, backup and recovery management, and a high level of security by using the NT file system (NTFS) allowing permissions to be set on a file and directory basis.

Novell NetWare has the largest installed base of any LAN operating system on the market. NetWare possesses a number of important security features, including a technology for data encryption that prevents computer hackers from extracting password and other ID information from network traffic. NetWare also features an auditor function that prevents even systems administrators from changing the audit logs, resulting in greater data integrity and administrator confidence in the activity logs. As with Microsoft NT, a major benefit of Novell NetWare is that it works on multiple client operating systems.

Table V Popular LAN Operating Systems

Novell NetWare
Microsoft Windows 95, 98 and 2000
Microsoft Windows NT
IBM OS/2 Warp
Unix
Linux
Banyan Vines
Artisoft LANtastic
Macintosh Apple Talk

Linux (pronounced "lih-nucks") is a free or very low-cost operating system comparable to traditional and usually more expensive Unix systems. It is a full-featured, multiuser, multitasking operating system that runs on 80386 processors and beyond. Linux comes in versions for all the major microprocessor platforms including the Intel, PowerPC, Sparc, and Alpha platforms. Linux is a complete operating system, including a graphical user interface, X Window System, TCP/IP compatibility, the Emacs editor, and other components commonly found in more comprehensive operating systems. Development of the Linux OS began as a postgraduate project by Linus Torvalds, a student at the University of Helsinki in Finland. The first official version of the Linux OS was released in October 1991, and with the improvement efforts of hundreds of Unix programmers' worldwide, new features have been added every day. Despite its power, Linux is still simple enough for end users that need an inexpensive, efficient OS for accounting, word processing, or Internet browsing.

C. Mainframes

Two popular operating systems in the mainframe environment are the virtual memory system (VMS) and multiple virtual storage (MVS). In 1979, Digital Equipment Corporation (DEC) developed an operating system called the Virtual Memory System (VMS) for the new VAX mainframe (successor to the PDP-11 computer). VMS is a 32-bit operating system that exploits the concept of virtual memory. As explained earlier, virtual memory, by using hard disk space, allows programmers to use a very large range of memory addresses for stored data. The operating system is capable of mapping virtual addresses (RAM) to real hardware storage addresses. In addition, a virtual memory system could manage swapping data between active storage (RAM) and the hard disk or other storage media, reducing the amount of physical storage requirements and speeding up overall system performance.

VMS was renamed OpenVMS when it was scaled down and redeveloped for the Alpha line of microprocessors and computer systems, also developed by DEC. The "Open" part of OpenVMS is a reference to the support added for Unix-like interfaces.

IBM developed multiple virtual storage (MVS) for use on most of its mainframe and large server computers. The most critical functions associated with business information processing are still performed to this day on systems that are based on the MVS system. Over the past few years, IBM has begun to reposition MVS into the large server and distributed processing market. It is likely that the term MVS will continue to live in the lexicon of computer terminology due to its significant influence in computing environments of all types.

IX. CRITERIA FOR SELECTING AN OPERATING SYSTEM

The selection of an appropriate operating system is largely determined by the particular software applications that are required in any given environment. After determining which software application best meets the needs of the user, the next task is to identify the operating system and hardware requirements of the selected software. In some cases, the operating system and hardware requirements can be very limited. For instance, in the school business environment, if the SCHOOL3000 administrative financial system software marketed by QSS (Quintessential School Systems) is selected for implementation, the choice of operating system and associated hardware is made as well since SCHOOL3000

Table VI Important Criteria for Selecting a LAN Operating System

Compatibility with other networks (the installed base)
Cost (initial, maintenance, and upgrade)
Degree of expandability
Distance coverage
Documentation (online and offline)
Ease of administration, maintenance, and management
Fault tolerance
File services, the ability to store many different file formats and data compression
Network management, a single point of control for all network resources
Number and types of printers supported
Number of concurrent users
Number of workstations supported
Protocol support
Routing support, the ability to use a variety of network layer routing protocol
Security
Speed
Types of workstations supported
Vender support, reputation, and financial strength

software runs only on HP3000 computer models running the MPE (multi-programming executive) operating system. MPE is a multiuser operating system that Hewlett–Packard developed in the early 1970s for the HP3000 line of computers. Still, at other times the selection of an operating system can leave the user with a myriad of bewildering choices, each with unique advantages and disadvantages. To illustrate, let's examine the considerations associated with the selection of a network operating system (NOS).

Selecting a NOS is a complex and a multidimensional task. Many IS managers today deploy both Windows NT and Novell NetWare network operating systems to meet their business needs. Similar to other software and hardware selection considerations, a number of criteria must be carefully examined. An organization first must define its goals and objectives. The existing computing environment, security needs, present and future applications, size of the organization and future growth must be considered. Table VI provides a list of important criteria for selecting a NOS.

SEE ALSO THE FOLLOWING ARTICLES

Computer Hardware • Computer Software

BIBLIOGRAPHY

Bidgoli, H. (2000). *Business data communications: A managerial perspective.* San Diego: Academic Press.

Brumfield, J. A. (1986). A guide to operating systems literature. *Operating Systems Review,* Vol. 20, No. 2, 38–42.

Coffman, E. G., Denning, P. J. (1973). *Operating systems theory.* Englewood Cliffs, NJ: Prentice Hall.

Deitel, H. M. (1990). *An introduction to operating systems,* 2nd ed. Reading, MA: Addison–Wesley.

Pinkert, J., and Wear, L. (1989). *Operating systems: Concepts, policies, and mechanisms.* Englewood Cliffs, NJ: Prentice Hall.

Silberschatz, A., Galvin, P. B., and Gagne, G. (2001). *Operating system concepts,* 6th ed. New York: Wiley.

Tanenbaum, A. S., and Woodhull, A. S. (1997). *Operating system design and implementation,* 2nd ed. Englewood Cliffs, NJ: Prentice Hall.

Operations Management

Chen H. Chung
University of Kentucky

GLOSSARY

aggregate planning Aggregate planning is also called aggregate production planning or simply production planning. The goal of aggregate planning is to determine the aggregate levels of production, inventory, and workforce to respond to fluctuating demand in the medium-term future.

automated manufacturing system (AMS) An interconnected system of material processing stations capable of automatically processing a wide variety of part types simultaneously under computer control. The system is not only interconnected by a material transport system, but also by a communication network for integrating all aspects of manufacturing. An AMS may include several enabling technologies such as computer-aided design (CAD), computer-aided manufacturing (CAM), computer-aided process planning (CAPP), automated materials handling (AMH), etc. Sometimes the terms CIM (computer integrated manufacturing) and AMS are used interchangeably.

communication-oriented production information and control system (COPICS) An integrated computer-based manufacturing system developed by IBM in the early 1970s. COPICS is oriented to production and related applications. It also provides data to other major areas such as sales, finance, personnel, design, research, quality assurance, etc.

computer-integrated manufacturing (CIM) An umbrella term which includes automated technologies such CAD, computer-aided engineering (CAE), CAPP, CAM, and hardware systems such as computer numerically controlled (CNC) machines, AMH system, automated storage and retrieval systems (AS/RS), automated guided vehicle (AGV) system, and flexible manufacturing systems (FMS).

distribution requirements planning (DRP) When the manufacturing requirements planning logic is applied to a firm's distribution system, the application is called distribution requirements planning. DRP provides the basis for tying the physical distribution system to the manufacturing planning and control system. When DRP is linked to MRP in a typical manufacturing firm, the forecasts can be driven through the organization from distribution to manufacturing and on to procurement.

enterprise resources planning (ERP) On one hand, ERP extends MRP II to include engineering, finance, human resources, and other activities in the enterprise, and thus called enterprise resources planning. On the other hand, the emergence of local and global networks makes it possible for firms to go beyond traditional modes of developing systems for individual functional areas in a fragmented manner. ERP intends to integrate and automate various modules (i.e., softwares) for managing and controlling traditional back-office functions and activities such as finance, human resources, marketing, and operations.

e-operations (e-ops) A term used for describing the application of the Internet and its attendant tech-

nologies to the field of OM. E-ops applications can be used for the internal processes of an organization, such as back-office support and common factory processes of ordering, scheduling, product design and development, quality monitoring, etc. E-ops can also be used for processes that support the external linkages between a company's suppliers and customers.

group technology (GT) A manufacturing philosophy which groups dissimilar machines into work centers (or cells) to work on products that have similar shapes and processing requirements.

manufacturing planning and control (MPC) system An MPC system is simply another name for MRP II. That is, the basic functions of an MRP II system are to plan and control manufacturing activities.

manufacturing resources planning (MRP II) An MRP system augmented with front- and back-end activities. The front end includes demand management, aggregate planning, master production scheduling, etc. The back end includes implementation of MRP such as inventory management, priority management, capacity management, etc.

master production schedule (MPS) An anticipated build schedule for manufacturing end products or product options. It represents what the company plans to produce in terms of models, quantities, and dates. On the one hand, the MPS translates the company's aggregate production plan into specific schedule for production activities. On the other hand, the MPS drives the "engine" and the "back end" of a manufacturing planning and control (MPC) system.

material requirements planning (MRP) An MRP system is a massive information system which performs detailed planning for the timing and sizing of material requirements. Its objective is to provide the right part at the right time to meet the schedules for completed products.

operations management (OM) Concerned with the economical use of inputs (human resources, capital, materials, etc.) in a transformation process that results in goods or services. It involves decision making on how to best design and operate a production system.

supply chain management (SCM) SCM deals with the management of the system which an organization procures raw materials, components and parts, transforms them into intermediate and final products, and delivers the products and services to their customers.

I. INTRODUCTION

Every organization produces some good or service. Operations management (OM) deals with making such production processes effective and efficient. The design, operation, and improvement of the production system require substantial information and decision support. This article surveys the state of the art of information systems for OM and discusses the design and implementation issues of these systems. It also provides some perspectives on the future development of such systems.

A. Operations Management Defined

Operations management is concerned with the economical use of inputs (human resources, capital, materials, etc.) in a transformation process that results in goods or services. It involves decision making on how to best design and operate an operations system or generally referred to as production system. The term *production system* includes organizations that manufacture (tangible) products or offer (intangible) services. Examples of the latter are hospitals, banks, entertainment industries, government agencies, charity organizations, and restaurants.

B. Key Decisions in Operations Management

Figure 1 outlines key decisions in OM and the roles played by information systems in supporting these decision-making processes. It should be noted that the OM system depicted in Fig. 1 is applicable to both manufacturing and service sectors. All the OM decisions will also be encountered in the global operations settings. The categorization of decisions into long, medium, and short terms is simply for the convenience of discussion. Although these decisions correspond to the strategic planning, tactical planning, and operational control activities in a firm, the time horizons for long, medium, and short terms are usually defined in a relative sense. Also, all temporal decisions in the OM system will have impact upon other decisions within or across horizons.

Figure 1 shows that all OM decisions rely on the support of information systems. Indeed, all planning activities require past data as well as forecasts of future events (e.g., demand forecasts). In addition to

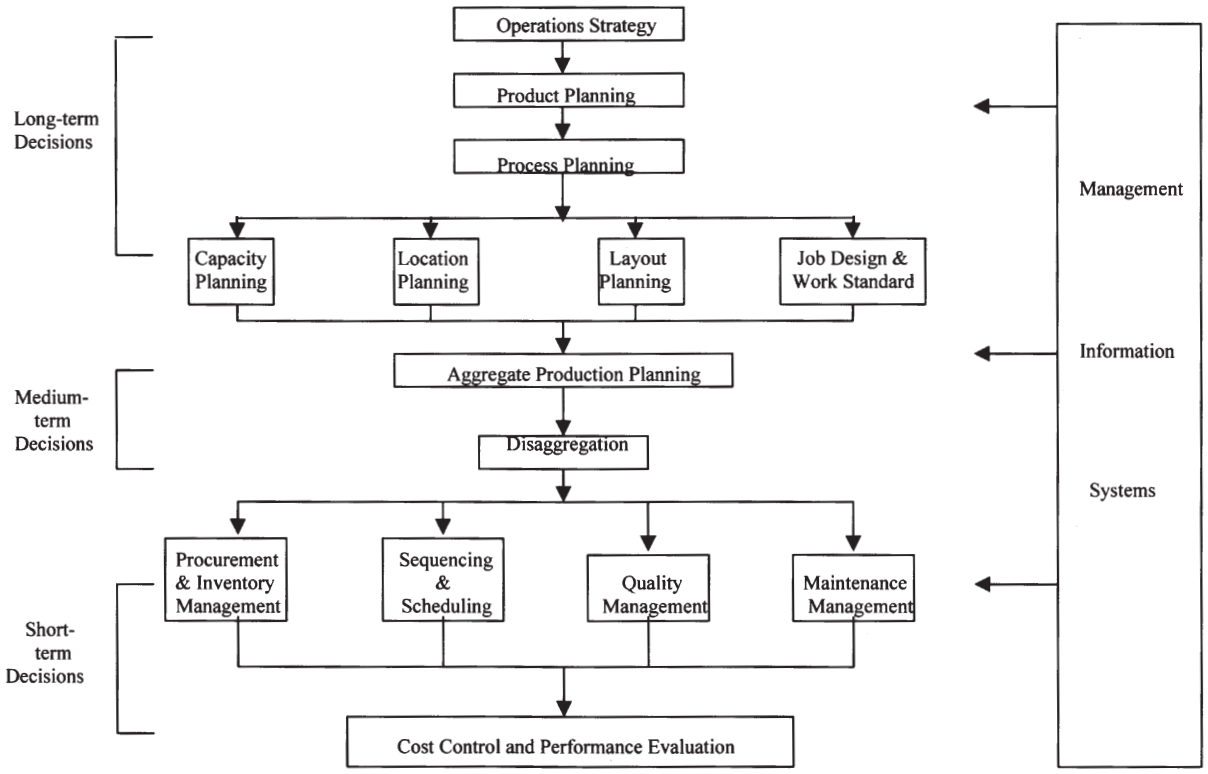


Figure 1 The operations management system.

providing data and information, the information system also includes knowledge-based decision support systems (DSS) to provide decision and planning aids.

The long-term decisions in the OM system basically deal with operations strategy, product planning, and the design of facilities and processes. Issues to be addressed include what to produce (product planning), how to produce (process planning), how much can be produced (capacity planning), and where to produce (location and layout planning). The medium- and short-term decisions are concerned with planning and control of production activities.

In the following sections, our discussion will be organized according to these temporal decisions. After addressing the issues associated with the information and DSS for operations strategy, we will discuss information systems for the design of facilities and processes, followed by discussions on operations planning and control systems. We then briefly outline the issues in service operations. For details of information systems for service operations, readers are referred elsewhere in this encyclopedia. We conclude our discussion by providing some perspectives on future development in information systems for OM.

II. INFORMATION AND DECISION SUPPORTS FOR OPERATIONS STRATEGY

A. Operations Strategy Defined

Operations strategy is concerned with setting the long-term direction of a firm's operations function to best support its overall corporate strategy. This does not mean that operations strategy should always assume a subordinate, reactive role in the corporate strategy. On the contrary, operations strategy quite often dictates a firm's corporate strategy. Either way, a firm's operations strategy should be integrated with its corporate strategy.

In 1969, Skinner introduced the concept of manufacturing strategy and suggested that the manufacturing function can provide a firm with formidable competitive potential. A similar conclusion is equally applicable to the service sector. In a 1992 *Harvard Business Review* article, Stalk, Evans, and Shulman point out that competition is a "war of movement" in which success depends on anticipation of market trends and quick response to changing customer needs. Successful competitors move quickly in and

out of products, markets, and sometimes, businesses—a process more akin to an interactive video game than chess. In such an environment, firms are actually competing on the basis of information. To compete successfully one must gather information about market and competitive conditions quickly, rapidly analyze and make decisions based on available information, and deploy resources to implement corporate strategy in a timely manner. Indeed, the importance of providing information and decision support to the develoment and implementation of corporate/operations strategy can never be overemphasized.

There are two types of (not necessarily mutually exclusive) operations strategy. The first type of operations strategy involves decisions that relate to the design of a process and the infrastructure needed to support the process. Process design decisions involve the selection of technology, the capacity of the process, the location and layout of the process, etc. The infrastructure decisions include the development of the planning and control systems, quality management systems, the organization of the OM functions, etc. The other type of operations strategy refers to the firm's competitive strategy in which operations will play the key role. For example, the so-called activities-based strategy, capabilities-based strategy, and time-based competition (TBC) are considered operations strategies.

B. Decision Supports for Operations Strategy

Although strategic decision support systems (SDSS) have received substantial attention in the literature, very few SDSS are specifically designed based on a particular competitive strategy. While a general SDSS may have the advantage of being flexible, a support system which is developed based on a particular competitive strategy will better facilitate the implementation of that strategy. Figure 2 is a general framework for SDSS. The framework consists of four subsystems of a general DSS: user interface subsystem (UIS), problem processing subsystem (PPS), knowledge subsystem (KS), and subsystem interface management software (SIMS). The support for strategic decision making is performed by several programs within these subsystems. For example, there is a missions/goals

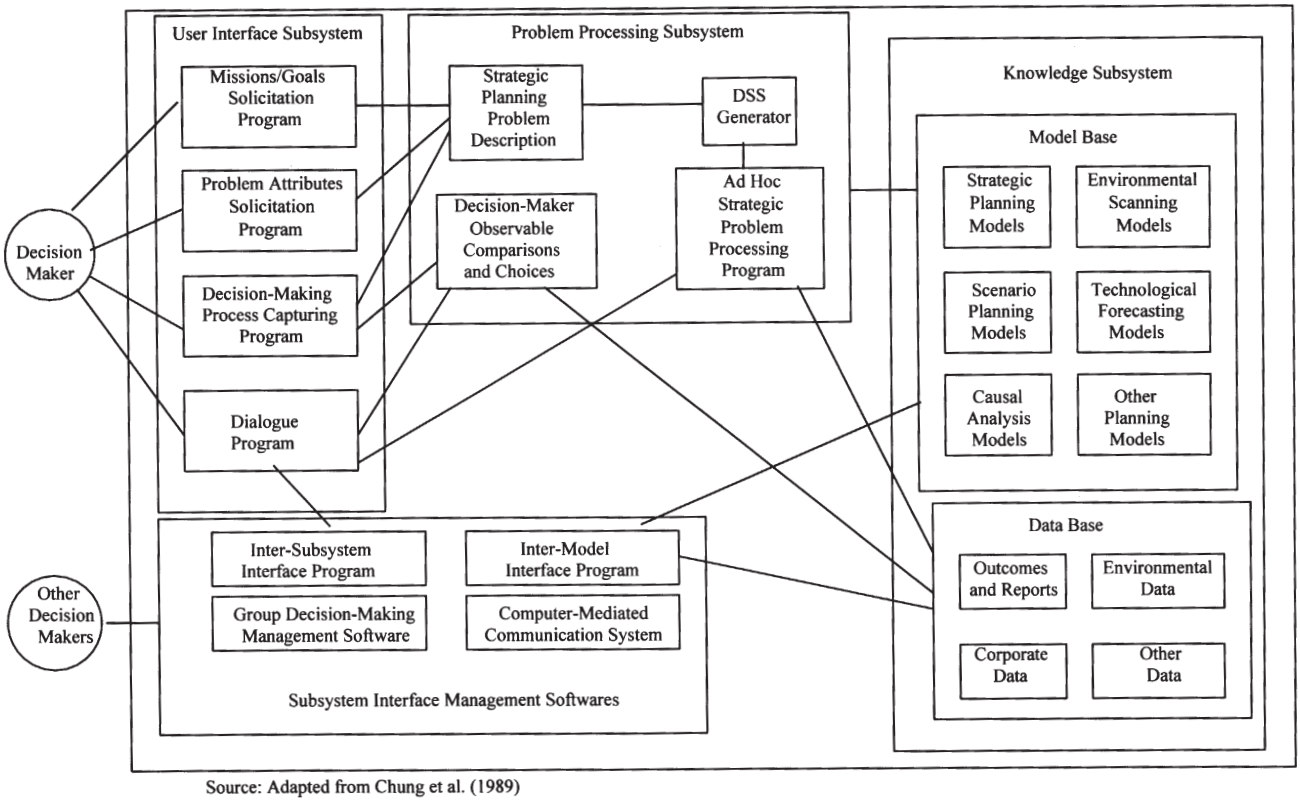


Figure 2 A framework for strategic decision support systems. [Adapted from Chung *et al.* (1989). *Omega,* 17(2), 135–146.]

elicitation program in the UIS. There are strategic problem descriptions and ad hoc strategic problem processing programs in the PPS. In the model base of the KS, there are strategic planning models, environmental scanning models, scenario planning models, technological forecasting models, etc.

Decision supports for operations strategy can be built upon the above SDSS framework. For example, Cook and colleagues incorporate a TBC model called CCP—change, causation, and possibility—in the above SDSS. The CCP model is a cycle of three-stage tasks. The environmental scanning model of the SDSS is to constantly monitor the changes in both the firm's external and internal environments. Diagnostic problem-solving systems or expert systems will perform causal analysis upon the relevant and significant changes detected in the first stage. Based on the results of causal analyses, creativity facilitation systems and scenario planning systems will then search for meaningful changes and the associated possible scenarios. The scenario-driven planning (SDP) process paves the way for the implementation of (TBC-based) operations strategy.

III. INFORMATION SYSTEMS IN THE DESIGN OF FACILITIES AND PROCESSES

The design of facilities and processes actually starts with product planning. Product engineering and process engineering are usually inseparable. This is particularly true when the so-called concurrent engineering (CE) approach is used. Rather than a simple serial process in which the design project proceeds from one phase to another, CE emphasizes cross-functional integration and concurrent development of a product and its associated processes.

A. From Product/Process Design to Computer Integrated Manufacturing

There are many software programs available for facilitating the product/process design activities. For example, QFD (quality function deployment) and VA/VE (value analysis/value engineering) software help design engineers identify customer requirements, technical characteristics, ways to eliminate unnecessary costs, etc. On the other hand, computer-aided design (CAD) is an approach to product and process design that utilizes the power of the computers. Computer-aided engineering (CAE) is the process that helps the evaluation of the engineering characteristics. Computer-aided manufacturing (CAM) applies computers and microprocessors to direct manufacturing activities. Computer-aided process planning (CAPP) is used to design the computer part programs that serve as instructions to computer-controlled machine tools, and to design the programs used to sequence parts through the machine centers and other processes needed to complete the parts. Computer-aided process planning bridges the gap between CAD and CAM. Basically, CAPP determines how to convert a product design to the final form of the manufacturing product. Different volume representations and decompositions would entail different cutting paths, machine tools, setup and machining costs, etc. Thus, the actual manufacturing activities and related tool requirements are greatly affected by CAPP. All these automated technologies, and other hardware systems such as computer numerically controlled (CNC) machines, automated materials handling (AMH) system, automated storage and retrieval systems (AS/RS), automated guided vehicle (AGV) systems, and flexible manufacturing systems (FMS) are brought together under the notion of computer-integrated manufacturing (CIM).

B. Virtual Factory

Although called "integrated," CIM is usually a collection of various automated technologies. A truly "integrated" system will place more emphasis on the interfaces among technologies, functions, programs, etc. This is particularly crucial for the case of the virtual factory. The term virtual factory refers to manufacturing activities carried out not in one central plant, but rather in multiple locations by suppliers and partner firms as part of a strategic alliance. In this setting, it is essential for the virtual manufacturer to have a deep understanding of the manufacturing capabilities of all parties in the production network. An integrated information system becomes crucial to the success of carrying out the difficult task of coordination.

IV. INFORMATION SYSTEMS FOR PLANNING AND CONTROL OF OPERATIONS

The medium- and short-term decisions in an OM system (as shown in Fig. 1) deal with a firm's production planning and operational control activities. In this section, we discuss the evolution of information systems for such activities—from the early Communication Oriented Production Information and Control

Systems (COPICS) by IBM to the most recent developments in enterprise resource planning (ERP) systems.

A. COPICS

In the early 1970s, IBM developed an integrated computer-based manufacturing system called COPICS. It is oriented to production and related applications. It also provides data to other functional areas such as sales, finance, personnel, design, research, quality assurance, etc. It is built around a database creation and maintenance system that permits easy file reorganization when system changes occur. Data can be reorganized without incurring the significant expense of modifying existing programs that will have to use the new files. Data duplication is thus significantly reduced. This also leads to reduction in data storage and maintenance costs as well as computer processing time. Real-time processing is another essential feature of COPICS. Data are transmitted on-line, that is, by means of terminals linked directly to the computer. The records are updated immediately after the transaction occurs. Consequently, the system is in position to respond (if response is required) without the usual delay inherent in periodic batch processing.

The applications covered by COPICS range from engineering and production data control to cost planning and control. Table 1 summarizes these applications.

Since COPICS involves substantial data processing, it needs to be run on mainframe computers. This was the technological constraint in the 1970s. The rapid advancement of computer and information technologies during the past two decades has resulted in shifting most applications away from mainframe computers. However, the basic principles of COPICs provide the foundation for later developments in material requirements planning (MRP), manufacturing resources planning (MRP II), and ERP.

B. Material Requirements Planning

An MRP system is a massive information system which performs detailed planning for the timing and sizing of material requirements. Its objective is to provide

Table I The Applications Covered by COPICS

Applications	Functions
Engineering and production data control	Creates and maintains basic engineering records
Customer order servicing	Links the sales information system to manufacturing
Forecasting	Provides techniques to project finished product demand and establish management standards to control manufacturing activity
Master production schedule planning	Allows quick assessment of the impact of alternate production plans on plant capacity
Inventory management	Determines the quantities and timing of each item to be ordered—both manufactured and purchased—in order to meet the requirements of the master production schedule
Manufacturing activity planning	Is used to plan detailed capacity requirements and to adjust the date of planned order release to be consistent with plant capacity
Order release	Is to create the documents authorizing production or purchase of the required material
Plant monitoring and control	Traces the progress of each shop order as it moves through the shop; coordinates many of the supporting activities, such as inspection, materials handling, and tools
Plant maintenance	Addresses maintenance manpower planning, work order dispatching and costing, as well as preventive maintenance scheduling
Purchasing and receiving	Maintains current purchase quotations, creates purchase orders, and follows the progress of the order from the time of requisition, through acknowledgement, follow-up, receipt, quality control, and deposit in stores
Stores control	Keeps track of material location and determines where to store new material
Cost planning and control	Is addressed particularly to the financial executive and provides techniques whereby the information created and maintained for production purposes can be used for budgeting and accounting applications

the right part at the right time to meet the schedules for completed products. An MRP system will generate the production and/or procurement plans for each part number, including raw materials, components, and finished goods.

Based on a time-phased (i.e., stated on a period-by-period basis) master production schedule (MPS), MRP constructs a time-phased requirement record for any part number. The data can also be used as input to the detailed capacity planning models.

The MRP logic is based upon the basic accounting procedure for calculating inventory balances. For any single item (part number), the gross requirements (i.e., the anticipated future usage of or demand for the item) in a period is subtracted from the sum of beginning inventory and scheduled receipts. The result is the projected available balance for the item at the end of that period. If this projected on-hand balance shows a quantity insufficient to satisfy gross requirements (i.e., a negative quantity), additional material must be planned for. This is done by creating a planned order release in time to keep the projected available balance from becoming negative. The planned order release is created by a procedure called lead-time offset—backing from the shortage date for the amount of lead-time to determine the date for the planned order release.

The above basic MRP logic provides the correct information on each part in the system. Linking these single part records together is essential in managing all the parts needed for a complex product or customer order. The MRP uses a gross-to-net explosion procedure to translate product requirements into component part requirements. Explosion is the process of determining, for any part number, the quantities of all components needed to satisfy its requirements and continuing this process for every part number until all purchased and/or raw material requirements are exactly calculated.

As mentioned earlier, an MRP system is a massive information system. Vollmann, Berry, and Whybark point out, "The enormity of the overall database even for small firms is awesome." In addition to basic files such as the item master file, the bill of material file, the location file, the calendar file, and open order files, an MRP system also needs to link to other data files in related areas such as forecasting, capacity planning, production scheduling, cost accounting, budgeting, order entry, shop floor control, distribution invoicing, payroll, job standards, and engineering. Indeed, an MRP system cannot be run in an isolated fashion.

C. MRP

A stand-alone MRP system is usually not sufficient for effective planning of production activities. Additional supporting functions are needed. An MRP system with augmented front and back ends is called an MRP II system or a manufacturing planning and control (MPC) system. The front end includes demand management, aggregate planning, master production scheduling, etc. The back end includes implementation of procurement and inventory management, priority management (e.g., sequencing and (re-)scheduling, dispatching), short-term capacity management, etc. A general framework for an MRP II system with an extension to decisions in Automated Manufacturing System (AMS) environment is depicted in Fig. 3.

The right half of Fig. 3 consists of the traditional MPC or MRP II system. Aggregate planning (AP) and master production scheduling (MPS) are the front end of most production/operations planning and control systems. In aggregate production planning, management is concerned with determining the aggregate levels of production, inventory, and work force to respond to demand fluctuations in the future. The AP provides an overall guideline for MPS which specifies the timing and sizing of the production of individual (end) products. The master scheduler takes the aggregate decisions as targets and tries to achieve them as much as possible. Ideally the sum of production (inventory) quantities in the master schedule over a time period should equal the aggregate production (inventory) quantities for that time period. However, deviations may occur due to considerations such as capacity limitations at critical work centers. The feedback of actual master scheduling performance provides information for modifying future AP. To be feasible and acceptable, an MPS should meet the resource constraints at critical work centers. This feasibility check is called resource requirements planning (RRP) or rough-cut capacity planning (RCP). If the RCP indicates potential infeasibility at some critical operations, either the master schedule is revised to accommodate these capacity limitations or a decision is made to maintain the master schedule and implement adjustments to the capacities at relevant work centers. It may also require adjustments to the long-term capacity plan or the aggregate plan. After the feasibility check via RCP, the feasible and acceptable master schedule becomes an authorized master schedule and can be further broken down into detailed schedules with the use of lower level planning and scheduling systems such as MRP. An MRP system determines the timing and size of the

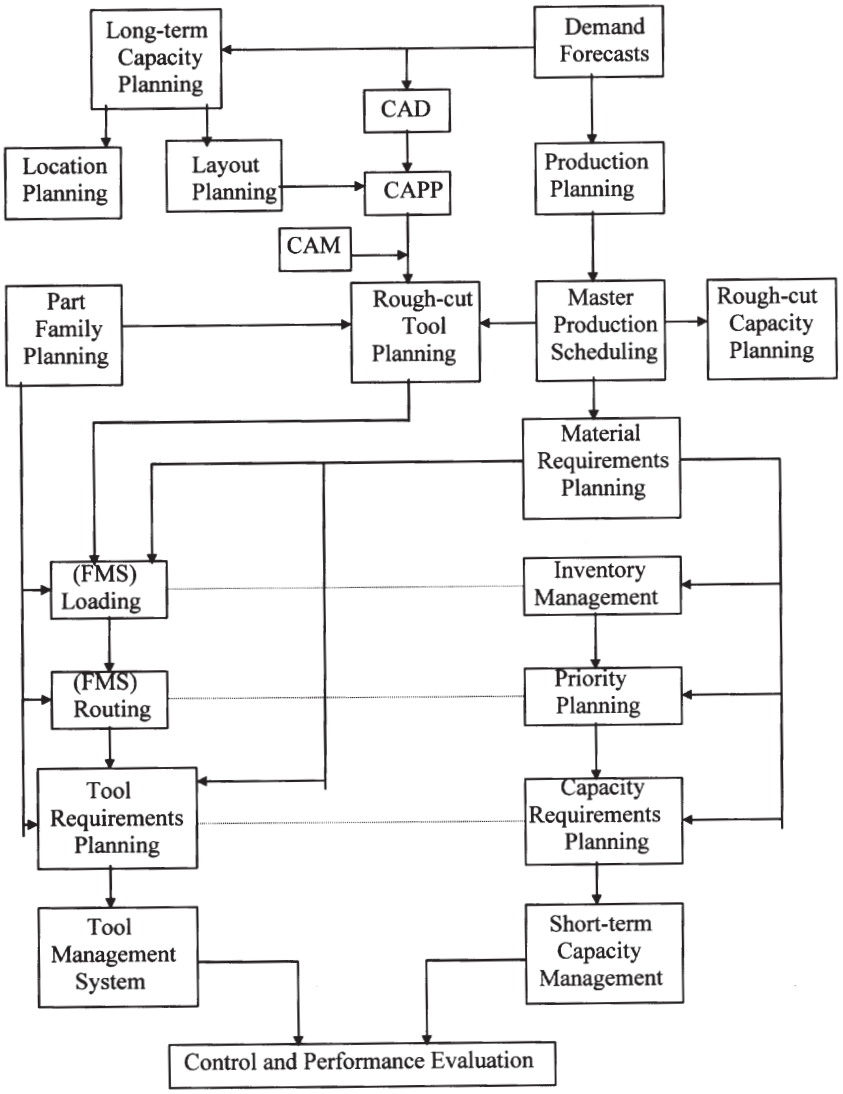


Figure 3 Manufacturing planning and control systems in traditional and automated manufacturing environments.

production of components and parts. It also serves as the basis for detailed inventory management (e.g., procurement and materials control), priority planning (e.g., sequencing and scheduling), and capacity planning. In RCP, the estimation of capacity requirements is based on the MPS information. The capacity requirements planning (CRP) activity, on the other hand, utilizes the information generated by the MRP system, including open orders and planned order releases. The CRP also takes into consideration the current status of all work-in-process inventories in the shop. Thus, CRP provides a more accurate account of the capacity requirements than does RCP.

The left half of Fig. 3 deals with important decisions in an AMS environment. Earlier in III. A, we discussed CIM related tools or software such as CAD, CAM, CAPP, etc. Quite often, it is desirable to pool operations with common tool requirements in AMS. Parts family grouping (PFG) is desirable for the purpose of loading FMS and planning tool requirements. This practice is consistent with the group technology (GT) philosophy which groups dissimilar machines into work centers (or cells) to work on products that have similar shapes and processing requirements. The routing decision is concerned with the order in which a part type visits a set of machines. The loading and routing decisions will have direct or indirect impacts upon the actual tool requirements. Rough-cut tool planning (RTP) is meant to provide a rough estimate of the tool requirements implied by MPS. Figure 3

shows that tool requirements planning (TRP) has CRP as its counterpart in the traditional MPC system. In most AMS, the actual tool requirements are quite often dictated by the tooling policies in the tool management system (TMS).

D. Distribution Requirements Planning

The MRP logic can be applied to a firm's distribution system. Such an application is called distribution requirements planning (DRP). It provides the basis for tying the physical distribution system to the MPC system. Like MRP, DRP is a time-phased, backward scheduling technique that integrates the planning of distribution inventories with manufacturing planning by providing information that identifies need dates, replenishment dates, and order dates for material requirements. Distribution requirements planning ties warehousing operations to transportation and reconciles forecast demand with transportation capacity and inventory. Plans derived from the DRP and from the resulting shipping requirements are the basis for managing the logistics system. Thus, DRP facilitates logistic activities such as vehicle capacity planning, vehicle loading, vehicle dispatching, warehouse receipt planning, etc. When DRP is linked to MRP in a typical manufacturing firm, the forecasts can be driven through the organization from distribution to manufacturing and on to procurement. Since the logic and record formats of DRP and MRP are compatible and all MPC modules are linked, DRP allows a firm to extend MPC visibility into the distribution system.

E. ERP

In the 1990s, MRP II was further extended to include engineering, finance, human resources, and other activities in the business enterprises. Consequently, the term ERP was coined.

The emergence of local area and global networks enables diverse users to share data and resources. This capability allows firms to go beyond traditional modes of developing systems for individual functional areas in a fragmented manner. Both data and processes can be integrated so that inefficiency in storing redundant data and and entering and formatting data from one system to another can be minimized.

Enterprise resource planning intends to integrate and automate various modules (i.e., softwares) for managing and controlling traditional back-office functions and activities such as finance, human resources, marketing, and operations. When implemented correctly, an ERP system links *all* the areas of the business. Thus, information flows quickly across the supply chain. Minimizing redundant data and information processing can lead to substantial savings. Furthermore, many redundant jobs can also be eliminated. With ERP, the firm may be "forced" to reengineer its processes so as to run the business more efficiently.

Consequently, one of the most promising markets for ERP is in supply chain management (SCM). A supply chain is the system with which an organization procures raw materials, components and parts, transforms them into intermediate and final products, and delivers the products and services to their customers. During the past decade, SCM has rapidly become an information technology (IT) challenge that cuts across business relationships, application infrastructures, and corporate cultures as well as the complete "web of partners" (i.e., company employees, suppliers, customers, etc.). While electronic data interchange (EDI) has been quite effective in supporting supply chain activities, a more integrated, extensive and dynamic system is needed as the scope of SCM expands rapidly. Some ERP vendors have recognized such needs and business opportunities. For example, SAP developed an SCM system which consists of three components: advanced planner and optimizee (APO), business-to-business (B2B) procurement, and logistics excution system (LES). The APO contains five major applications. Supply Chain Cockpit provides a configurable graphical user interface for modeling supply networks, retrieving information and event triggers that alert users about pending situations. Demand Planning provides forecasting and planning tools that allow planners to use data warehouse (i.e., SAP Business Information Warehouse) to develop sales forecasts. Supply Network Planning provides optimization and simulation techniques to coordinate activities and plan material flows along the supply chain. Production Planning and Detailed Scheduling supports multiplant planning, material and capacity monitoring, etc. Finally, the Global Available-to-Promise Module provides the capability for checking multilevel component and capacity availability to match supply with demand. The B2B procurement provides web-enabled, real-time procurement of maintenance, repair, and operating (MRO) supplies and services. The LES extends the warehouse management and transportation capabilities of the SAP core system (i.e., R/3).

For a detailed discussion on ERP (e.g., the prominent vendors, implementation issues, etc.) readers are referred to elsewhere in this encyclopedia.

F. Operations Scheduling and Project Scheduling

The Gantt chart, named after its inventor Henry L. Gantt during World War I, is one of the oldest scheduling tools that has seen wide application. Gantt charts became the foundation for later development in the network approach to project management such as the Critical Path Method (CPM), Program Evaluation and Review Technique (PERT), etc. For detailed discussion on information systems for project management, see elsewhere in this encyclopedia.

Gantt or bar charts are also useful for operations scheduling on the shop floor. Typical practice is to use a schedule board to display jobs and their durations. However, operations scheduling is much more complicated than simply showing jobs along time lines. While an operational scheduling is a plan with reference to the sequence of and time allocated for each item, or operations necessary to complete the item, many factors need to be considered to develop such a schedule: due dates, work center resource capabilities (labor and/or machines), availability of materials, job priorities, etc. Numerous sequencing/scheduling rules have been developed. Commercial scheduling software with varying degrees of sophistication are abundant. A typical software package may take advantage of computer technologies such as window, graphics, networks, etc., so as to provide easy access information, what-if analysis, real-time evaluation, (re-)scheduling, and so on.

V. INFORMATION SYSTEMS FOR SERVICE OPERATIONS

A. Characteristics of Services

Service production differs from manufacturing in many ways. First, services are intangible, while manufacturing has physical outputs from the process. Second, services are produced and consumed simultaneously. Therefore, services cannot be inventoried. Unlike manufacturing, where a firm can build up inventory during slack periods for peak demand and thus maintain a relatively stable level of employment and production plan, in services, demand needs to be met when it arises. Consequently, it is important to plan for sufficient capacity to satisfy customer demand. Third, service operations often require a high degree of personalization, rapid delivery speed, and a certain degree of customer contact and/or customer involvement. Furthermore, the inherent variability of the service encounter makes it difficult to standardize services. As a result of the above unique characteristics of services, it is obvious that the need for timely information and decision supports is even greater in service operations than in manufacturing.

B. Information and Decision Support for Service Operations

Information systems for service operations are usually industry dependent. For example, the systems developed for a bank will be quite different from those used in a physician's office, a hospital, or a restaurant. Here, we briefly discuss a few examples. Readers are referred to elsewhere in this encyclopedia for details about the information systems for individual industries.

In the mid-1980s, American Airlines introduced a computerized on-line reservation system, SABRE. The model has since been not only adopted by the whole industry, but also extended to other travel-related industries. The SABRE reservation system also allows American Airlines to constantly monitor the status of its upcoming flights and competitors' flights on the same route so as to make pricing and allocation decisions on unsold seats. This is what is called "yield management." Today yield management is practiced widely not only in the airline industry, but also in other industries such as hotels, etc.

Many retail giants have established private satellite networks using small-dish antennae placed on store roofs to receive and transmit masses of data. Such a communication network allows the firm to coordinate its multisite operations so as to realize substantial benefits. The instant transmission about the rate of sales, inventory status, product updates, etc., provides valuable information in a timely fashion for decision making.

Wal-Mart manages not only one of the world's largest chain of retail stores, but also one of the world's largest data warehouses. It will also squeeze even more value from these systems with a new data-mining application that will help it replenish inventories in stores. The systems house data on point of sale, inventory, products in transit, market statistics, customer demographics, finance, product returns, and supplier performance. The data can help Wal-Mart analyze trends, manage inventory, and have a better understanding of its customers.

C. E-Service

According to Chase and colleagues, e-service is the delivery of service using new media such as the Web. The spectrum of e-commerce ranges from selling goods with little or no service content to providing (pure) services on the Web. In between there are value-added services (e.g., on-line travel agents) and products sold with a high service content (e.g., ordering on-line customized computers). (Also see Section VI.C for discussion on e-ops).

VI. PERSPECTIVES ON FUTURE DEVELOPMENT

The discussion in the previous sections suggests that the rapid advancement of information technologies changes not only the ways information systems support operations management, but also the ways operations are managed. This trend will inevitably continue. Several issues are important to future developments in information systems for operations management.

A. ERP for Decision Support

First of all, ERP is expected to continue to grow. Undoubtedly, more business functions need to and will be integrated into the system. We should also expect more decision support capabilities to be built into an ERP. In other words, the role of ERP will not be limited to providing information and facilitating transactions. It will also be the key decision support system for operations management and for organizational management in general. That is, the DSS for operations strategy and the general SDSS should be incorporated in the ERP system.

B. Strategic Information Flows

Earlier we discussed the evolution of information systems for manufacturing planning and control—from COPICS to MRP, MRP II, and then to ERP. Such an evolution signifies the expansion of application scope and the increase of systems capabilities. Future "evolution" will necessarily require us to completely change our mindset. Traditionally, the concept of manufacturing management puts material flows first, while information flow plays a supportive role to the manufacturing function. Cook and colleagues point out that, as competition in the marketplace intensifies, managers in manufacturing firms need to think strategically. Thinking strategically requires that top priority be given to managing information flows properly. The flow of materials (i.e., the manufacturing function and the coupled logistics system) becomes only part of the firm's resource deployment aimed at achieving the strategic goals derived from the information flow.

C. E-Ops

Electronic commerce (EC) is another important development that will drastically change the production and delivery of goods and services. Electronic commerce and its central tool, the Internet, are heavily affecting operations and every other facet of business. Chase, Aquilano, and Jacobs (2000) coined the term e-ops (e-operations) to describe the application of the Internet and its attendant technologies to the field of OM. With e-ops, the infrastructure of a modern organization is viewed as an entity consisting of three conceptual levels:

1. *The business model level* Defines the organization's core business and strategy
2. *The operational level* Defines the processes needed to implement the strategy
3. *The information systems (IS) architecture level* Provides computer technology support to the above two levels

E-ops applications can be used for the internal processes of an organization, such as back-office support and the common factory processes of ordering, scheduling, product design and development, quality monitoring, etc. E-ops can also be used for processes that support the external linkages between a company's suppliers and customers.

D. Virtual Operations

In Section III.B, we briefly discussed the notion of the "virtual factory." It should be noted that "factory" does not necessarily imply "manufacturing." In fact, virtual factory, virtual operations, virtual corporations, etc., are conceptually equivalent. Literally, a virtual corporation can contract out all phases of the design, production, distribution, and sales of its products and/or services. Many SCM or ERP software vendors claim

that their products support virtual operations. Regardless of the validity of these claims, any manufacturing or service firm (if such a distinction is still necessary or meaningful) should rethink its business strategy and information architecture.

E. Globalization

Even without virtual operations, globalization is a reality and a must for many industries. Although terms like global market, global manufacturing, global sourcing, global supply chain, etc., are not new, globalization still poses a tremendous challenge for firms to effectively design and efficiently implement information systems to support their global operations. Indeed, e-ops, ERP, virtual operations, etc., should all be built around the architecture of information systems for global operations and vice versa.

SEE ALSO THE FOLLOWING ARTICLES

Benchmarking • Computer-Aided Design • Computer-Integrated Manufacturing • Enterprise Resource Planning • Productivity • Supply Chain Management • Total Quality Management and Quality Control • Value Chain Analysis

BIBLIOGRAPHY

Chase, R. B., Aquilano, N. J., and Jacobs, F. B. (2000). *Operations management for competitive advantage,* Ninth Edition. New York: McGraw-Hill/Irwin.

Chung, C. H. (1991). Planning tool requirements for flexible manufacturing Systems, *Journal of Manufacturing Systems,* 10(6), 476–483.

Chung, C. H., Lang, J. R., and Shaw, K. N. (1989). An approach for developing support systems for strategic decision making in business *OMEGA,* 17(2), 135–146.

Chung, C. H., and Luo, W. H. (1996). Computer support of decentralized production planning. *The Journal of Computer Information Systems,* 36(2), 53–59.

Cook, D. P., Chung, C. H., and Holsapple, C. W. (1995). Information flow first, material flow next! *APICS—The Performance Advantage,* pp. 38–39.

Cook, D. P., Maupin, D. J., and Chung, C. H. (1998). Strategic decision support systems for time-based competition. *The Journal of Computer Information Systems,* 38(4), 26–33.

Fitzsimmons, J. A., and Fitzsimmons, M. J. (1998). *Service management: operations, strategy, and information technology,* Second Edition New York: McGraw-Hill/Irwin

Handfield, R. B., and Nichols, E. L., Jr. (1999). *Introduction to supply chain management.* Upper Saddle River, NJ: Prentice Hall.

Hart, C. W. L., Heskett, J. L., and Sassar, W. E., Jr. (1990). *Service breakthroughs: Changing the rules of the game,* New York: Free Press.

Martin, A. J. (1983). *Distribution resource planning: Distribution management's most powerful tool.* Essex Junction, VT: Oliver Wight Ltd.

Mathieu, R. G. (1996). *Manufacturing and the Internet.* Norcross, GA: Institute of Industrial Engineers.

Nersesian, R. L. (2000). *Trends and tools for operations management: An updated Guide for executives and managers.* Westport, CT: Quorum.

Orlicky, J. (1975). *Material requirements planning.* New York: McGraw-Hill/Irwin.

Shtub, A. (1999). *Enterprise resource planning: The dynamics of operations management.* Boston, MA: Kluwer Academic.

Skinner, W. (1969). Manufacturing—The missing link in corporate strategy. *Harvard Business Review,* 47(3), 113–119.

Stalk, G., Jr., Evans, P., and Shulman, L. E. (1992). Competing on capabilities: The new rules of corporate strategy. *Harvard Business Review,* 70(2), 57–69.

Vollmann, T. E., Berry, W. L., and Whybark, D. C. (1998). *Manufacturing planning and control systems,* Fourth Edition. Homewood, IL: Irwin.

Wight, O. (1984). *Manufacturing resource planning: MRP II,* Essex Junction, VT: Oliver Wight Ltd.

Optimization Models

David L. Olson
University of Nebraska, Lincoln

GLOSSARY

assignment models Optimization model to assign each entity to exactly one position.
chance constrained programming Mathematical programming where constraints include parameters described by distribution functions, yielding constraints that are satisfied at a prescribed level of probability.
dual price (marginal value; shadow price) The marginal change in the objective function per unit increase in a right-hand side coefficient.
dynamic programming Formulation and solution of multistage problems, usually in the form of a sequence of events or probabilities.
goal programming Optimization of models with different target values for multiple objectives.
integer programming Optimization with solution values of decision variables required to have nonfractional values.
linear programming Mathematical modeling technique to optimize sets of linear relationships.
multiple objective optimization Methods optimizing combinations of multiple objectives.
network optimization Optimization models described as networks of flow from sources to destinations.
nonlinear programming Mathematical modeling technique to optimize nonlinear sets of relationships.
sensitivity analysis Analysis of how much an objective function coefficient, constraint limit, or functional coefficient change will affect the current solution.
stochastic programming (probabilistic programming) Optimization of models including uncertain elements, described by probabilities or probability distributions.
transportation model Optimization model to assign quantities to combinations of given sources and demands.

OPTIMIZATION MODELS are presented, including discussion of general mathematical programming formulation. The primary optimization modeling tool, linear programming, is described including required assumptions. Alternative optimization model forms such as nonlinear programming, integer programming, network optimization, transportation models, assignment models, other network models, and dynamic programming are discussed. Methods considering multiple criteria are identified. Major industrial applications are reviewed, including those for various information system problems. Sensitivity analysis is briefly described, along with a short discussion of recent solution method advances.

Optimization models are mathematical expressions of decision problems that can be solved to identify the best possible decision in terms of the objective function stated. Optimization models are useful as tools within information systems for decision-maker support, as well as analytic tools to better design information systems.

I. INTRODUCTION

Linear programming is one of the most powerful analytic tools available for decision support systems. Linear

programming provides a means to generate solutions that are optimal for the given objective function. Not only is the best possible decision (relative to the objective function) provided, but a great deal of economic interpretation of the limits to the decision are available. However, the conclusions to be drawn from linear programming are highly sensitive to the accuracy of the model. Errors in input data, or changes in demands, costs, or resource usage, can make major differences in model results. This is because by definition, linear programming seeks the very best possible solution, and thus optimal solutions are at extreme limits of at least some constraints.

A. Optimization

Optimization is a concept that has proven extremely useful in business decision making. If a decision problem can be modeled in mathematical form, solution of this model yields the best possible decision. Various mathematical forms require different techniques for solution. If the decision can be modeled by a simple function, such as the inventory order quantity under special conditions, the derivative of this function can be used to identify the optimal order quantity. Most decisions involve constraints on the decision in one form or other, resulting in models of constrained optimization.

Mathematical programming is the modeling of a problem in terms of variables and constraints, and applying a procedure to identify the optimal solution to the model. This structure makes it possible to consider much more complex systems than was the case with classical optimization of functions such as calculus. Mathematical programming can be applied to large systems of variables and relationships.

B. Forms of Mathematical Programming

There are a variety of solution methods that have been developed for various forms of general mathematical programming. The most widely used is linear programming, because of the ready availability of the simplex method (and other methods) for solution, along with the applicability of linear models to reflect many important decisions, such as connection of information systems components, telephone and cable wiring plans, and other decision problems. Table I shows some of the major categories of mathematical programming. Some linear programming models require restriction of solution values for variables to be integer or binary (zero-one). Special forms of linear programming, such as network optimization, transportation, and assignment models involve restricted types of constraint expression that can be solved by methods even more efficient than the general simplex method. There are some techniques that consider multiple objectives, including goal programming and vector optimization. Other models involve nonlinear functions.

Table I Forms of Mathematical Programming

Decision environment	Typical techniques	Special conditions
Certainty	Linear programming	
		Integer programming
		Zero-one programming
		Network optimization
		Transportation models
		Assignment models
		Shortest path models
		Minimum spanning tree
		Maximum flow
		Critical path
		Multiple objective models
		Goal programming
		Vector optimization
	Nonlinear programming	
	Quadratic programming	
Risk	Stochastic programming	
		Chance constrained programming
	Some types of dynamic programming	

C. General Mathematical Programming Formulation

The general mathematical programming model is:

$$\begin{array}{lll} \text{Maximize} & f(x_j) & \text{for } j = 1 \text{ to } n \\ \text{Subject to:} & g_i(x_j) \leq b_i & \text{for } i = 1 \text{ to } m \end{array}$$

where n is the number of variables, and
m is the number of restrictions.

Constraint functions here are shown as ≤, but can also be ≥ or = relationships. The constraint set may include finite upper bounds and/or lower bounds on specific variables x_j .

II. LINEAR PROGRAMMING

As stated above, the most commonly applicable model form is linear programming, where all functions $f(x_j)$ and $g_i(x_j)$ must be linear.

A. Linear Programming Formulation

$$\text{maximize} \sum_{j=1}^{n} c_j x_j$$

$$\text{subject to:} \sum_{j=1}^{n} a_{ij} x_j \leq b_i \text{ for } i = 1 \text{ to } m$$

$$x_j \geq 0 \text{ for } j = 1 \text{ to } n$$

Linear programming models can be minimized. This formulation still applies, by allowing objective function coefficients c_j to take on negative values. Constraints are allowed to be ≤ or = as shown, as well as ≥ or =, or strictly equal. Sometimes decision variables x_j are allowed to take on negative solution values. This can be accomplished through several methods, including modeling a duplicate set of variables, or modification of the solution technique.

Dantzig presented the simplex method for solution of this form of mathematical programming model. The simplex method is well studied and widely available. (Dantzig and his research group, working for the United States Air Force, developed the simplex method about 1947, and first published it in 1951.) This linear model is used in most of the applications we will discuss. However, other sets of assumptions allow more generality in modeling, although usually with correspondingly more complex solution problems.

B. Assumptions

A key element of linear programming (LP) models is the set of assumptions required. These assumptions are *linearity, certainty,* and *continuity.* It is not necessary to assume a single objective, although it must be realized that the optimal solution obtained is only optimal with respect to the function used as the objective.

Linearity—For linear programming, all functions (constraints and objective function) must be linear. This is often not a problem. However, objective functions may involve economies of scale, or diminishing returns, both leading to nonlinearities. Further, variables may have interactions, another source of nonlinearity.

Nonlinear programming—There are some decision models that involve nonlinear functions, either as objective functions or as constraints. These are more difficult to solve than linear models, but software capable of solving larger nonlinear models is becoming available on a more widespread basis. There are algorithms available for nonlinear optimization should some or all model functions be nonlinear. Lasdon and co-authors presented the generalized reduced gradient method (GRG) in 1978. This method is widely used for general nonlinear models. Models with specific forms of nonlinearity can be solved using special algorithms.

Certainty—The resulting LP solution will be optimal *if* the coefficients used are accurate and constant. There are three classes of coefficients in the formulation given above. If contribution coefficients (c_j) are estimates, or are random variables, you will get a feasible solution, but you are not guaranteed the best possible solution. If the coefficients b_i (right-hand side values) or the technological coefficients a_{ij} are estimates with some variance, the solution may not be feasible when implemented. There is a certain degree of sensitivity analysis which can be conducted to determine how much c_j or b_i coefficients can vary before it makes any difference. There is also a limited amount of sensitivity analysis that can be accomplished if a_{ij} coefficients vary. The validity of the resulting solution depends upon the accuracy of the model coefficients.

Stochastic programming—Linear programming methods require an assumption of certainty. If coefficients involve some risk, this introduces a form of nonlinearity. Stochastic programming is a form of nonlinear programming where the nonlinearity is due to randomness in model coefficients. Chance constrained programming is a specific form of stochastic programming, where constraints are viewed as to be satisfied a specified proportion of the time. In addition to

distributions on model parameters, there are also other causes for nonlinearity in nonlinear programming models under conditions of certainty. Dynamic programming models can involve complexity under conditions of certainty. Often dynamic programming models are applied to problems involving risk. Dynamic programming techniques are sensitive to model size (the curse of dimensionality). Birge provided a review of stochastic programming methodologies in 1997. The size of model that can be solved is much smaller than is the case with linear programming.

Continuity—Standard linear programming models assume continuity, allowing decision variables to take on continuous values. In some decision models this is not appropriate. Since the variable values in linear programming are the result of simultaneously solving equations, the optimal solution may well contain fractional values for decision variables. Often fractional solution values are not a problem, as rounding provides a useful answer to the decision problem. However, rounding will not guarantee the best integer or zero-one solution.

Integer and zero-one programming—There are some models, such as capital budgeting, portfolio selection, project scheduling, and fixed cost that require decision variables in solutions to be either 0 or 1. There also are mixed formulations where some of the decision variables must be integer or 0-1, such as inventory problems, or links between terminals in telecommunications networks. These problems require integer programming or zero-one programming techniques.

There are solution techniques which identify optimal integer or zero-one decision variable values. Nemhauser and Wolsey provided an overview of these methods. The primary methods are branch and bound, cutting plane, and for zero-one models, implicit enumeration. Table II displays the assumptions required in linear programming, along with techniques that deal with cases where these assumptions do not apply. Special cases of these techniques are given in the third column.

C. Special Forms of Linear Programming

Just as there are special techniques to optimize models with complicating assumptions (requiring application of less efficient solution methods), there are also solution techniques more efficient than simplex that are available for models with special characteristics.

1. Network Optimization

Network optimization provides the ability to solve larger models than can be done with the general simplex method, at greater speed than is possible with the general simplex method. Glover and Klingman presented solution techniques for models that can be described as a network of flows between layers of nodes. Network models involve restrictions on the type of constraints that can be included.

Generalized network models include constraints for balance equations, where inflow to a node equals outflow, in addition to supply and demand constraints. The more flexible the model formulation, the greater the sacrifice in solution speed. Generalized network models allow shrinkage and expansion, so they allow technological coefficients a_{ij} and the formulation appears the same as for a general linear programming model. However, general linear programming models allow other types of constraints in addition to balance equations.

Maximize $c_1x_1 + c_2x_2 + \ldots + c_nx_n$

Subject to: $a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1$

can be $\geq$, $=$ relationship

Table II Linear Programming Assumptions and Other Techniques

Assumption	Techniques dealing with exceptions	
Linearity	Nonlinear programming	Generalized reduced gradient Quadratic programming
Certainty	Stochastic programming Dynamic programming	Chance constrained programming
Continuity	Integer programming Cutting planes	Branch-and-bound method
	Zero-one programming	Branch-and-bound method Implicit enumeration

$$a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leq b_2$$

$$\ldots$$

$$a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \leq b_m$$

$$x_j \geq 0 \quad \text{for all } j$$

A pure network model has technological coefficients a_{ij} limited to values of 0, 1, or -1. Generalized network algorithms are more efficient than the general simplex method, while pure network algorithms are more efficient than generalized network algorithms.

2. Transportation Models

As more restrictions on the form of the model are applied, generalized network models include (with other variants along the way) transportation models. Hitchcock presented this type of model in 1941. In transportation models, variables are the quantities shipped from each source to each demand. Functions have variable coefficients (technological coefficients a_{ij}) of 1 (with 0 used if a combination of source and demand is blocked). The objective function coefficients represent the cost of shipping from each source to each demand. Constraint right-hand sides represent demand requirements and source availabilities.

Minimize $c_1x_1 + c_2x_2 + \ldots + c_nX_n$

Subject to: $x_1 + x_2 + \ldots + x_n \geq b_1$

can include $\leq$, $=$ relationships

$$x_1 + x_2 + \ldots + x_n \geq b_2$$

although at least one must be $\geq$

$$\ldots$$

$$x_1 + x_2 + \ldots + x_n \geq b_m$$

$$x_j \geq 0 \quad \text{for all } j$$

3. Assignment Models

Assignment models were developed by D. König, with one early report by Kuhn in 1955. Assignment models are a special form of transportation model, with all constraint right hand sides having values of 1. Variables represent the assignment of an object to a specific task. Both transportation and assignment models can be solved very efficiently by methods other than simplex.

Maximize $c_1x_1 + c_2x_2 + \ldots + c_nx_n$

Subject to: $a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = 1$

$$a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = 1$$

$$\ldots$$

$$a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = 1$$

$$x_j = 0 \text{ or } 1 \quad \text{for all } j$$

4. Other Network Models

There are a number of special algorithms that apply to specific types of network models. Networks of nodes and arcs represent activities or processes and their possible connections. The three most widely modeled decisions in this category are shortest path models, minimum spanning tree models, and maximum flow models. Special solution techniques exist for each type of model.

Shortest-route networks model the possible paths that an entity can take to get from an originating point to a destination, given connections with distances. The objective is to minimize distance (or time, or cost, or other measures). An example would be routing a message through a network. A number of different node connections would exist, each with measurable distances and costs.

The minimum spanning tree problem requires connecting all nodes on the system in the cheapest way possible. An example is wiring a computer network, seeking minimum cable. The model solution is a tree connecting all points in the network. Many real problems of this type exist, including determining the minimum length of transmission lines required to connect all service areas of an electric utility, cable to connect all subscribing homes to a television cable system, canals to link an irrigation system, and pipelines to connect terminals.

The maximum flow problem involves networks where branches have limited flow capacities. The objective is to determine a particular route that would maximize the total flow from a source to a specified destination. Real problems include traffic flow over a highway system, water flow through irrigation systems, oil flows through pipelines, electricity flows through transmission networks, and telephone call flows through telecommunications systems.

Critical path models are special kinds of networks used in project scheduling. Critical path solutions identify the fastest completion time for a project, as well as the slack time available for each activity. Inputs are the list of activities, their estimated durations, and predecessor relationships. The assumption of certainty is very limiting for these models, as durations involve constant change in many practical environments. However, the analysis is useful for planning.

5. Multiple Criteria Optimization

Real decisions often involve consideration of multiple criteria. There are two broad approaches that have been applied to this type of decision problem. Charnes and Cooper presented goal programming, using target values for each objective function. Lee used preemptive goal programming, using the priority structure obtained from decision-maker preference expressed through prioritizing objective-target pairs. Alternative multiple criteria optimization methods were given by Steuer in 1986, including optimization of a combined weighted objective function reflecting all criteria simultaneously. Target objective function values can be set at ideal values. There are alternative optimization models reflecting multiple criteria, such as the Step Method.

6. Nonlinear Optimization

A number of optimization problems require relaxation of the assumption of linearity. Nonlinear programming is a diverse field, with a number of techniques available for specific circumstances. These techniques are for the most part extensions of differential calculus. For unconstrained functions, as in economic order quantity calculation in inventory management, calculus is sufficient for solution. To guarantee solution, functions must be unimodal. Some problems involve nonlinearity in the objective function in the form of quadratic functions, subject to linear constraints. In this case, quadratic programming methods exist for solution. Lagrange multipliers can be used to solve restricted classes of nonlinear programming models.

7. Dynamic Programming

Dynamic programming is a mathematical modeling theory that is useful for solving a select set of problems involving a sequence of interrelated decisions. Dynamic programming provides a systematic means of solving multistage problems over a planning horizon or a sequence of probabilities. As an example, a stock investment problem can be analyzed through a dynamic programming model to determine the allocation of funds that will maximize total profit over a number of years. Decision making in this case requires a set of decisions separated by time.

Dynamic programming is a powerful tool that allows segmentation or decomposition of complex multistage problems into a number of simpler subproblems. These subproblems are much easier to handle than the complete problem. Solution of specific forms of dynamic programming models have been computerized, but in general, dynamic programming is a technique requiring development of a solution method for each specific formulation. These solutions are often not difficult, and can be supported by simple technology such as spreadsheets. The key is to develop the dynamic programming model. The number of stages involved is a critical limit.

III. APPLICATIONS

Linear programming models are among the most profitable quantitative analytic models applied to business decision making. While not all decision problems have the required conditions of linearity and certainty required, there are some very important applications that do.

A. Standard Linear Programming Applications

Objective functions can reflect anything that the modeler wishes to optimize. In technical applications, the objective function may be computer time, memory required for storage, or meters of cable. Many optimization models relate to business decisions.

Profit is generally accepted as the paramount concern of business, and is often the ideal objective function. However, accurate profit data may not be readily available. Therefore, other objectives, such as minimizing cost, minimizing time, or maximizing sales are often used for specific analyses. Further, as society becomes more complex, and as we need to think in terms of longer range impacts of decisions, factors other than profit need to be considered as well. Individual decision makers have always had to balance trade-offs in objectives, such as profit, satisfaction, living conditions, etc. Companies are having to consider more complicated combinations of objectives, such as public opinion, reliance upon suppliers, labor relations, and many other elements which do not easily translate to functions compatible with profit. Standard model types that have proven widely useful are given in Table III.

B. Petroleum Industry Applications

The petroleum industry has long relied upon linear programming models to aid in production scheduling. This industry faces a wide variety of refining options, well-defined limits to production, and a highly dynamic decision environment. Refinery operations

Table III Common Linear Programming Applications

Diet models—How much of each ingredient to include Minimize cost of ingredients such that nutritional requirements are met
Capital budgeting—What to invest in at what time Maximize return subject to budget limits, cash flow requirements
Product mix—What to produce Maximize profit subject to budget and resource limits
Blending models—What to produce with what ingredients Maximize profit subject to resource limits and contractual requirements
Work scheduling—How many people work at what times Minimize cost subject to adequate staffing
Inventory models—What to produce to stock or purchase at what time Minimize cost subject to having sufficient stock available by time period
Media planning—What to advertise where and when Maximize exposure subject to budget limitations
Input/output models—"Optimal" assignment of production responsibilities

involve many equations representing the physical and chemical relationships of input and output processes. There are a variety of input sources in most cases, and many products that can be produced from these inputs. Prices for inputs and products vary daily. Refinery models are often large in scale, and are usually run on a regular (daily) basis. Some petroleum planning models have involved economic relationships between price and volume reflecting price elasticity, resulting in nonlinear models.

C. Airline Industry Applications

Optimization applications have been very profitable in the airline industry. Airline operations include many problems requiring intelligent management, including selecting routes to cover demand for services, scheduling crews to provide professional service without overtaxing human endurance, assigning high-capital equipment efficiently, and scheduling arrival slot allocation in busy airport terminals. Network optimization has been very effective in many of the scheduling applications of the airline industry, sometimes applied to models in excess of one million variables. Nonlinear optimization also has been effectively applied in dealing with yield management, the intelligent pricing of tickets to optimize airline profit.

D. Decision Support System Applications

Optimization models have been applied in many business and engineering applications to great effect. Models have been incorporated within decision support systems to support critical decisions at both strategic and tactical levels of management. At the strategic level, allocation of productive capacity to different products has long been an effective linear programming application. A typical kind of tactical application of this type is daily determination of production plans in operations such as paper cutting. Decision support systems often involve the need for special kinds of optimization models, containing either network features, nonlinear model components, or both.

Sometimes special technology can be incorporated into systems including optimization models. Geographic information systems can be used as the basis of vehicle routing and scheduling systems. Sears reportedly saved over $42 million annually by using such a system to dispatch technicians and home deliveries. Geographic information system technology is also widely used in healthcare and emergency planning, controlling the use of ambulances and other emergency vehicles. It also has been applied at the strategic level. Proctor & Gamble has applied integer programming, network optimization, and a geographic information system to reduce its number of plants in North America by about one-fifth, claiming savings of over $200 million per year.

E. Information System Applications

Optimization models have been widely applied to information system design problems. Linear programming models have been used to improve the efficiency of file allocation in distributed information systems.

The objective function of this type of model is to minimize the differences between response times of servers. Reference librarians also have technology available using optimization models to identify search strategies. Some file allocation and query routing models for distributed information systems involve multiple objective optimization models.

Telecommunications involve many networks, which can be optimized to minimize cost. More and more the reliability aspects of such networks are modeled, resulting in models that ensure desired levels of redundancy through constraints.

IV. SENSITIVITY ANALYSIS

Sensitivity analysis refers to determining how much any one coefficient could change before model results would change. Most sensitivity analysis focuses on only one coefficient change at a time in changes in contribution rates (c_j) and right-hand side values (b_i). More advanced techniques are required to analyze the impact of changes in technological coefficients (a_{ij}). If multiple changes are expected simultaneously, what-if analysis in the form of rerunning models with various combinations of changes has to be applied. There are some valuable concepts available from sensitivity analysis.

Each constraint limits the model solution. If it weren't for a particular constraint, it is possible that the objective function would be better. Simplex, the basic solution method for linear programming, is essentially solution of simultaneous equations. For a model with n variables and m constraints, adding a slack or surplus variable to each constraint yields a system with m equations in $m + n$ variables. Simplex selects precisely m of these variables to be basic, or allowed to take on nonzero values, which are determined by solving the subset of m equations in m unknowns. If the equations are independent, there is precisely one solution to this system. For each constraint in a given solution, there are two possibilities: the constraint is at its limit (binding), or the constraint is not at its limit (nonbinding). A nonbinding constraint has slack (for $\leq$ constraints) or surplus (for $\geq$ constraints). A nonbinding constraint implies that the slack/surplus variable associated with that constraint is basic.

A. Reduced Costs

Reduced costs are by definition the amount that a decision variable contribution coefficient must improve before that decision variable would be introduced into the solution. In more mathematical terms, reduced costs are the amount that a c_j must improve before it is attractive enough to be part of the basic solution (take on a nonzero value). The optimal decision will remain the same as long as any one c_j stays within its allowable range, and no other model coefficients change. In effect, a reduced cost is how much a product is underpriced. For minimizations, it is how much a variable is overpriced. Nonzero reduced costs are only possible for nonbasic variables.

Sensitivity analysis of contribution coefficients is only useful for change in one contribution coefficient, with the exception of the 100% rule. The 100% rule provides a small bit of added power to determine if an optimal solution would change under conditions of *multiple* changes in contribution coefficients.

$$\sum_{j=1}^{n} \frac{change(c_j)}{allowable\ change\ in\ c_j}$$

If the sum of ratios of changes in contribution coefficients divided by their allowable changes over all j decision variables total no more than 1.0, the 100% rule establishes that the current optimal solution would still be optimal given the changes. If the sum of changes is greater than 1.0, nothing is proven. In that case, you would not know if the current solution would remain optimal or not.

B. Dual Prices

The definition of a dual price is the rate of change in the objective function per unit change in right-hand side coefficient for a particular constraint. The dual price will remain constant as long as the associated b_i stays within its allowable range, and no other model coefficients change. Economic interpretation of dual prices are useful as marginal values of resources, or marginal costs of requirements.

The dual price for a nonbinding constraint is by necessity 0. Because the constraint is not at its limit for the current solution, changing the right-hand side less than the slack/surplus value would have no impact upon the optimal solution. Therefore, the rate of change in the objective function per unit change in right-hand side would be zero within this limit. For binding constraints, the rate of change is equal to the dual price until a new constraint is activated. The range of right-hand side change over which the current dual price is valid is reported by most linear programming software.

V. RECENT ADVANCES IN SOLUTION METHODS

Other classes of problems can be modeled with linear programming. Transportation models, assignment models, and critical path scheduling models could all be solved with linear programming. There are, however, more efficient means of solving these special classes of problems than the conventional linear programming solution technique (simplex). The types of decisions that can be analyzed are limited by imagination, as well as the need to assure that the required assumptions for linear programming are valid in the situation being modeled.

Khachian proposed an ellipsoidal algorithm as a means to solve linear programming models alternative to simplex. This algorithm is polynomial, meaning that it has a much more reasonable upper bound on solution effort than is true of the simplex method. Karmarkar developed an algorithm with an even better worst-case complexity. Karmarkar's algorithm has been developed into a system for solving large-scale linear programs by AT&T in a product named KORBX. The KORBX system uses parallel processing technology including primal, dual, primal-dual and power series algorithms. Primal algorithms work on the original linear programming model presented earlier. A dual model is an alternative mathematical expression of a linear programming problem that will yield the equivalent solution to the primal model. A dual model for the primal model given in Section II.A is:

$$\text{minimize} \sum_{i=1}^{n} b_i u_i$$

$$\text{subject to:} \sum a_{ji} u_i \geq c_j \text{ for } j = 1 \text{ to } n$$

$$u_i \geq 0 \text{ for } i = 1 \text{ to } m$$

When there are many variables and few constraints, models can be solved faster than when there are many constraints and few variables. If a model consists of many constraints and few variables, its dual formulation is easier to solve. Some computer codes attack a particular model with both forms used in parallel (primal-dual). Work continues on development of new algorithms, and utilization of new computer technology in the solution of mathematical programming models.

VI. CURRENT RESEARCH AREAS

Modeling decision problems is a matter of imagination. The problem must be recognized, and a mathematical formulation developed. It is helpful to view models of similar problems, such as are given in Table III. There are constantly new formulations that are valuable in many areas of decision making, because there are constantly new developments in technology and in the way in which business is conducted. This research is done by users, who have real problems that need solutions.

Technical research into new solution techniques continues as well. This research is becoming ever more concentrated and specialized. Section V reviewed the major developments in solution of linear programming. There also have been advances in solution in the airline industry with respect to the size of model that can be usefully solved, as discussed in Section III.C. Work also continues on techniques for more efficient solution of special mathematical programming types, such as nonlinear, integer, and network programming.

SEE ALSO THE FOLLOWING ARTICLES

Goal Programming • Statistical Models (Non-Optimization Models) • Uncertainty

BIBLIOGRAPHY

Birge, J. R. (1997). *Introduction to stochastic programming.* New York: Springer.

Gass, S. I. (1985). *Linear programming: Methods and applications.* Danvers, MA: Boyd & Fraser.

Hillier, F. S., Hillier, M. S., and Lieberman, G. J. (2000). *Introduction to management science.* Boston, MA: Irwin McGraw-Hill.

Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. *Combinatorica,* Vol. 4, No. 4, 373–395.

Nemhauser, G., and Wolsey, L. (1988). *Integer and combinatorial optimization.* New York: John Wiley & Sons.

Steuer, R. E. (1986). *Multiple criteria optimization: Theory, computation, and application.* New York: John Wiley & Sons.

Organizations, Information Systems Impact on

George P. Huber *University of Texas*
C. Brad Crisp *Indiana University*

GLOSSARY

computer-aided technologies Devices that employ a computer to transmit, display, manipulate, analyze, store, or use information.
computer-mediated communications Messages conveyed through a computer-aided technology.
infinitely flat organization An organization consisting of two layers, a headquarters unit and a tier of nearly identical operating units.
information technologies Devices that employ a computer to transmit, display, manipulate, analyze, store or use information as part of a communication or decision-making process. This definition is intentionally specific to the context of this chapter, and is therefore narrower than the definition used by most information professionals and social scientists.
organizational intelligence The output or product of an organization's efforts to acquire, process, and interpret information from outside of the organization.
organizational network A set of organizations that communicate with one another for collaborative purposes.
rich media Media that can convey a wide variety of complex stimuli and that permit rapid feedback about the effects of sent messages.
virtual organization A set of independent organizations involved in a collaborative relationship.

Incorporating, updating, and extending earlier work by Huber from 1990, this chapter describes the effects of information technologies on organizations, where *information technologies* are narrowly defined, here, as *devices that employ a computer to transmit, display, manipulate, analyze, store, or use information as part of a communication or decision-making process*. Specifically, this article considers the effects of computer-aided communication and decision-support technologies on organizational design, intelligence, and decision making. The first section introduces the role of information technologies in organizations. Subsequent portions of the chapter overview the effects of information technology at three levels of organizational design and on organizational intelligence and decision making. The final section summarizes the earlier content and describes opportunities for future research.

I. INFORMATION TECHNOLOGIES IN ORGANIZATIONS

Everyday observation makes clear that use of information technology influences organizational structures and processes. The Internet, for example, enabled faster and broader communications both inside and outside of organizations and across organizational boundaries. As a result, it changed not only the means of interaction but also the individuals and groups involved in a particular organization's affairs. However, beyond this general observation that information technology makes a difference in organizations, managers and information systems professionals require a more

specific understanding of how information technology affects organizations, an understanding grounded in systematic study and scholarship.

Space constraints limit this article to communication and decision-support technologies, excluding, for example, production-assisting technologies. Further, the constraints limit this article's focus to the effects of information technologies on organizations, as contrasted to the effects on individuals within organizations or on industries or other elements of society. Finally, within the context of organizations, constraints limit this article's content to the effects of information technology on three specific organizational features—organizational design, intelligence, and decision making.

The information technologies considered here have either made their appearance since 1970 or exist in a much more influential form than they did prior to 1970. They almost always provide higher levels of *basic characteristics*—such as data storage capacity, transmission capacity, and processing capacity—than do devices or technologies unaided by a computer. Although these basic characteristics are relevant to users, often it is the particular configuration of the levels of these characteristics in an information technology that makes the technology appropriate for a particular task. Some authorities have considered these configurations when comparing today's information technologies with traditional information technologies, and have made generalizations about the resultant *properties* of information technologies. Because these properties cause the use of information technologies to have effects such as those noted in this article, some of these generalizations are reviewed here.

A. Communication Technologies

Communication technologies (or communication media) have increased in number and effectiveness over the past few centuries (e.g., signaling flags, the Pony Express, telegraph, radio, and telephone). Computer-aided communication technologies (e.g., electronic mail, image transmission devices, computer conferencing, and videoconferencing) extend this trend and facilitate access to people inside and outside the organization with an ease not previously possible. Properties of these technologies include those that facilitate the ability of the individual or organization (1) to communicate more easily and less expensively across time and geographic location, (2) to communicate more rapidly and with greater precision to targeted groups, (3) to record and index more reliably and inexpensively the content and nature of communication events, and (4) to more selectively control or extend access and participation in a communication event or network.

B. Decision-Support Technologies

More sophisticated, user-friendly forms of computer-assisted decision-support technologies (e.g., expert systems, decision-support systems, on-line management information systems, and external information retrieval systems) have also appeared. Properties include those that facilitate the ability of the individual or organization (1) to store and retrieve large amounts of information more quickly and inexpensively; (2) to more rapidly and selectively access information created outside the organization; (3) to more rapidly and accurately combine and reconfigure information so as to create new information (as in the development of forecasting models or financial analyses); (4) to more compactly store and quickly use the judgment and decision models developed in the minds of experts, or in the mind of the decision maker, and stored as expert systems or decision models; and (5) to more reliably and inexpensively record and retrieve information about the content and nature of organizational transactions.

C. Relationships with Traditional Technologies

Occasionally an unstated assumption is encountered—that information technologies are universally inferior or superior to traditional communication or decision-support technologies such as highway billboards, face-to-face meetings, or heuristics. This impression is mistaken because the properties delineated above may be less important than other properties possessed by a more traditional technology, such as social acceptability, ease of use, the ability to convey emotion precisely, or the ability to register information in human memories through vividness or contextual cues, or a combination of these properties. The impression is also mistaken because traditional technologies may have "scores" on these properties that overlap with the scores of information technologies, thus making unreasonably simplistic claims of universal inferiority or superiority.

In a related vein, it is a mistake to view information technologies solely as substitutes for traditional technologies. To the contrary, information technologies

are frequently used as supplements and complements to traditional technologies, rather than as substitutes. For example, electronic mail is often used to confirm with text what was said in a phone conversation or to set up face-to-face appointments, and image transmission devices are often used to make available drawings that will later be discussed face-to-face or in a conference call, after all the parties have had a chance to study them. Of course, people do substitute computer-assisted media for traditional media when it seems efficacious to do so. Overall, the availability of information technologies increases the communicating or decision support options for potential users. As organizational members learn to choose and employ technologies wisely, the effect is to increase the quality (broadly defined) of the user's communication or decision-making processes.

II. INTERACTIONS WITH ORGANIZATIONAL NORMS AND PRACTICES

Before identifying effects of information technologies on organizations, it may be useful to consider the difficulty of establishing cause-effect relationships within and across organizations. As an illustration of this difficulty, George and King in 1991 discussed four underlying assumptions commonly invoked to understand the role of information technology on centralization: (1) the technological imperative, (2) the organizational imperative, (3) the managerial action imperative, and (4) no inherent relationship. In its purest form, the technological imperative suggests that the introduction of information technology *always* leads to centralization (or always leads to decentralization). However, conflicting empirical evidence casts doubt on this approach, leading some to conclude that there is no inherent relationship between computerization and centralization. In contrast to the technological imperative, the organizational imperative proposes that the use and effects of information technology reflect the nature of the organization rather than influencing the organization. A variant of this view says that managerial intent and action shape the use and effects of information technology.

Except for the first, each of these views suggests that organizations are as likely to influence the use of information technologies as they are likely to be influenced by the use of the technologies. While this chapter emphasizes one direction of the relationship between information technology and organizations (i.e., the effect of information technology on organizations), it is important to recognize the reciprocal nature of the relationship and to avoid accepting a technological determinism line of reasoning that ignores context and managerial intent. This important issue was elaborated upon in 1992 by Orlikowski and in 1997 by DeSanctis and Poole.

Without discounting the above, this chapter contends that use of information technologies does tend to generate some patterns of favorable organizational outcomes. This occurs in contradiction to the mistaken belief that, while information technologies may lead to more favorable organizational outcomes (such as information that is more accurate and comprehensive or decisions that are more timely) in organizations characterized by strong adherence to a norm of economic rationality, these outcomes are unlikely in more highly politicized or power-driven organizations. Three observations are offered below that explain why favorable outcomes tend to occur and why the concern with the effects of power and politics is unwarranted.

The first is that the external environments of many organizations are sufficiently competitive that, in order to survive, the organizations must adopt and properly use rationality-enhancing communication and decision-support technologies. If organizational politics interfere with such adoption or use, the marketplace or parent organization intervenes until universal conformance is achieved. Thus, in their time, the telegraph became a pervasive technology in railroads, the calculator in brokerage houses, and the radio in armies. In the organizations that survived, those managers whose personal inclinations caused them not to use the technologies to further organizational goals (such as timely delivery of freight; accurate and comprehensive information for investors; or effective coordination in battle) were evidently converted or purged. In essence, superordinates or organizations require subordinates or subunits to help them compete effectively or otherwise satisfy environmental demands, and if rational use of technology is necessary, it occurs in the long run, whatever the personal inclinations of the subordinates or subunits.

The second observation is that highly politicized or power-driven organizations also have highly competitive internal environments, and in such environments it is necessary for managers to maximize their own competitive effectiveness by appearing to satisfy the goals of resource controllers on an issue-by-issue basis. In these environments, technical or financial analyses are widely used to persuade the resource controllers that the manager's proposals best satisfy the resource controller's goals. Thus, even in organizations where power plays a significant role in resource

allocation, so also do "the numbers." Managers who do not employ the most appropriate technologies in developing and selling analyses are at a competitive disadvantage; they must adapt or lose out.

The third observation is that, in almost all organizations, effective fulfillment of organizational responsibilities contributes to the development and maintenance of a manager's reputation. Thus, aside from whatever a manager might do to negatively or positively affect the quality or timeliness of the design, intelligence, or decision making of superordinate units, he or she is likely to employ communication or decision-support technologies whenever their use can contribute to his or her personal effectiveness or the effectiveness of his or her own unit.

Together, these observations suggest that even though power and politics influence organizational design, intelligence, and decision making, so too do information technologies. *For advancement of their own interests, organizational participants use information technologies to increase their effectiveness in fulfilling organizational goals.* This fundamental assumption underlies the remainder of the chapter. Importantly, the assumption does not rule out the possibility that, for advancement of their own interests, organizational participants use information technology to create an impression of thoroughness or rationality.

Effects of information technology on organizations are here grouped for expositional purposes into four sections. The first three sections portray the effects of information technologies on organizational design, that is, the effects on structure and process at the subunit, organizational, and interorganizational levels. The fourth section sets forth the effects of information technology on organizational intelligence and decision making. As an overview and outline of these four sections, the various effects are summarized in Table I. Of course, there will be exceptions to the effects described. The intention is to imply that across a large number of cases, *ceteris paribus,* there will be a tendency for the described effect to occur.

III. EFFECTS AT THE SUBUNIT LEVEL

The focus in this section is on the effects of information technology at levels of analysis lower than the organization level. The subunit level might include, for example, product design teams, crisis management groups, top management teams, or operating departments.

A. Participation in Decision Making

In many organizational decisions, both technical and political considerations suggest that the development, evaluation, or selection of decision alternatives would benefit from communication among a moderate-to-large number of experts or partisans. But communication takes time and effort, and so the variety and number of participants is often smaller than post hoc analyses determine to be appropriate. Assuming that the time and effort involved in communicating are critical determinants of the number of individuals who participate, the following question arises: What is the effect of computer-aided communication technology on the breadth of participation in decision making?

Because computer-aided communication technology can greatly reduce the effort required for individuals who are separated in time or physical proximity to exchange information, its availability increases

Table I Organizational Attributes Influenced by Information Technologies

Subunit level	Organizational level	Interorganizational level	Intelligence and decisions
Participation in decision making	Centralization of decision making	Size and dispersion	Quality and timeliness of interpretations and intelligence
Size and heterogeneity of decision units	Number of organizational levels involved in authorization	Diversity	Quality and timeliness of decisions and action authorizations
Decision unit processes	Number of nodes in the information-processing network	Variety of configurations	
Temporal and physical proximity	Flexibility and permeability of organizational boundaries		
Effectiveness of meetings			
Frequency of meetings			

the likelihood that people not part of the formal decision unit will serve as sources of information before or during deliberations. Individuals in communities of practice and other informal groups regularly use computer-mediated communication to provide answers to questions and to otherwise communicate with other organizational members—members whom they might never meet in a face-to-face setting. As noted particularly by DeSanctis and Poole in 1994, research shows that, all else equal, availability of computer-aided communication technology leads to greater information sharing within and between groups.

In contrast, some have argued that computer-aided communication technologies make it difficult for decision makers to obtain certain types of information (e.g., soft, rich, contextual, sensitive) or the meaning of information. To the extent that this argument is valid, it could preclude or diminish the use of computer-aided communication technologies where the need for such information is paramount. However, the circumstances where this argument is valid are fewer than what might be imagined. That is, while it is true that computer-mediated communication provides fewer cues than does face-to-face communication, it is also true that managers and other professionals generally choose the communication medium that fits the communication task and the people who are communicating. Oftentimes computer-mediated communication is used to exchange factual or technical information and subsequently other media are used to elaborate on this information or to exchange other types of related information.

The issue is not one of the computer-aided technologies driving out richer media, but rather of the technologies enabling communications that otherwise would be unlikely to occur. For example, the availability of electronic mail has been found to increase the overall amount of communication among organizational members; there is not a one-for-one trade-off between media. Overall, the preponderance of arguments and the available empirical evidence suggest that *use of computer-aided communication technologies leads to a larger number and variety of people participating as information sources in subunit deliberations or operations.*

B. Size and Heterogeneity of Decision Units

In many situations, organizational subunits are responsible for developing, recommending, or selecting a proposal for action. Thus, aside from the many individuals who participate in this process, there is usually an individual or group of individuals who is formally accountable for the decision. Such an individual or group is referred to as a *decision unit.*

What is the effect of computer-aided communication and decision-support technologies on the size and heterogeneity of decision units? As a point of departure for answering this question, recall that research on face-to-face groups suggests smaller and more homogeneous groups provide more satisfying experiences for their members and accomplish decision-related tasks more quickly, if they have the necessary expertise. Thus, forces exist that tend to limit the size of decision units. But units do grow large and diverse, because of the need for a variety of experts or of representatives of various constituencies.

Note, however, that the previous discussion implies that computer-aided communication technology can help decision units become relatively smaller and more homogeneous by obtaining information more easily from those outside the decision unit. Both experts and constituency representatives often make their knowledge and concerns available through electronic mail, teleconferencing, or videoconferencing, either in real time during the meeting or off-line in anticipation of the meeting. Cost considerations cause organizations to seek such efficiencies in their use of "human information repositories." In some cases, an expert's expertise can be replaced by an expert system, and information keepers can be replaced by management information systems. To the extent that a decision unit can properly use such computer-resident information for resolving uncertainties, the expert or information keeper need not be a member of the decision unit and consequently the decision unit's size and heterogeneity decreases.

Certain factors and circumstances limit, or in the future will limit, these uses of information technology for reducing the size of decision units and for increasing their efficiency. For example, as organizational environments and decision situations become more complex, collective interpretation of new information will more and more be a significant component of the decision-making process. As another example, management policies that emphasize employee participation or widespread representation of key groups or constituencies work against efforts to decrease the size or heterogeneity of decision units. Nevertheless, the increasing efficacy of these technologies and the growing need for efficient use of human resources increasingly generate situations where *use of information technology leads to decreases in the number and variety of members comprising the formal decision unit.*

Thus, although the *total number* and *variety* of participants serving as information sources tend to increase with use of information technology, the *number and variety of members within a formal decision unit* tend to decrease with use of information technologies. While all decision units are subject to these forces, it important to note that the way traditional face-to-face groups deal with these issues frequently differs from groups whose members are not able to meet because they are distributed in time or location. The desirable composition of a decision unit might change in ways that are not fully understood when the primary means of interaction changes. Further, the existence of information technologies raises questions about the range of situations where face-to-face decision units are most useful.

What is the effect of computer-aided communication technologies on the use of traditional face-to-face decision units? The use of virtual or distributed work teams in and between organizations is extensive. These often temporary, boundary-spanning groups can combine global scale and scope with local presence and familiarity. When it is important to make decisions quickly and when it would cause hardship for the decision unit members to leave their multilocated "home units" for face-to-face meetings, enabling people with the requisite skills and information to communicate from their distributed normal work sites is highly desirable. As a result, the use of computer-aided communication technology to support distributed decision units within and between organizations has become commonplace. Thus, *use of computer-aided communication technologies increases the variety of decision unit processes used in and by organizations and leads to proportionately less reliance on traditional face-to-face decision units.*

C. Changed Nature of Meetings

The making of important organizational decisions can take many weeks or even months. Meetings are used to speed up decision processes by creating situations where the rate of decision-related information exchange among key participants is generally higher than that which normally occurs outside of meetings. Meetings, whether ad hoc processes or cojoined with more permanent structures (e.g., standing committees), are an important component of organizational decision processes and occupy a good deal of the time of managers and other professionals. However, use of information technology raises at least two questions: What constitutes a meeting? How do meetings differ when participants are able to use computer-aided communication and decision-support technologies?

Historically, meetings were thought of as events where information is exchanged within a short period of time, generally not more than a couple of hours. Thus, a *meeting* was the real-time (synchronous) interaction of the participants. However, using computer-aided communication technologies, information is often exchanged intermittently to participants distributed in time as well as—possibly—in location. When collocation of meeting participants in time and place is difficult, use of computer-aided technologies can still enable the relevant information to be shared or discussed, accomplishing many of the same functions as a face-to-face meeting. The desirability of this leads to the situation that the *use of computer-aided communication technologies increases the proportions of meetings that are distributed in time and the proportion where meeting participants are distributed in physical proximity.*

During traditional face-to-face meetings, discussion is often halted and another meeting scheduled because some information needed to continue is missing. On-line management information systems or other query-answering technologies, including expert systems, reduce some of these occurrences. Also, electronic mail and other computer-aided communication media are sometimes used to access soft information that can be obtained only by asking people. Further, decision-support systems are used within meetings to conduct analyses that provide new information with which to resolve disagreements about the significance of effects of different assumptions, and thereby allow progress to continue rather than forcing adjournment until subsequent staff work can clarify the effects and another meeting can be scheduled. Altogether, these uses lead to the conclusion that *use of information technology can improve the effectiveness of meetings, thereby increasing the quality and timeliness of decision unit choices.*

Reflection indicates that each of the technologies just mentioned as facilitating the effectiveness and efficiency of meetings sometimes leads to the cancellation of meetings. That is, with the added communication and computing capabilities, organizational members occasionally connect and accomplish the purpose of the meeting before the meeting takes place. Finally, because group-decision-support systems enhance information retrieval and analysis, they contribute to the effectiveness of each meeting in a series of meetings and, thus, enable decision groups to complete their tasks with fewer meetings. In contrast, if managers and others involved in making organiza-

tional decisions believe that use of the technologies results in more effective meetings, the availability of the technologies sometimes encourages them to have more decision-related meetings than they would otherwise. In addition, electronic mail, decision-support systems, and other information-sharing and -generating technologies facilitate additional communication outside of meetings. This preempts the need for the larger, formal meetings, but the result is often more communication in total. Seldom is this additional communication wasteful; generally it contributes to the quality of the decision and/or the future relationships among organizational members. Altogether, these ideas indicate that *use of information technology decreases the frequency of formal meetings.*

Computer-mediated meetings usually provide a much different experience for the participants than do face-to-face meetings. There are obvious changes in process that occur with mediated communication, particularly if a group-decision-support system is employed to facilitate or guide the interactions of participants (e.g., allowing participants to vote anonymously). The content of meetings may also vary with the communication or decision-aiding technologies employed, as managers choose the most appropriate technology to pursue desired outcomes. For example, simple information sharing may be preferred through electronic mail, while face-to-face meetings can focus on higher level examination or integration of ideas. Thus, traditional face-to-face meetings serve a different instrumental purpose. And, even if more is accomplished in other venues, face-to-face meetings may still persist for their symbolic benefits, for example, as participants show commitment to their task and to each other by traveling great distances to be together at the same time and place.

IV. EFFECTS AT THE ORGANIZATIONAL LEVEL

This section deals with the effects of information technologies at the organizational level.

A. Centralization of Decision Making

Top managers are understandably motivated to make decisions that address local, lower level situations. By virtue of having achieved their positions, top managers tend to have confidence in their decision-making ability. This may be a greater degree of confidence than they have in the ability of their subordinates. In addition, in many cases top managers know more about the possible spillover effects of a decision on adjacent units or on the organization as a whole. So, it is not unreasonable that top managers are sometimes motivated to retain decision-making authority, or to hierarchically centralize decision making. Unflattering motivations to retain decision authority might include the desire to appear in control, to exhibit power, or to achieve recognition. Whatever the motivation, by enabling top managers to obtain undistorted and timely local information, information technologies can increase the frequency with which upper level managers make decisions about lower level situations that they, otherwise, might be unwilling to make. The opportunity to more easily obtain clarification about local situations with computer-aided communication technologies amplifies this tendency. Thus, information technology, on occasion, enables decisions to be made at hierarchically higher organizational levels than if these systems were not available.

Conversely, these same communication technologies enable lower and middle-level managers to stay better informed about the organization's overall situation and about the nature of the organization's current problems, policies, and priorities and, consequently, permit decisions made by these managers to be more globally optimal, more satisfying of the organization's collective criteria. Further, computer-aided communication technologies also allow lower level units to obtain information about organizational priorities or possible spillover effects of certain choices in a more timely manner. Thus, on some occasions, computer-aided communication technologies cause decisions to be made at organizational levels lower than if such technologies were not available. Motivations that lead top managers to permit this practice include the desire to decrease the time for organizational units to respond to problems or the desire to provide autonomy for subordinates or to develop subordinates' decision-making ability.

Therefore, is the net effect of the use of computer-mediated communication and decision-support technologies to increase centralization or to decrease it? Together, the arguments in the previous two paragraphs suggest that, when used to provide more organizational levels with information that was formerly known to only one or a few levels, *use of information technologies enables organizations to make decisions of a particular type across a greater range of hierarchical levels* without suffering as much of a loss in quality or timeliness, as would be the case if the technologies were not so used.

What would interfere with the consequent tendency—in a given organization—for decisions of any

given type to be made at different levels on different occasions and, in sharp contrast, increase the tendency for decisions of all types to be made either centrally or locally? Either of two conditions lead to a preponderance of all decisions being made at either of these extremes, i.e., lead to either greater centralization or to greater decentralization. One is a strong ideological commitment to centralization, or to decentralization, as well-designed decision-aiding information technology greatly reduces the cost in quality or timeliness of making decisions at either extreme hierarchical level. The other condition that causes the preponderance of decisions (of any given type) to be made invariably at a given level is the gradual evolution of organizational routines that caused such decisions to be made more easily or efficiently at that level.

More generally, *when a strong ideological commitment exists at the top management level about the level at which such decisions should be made, or when the organization's environment permits the rather uninterrupted evolution of decision routines, then the availability of computer-aided communication and decision-support technologies leads to a narrow range of organizational levels where a particular type of decision will be made.* Not infrequently, this narrow range will be just one level—the top level, the first-line manager level, or the non-management level where the decision situation is located. *When neither of these conditions is in effect, the availability of computer-aided communication and decision-support technologies leads to a broad range of organizational levels at which a particular type of decision will be made.* Because it is easier to envision, we here addressed the locus of decision making as if the issue is one of hierarchical level. But, especially in an era of less hierarchical organizations, it is important to recognize that in each of the two conclusions just stated, one could replace "hierarchical level" with "organizational unit" and maintain the validity of the conclusion.

It is important to emphasize that, by increasing the hierarchical range across which a particular type of decision is made without a corresponding loss in decision quality or timeliness, computer-mediated communication and decision-support technologies allow other decision-location considerations to be applied without prohibitive costs. Examples are a diversity of views among the organization's elite about the appropriate locus of decisions or adherence to organizational traditions, norms, or culture in spite of the specifics of the decision situation. Because the relative influence of such considerations varies from organization to organization, *for a given population of organizations, use of computer-aided communication and decision-support technologies leads to greater variation across organizations in the levels at which a particular type of decision is made.*

B. Number of Organizational Levels Involved in Authorization

Consider the common situation where at least some conclusions of lower level units about what actions should be taken must be authorized by higher level units before being acted upon, and where these conclusions are forwarded upward as proposals. Because each hierarchical level requires time to process a proposal (e.g., waiting for other proposals or for related information to arrive), the more levels involved, the longer the process takes. Each additional time increment in the duration of the approval process adversely affects both the timeliness of the authorized action and the enthusiasm with which those who propose the action carry out the action once it is authorized.

Why, then, do organizations commonly involve several levels in authorizations? The answer is that each level in the hierarchy generally has knowledge or decision-specific information that qualifies it to apply criteria or decision rules that less well-informed lower-level units cannot apply. For example, each higher level in an organization tends to know more about organization-wide issues, needs, and resources, and more about the nature of currently competing demands for resources, than does its subordinate units. The greater the amount of such information that is relevant, the more hierarchical levels are involved in the authorization process.

What is the effect of communication and decision-support technologies on the number of hierarchical levels involved in authorizing a particular decision? Information technologies cause a decrease in the number of hierarchical levels involved in authorizing a proposal because technologies such as management information systems, expert systems, electronic mail, and electronic bulletin boards make information more quickly and widely available. In some cases, organizational levels obtain information that was previously unavailable and, thus, they can apply criteria or decision rules that they previously were not qualified to apply. Consequently, because the technologies facilitate the vertical distribution of information and knowledge (understanding about how to use information), there is more commonality (less extreme differentiation) of information and knowledge across organizational levels. Therefore, except when information technologies are allowed to create a problem of information over-

load, a given organizational level is more likely to be qualified to apply more criteria and decision rules than it would be without the technologies. Assuming that use of the technologies is not allowed to lead to information overload and does not somehow increase the number of decision rules used, it follows that *use of computer-aided communication or decision-support technologies reduces the number of organizational levels involved in authorizing proposed organizational actions.*

C. Number of Nodes in the Information-Processing Network

Decision-making individuals and units obtain much of the information used to identify and deal with decision situations through an information-processing network. The outer boundaries of the network are the sensor units (such as market analysts, quality control personnel, and accountants) that identify relevant information from either inside or outside the organization. These units serve as information sources, and in many situations they pass on their observations in the form of messages to intermediate units closer to the ultimate user, the decision-making unit. Quite often these intermediate units are at hierarchical levels between the sensor unit and the decision-making unit.

The recipients of the sensor unit's message process the message and pass it on to a unit that is closer still to the decision-making unit. The information processing performed by such intermediate units ranges from straightforward relaying to elaborate interpreting. For a variety of reasons, the number of such units—the number of nodes on the network path connecting the sensor unit to the decision unit—may be greater than warranted for a specific decision situation.

Besides the unnecessary costs implied in having people simply relay information, each information-handling unit in the network path tends to contribute distortions and delays. For these reasons, top managers sometimes reduce the role of such units by using computer-assisted technologies as alternative means for obtaining the information. This reduces the workload that justifies the existence of these intermediate units and levels and, consequently, reduces their number. Extending this reasoning, and seeking to reduce costs, computers are sometimes used to merge, summarize, filter, and even interpret information, thus eliminating clerical workers, managers, and the organizational units of which they are a part. These observations suggest that *use of information technology leads to fewer intermediate human nodes within the organizational information-processing network.* If the network processes information across hierarchical levels, then *use of information technology reduces the number of organizational levels involved in processing messages.*

However, this effect is mitigated in many situations. As noted at the subunit level, computer-mediated communication usually increases the number of units providing information to a decision unit on any given decision. This, combined with the elimination of intermediate nodes in the network, can result in information overload on the decision unit. When the interpretation and other processing functions performed by intermediate information-processing nodes cannot be as efficiently or effectively performed with technology or changed practices, there are few or no reductions in the number of such nodes.

D. Flexibility and Permeability of Organizational Boundaries

A core question in organization theory is how to identify or define an organization. Answers often include references to the boundaries of the organization as a means to delineate what is and what is not a part of the organization. Unfortunately, this approach is muddied by changes in the way organizations design their processes and interface with other organizations, such as suppliers or business partners, changes that are often enabled by information technology. Thus, a fundamental question is: How does information technology change organizational boundaries?

An initial consideration is what organizations choose to accomplish internally versus what they acquire from other organizations to accomplish their own purposes; decisions of this nature impact where the "line" is drawn between a focal organization and others. As in the previous centralization-decentralization discussion within organizations, there is a question about whether or not information technology always leads more activities to occur between organizations (so-called market relationships) or always leads more activities to occur within organizations (often referred to as hierarchy). Arguments for movement toward hierarchy suggest that information technology reduces transaction costs for managing internally, creating greater opportunity to manage and control organizational activities internally. Arguments for movement toward markets suggest that reduced transaction costs for communicating and contracting externally create opportunities for organizations to focus on their internal strengths and purchase noncore products and services from organizations who can provide them more cheaply and effectively. While recognizing that the impact is probably no more monolithic

than it is in the case of centralization, it is clear that *availability of information technology enables more choices in organizational design options, thereby creating more flexibility in the location of organizational boundaries.*

Despite these increased options, few organizations are capable of gathering all of the resources and competencies needed to create a product or service that serves a particular need of end consumers. In most cases, the task of transforming inputs (e.g., automobile parts) into the desired output (e.g., an automobile) is broken into pieces or tasks that are produced or accomplished by different organizations. Interdependence among these organizations is a natural consequence of decomposing the overall task. Historically, individual organizations (or people) have acted independently to buffer themselves from their environments, including suppliers, distributors, and other business partners. In some cases, this has led to vertical integration, so that the focal organization has more direct oversight over the entire task. In other situations, organizations build inventories or flexible capacity to buffer themselves from fluctuations in the supply or demand of other organizations or subunits in a supply chain. Such responses to the interdependencies can lead to excess costs and inefficiencies for each organization and the organizations as a group.

The introduction of computer-mediated communication has made other options for handling the interdependencies more feasible. Faster communication between organizations (e.g., through electronic mail or electronic data interchange) permits faster reactions within and across organizations. As managers pursue lower costs, they use these opportunities to decrease unnecessary buffers like excess inventories. As managers work together across organizations, they often reorganize and synchronize their processes to further pursue efficiencies. Thus, *use of computer-aided communication technologies reduces the costs of interdependencies and, consequently, enables tighter coupling of organizations and of subunits.* The result is that more of the operational technologies or activities cut across organization boundaries than was the case when computer-aided communication technologies were not available. As such, an organization, defined as an interdependent set of actors working toward a common cause, is a much broader enterprise than an organization defined as a legal entity, blurring the line separating organizations.

A final concern is how an organization interfaces with its environment, specifically the permeability of its organizational boundaries. Information technology creates more points of contact and information exchange among people, regardless of their organizational affiliation. Rapid exchange of data among business partners (e.g., through electronic data interchange) allows organizations to act together more seamlessly. Other computer-aided communication technologies such as electronic mail permit more interaction among members of different organizations, in some cases bypassing formal liaison roles. Technologies such as the World Wide Web also change interfaces with end consumers. These technologies offer more channels and more direct channels to access the end consumer. For example, some manufacturers work outside of their traditional distribution channels and gain more information by interacting directly with their customers. Taken together, changes in organizational design and means of communication suggest that the *availability of computer-aided communication technology leads organizations to exchange more information across their boundaries.* Important practical implications of this fact have been discussed by Quinn, Anderson, and Finkelstein and by Pickering and King.

V. EFFECTS ON SIZE, DISPERSION, AND COMPLEXITY OF ORGANIZATIONS AND ORGANIZATIONAL NETWORKS

An organizational network is a group of organizations that communicate with one another for collaborative purposes. The actions of the network members can range from limited information sharing to more elaborate forms of coordination or collective action. Without information technologies, communication inefficiencies cause networks to be somewhat geographically and, hence, numerically limited. In contrast, the advantageous properties of computer-aided communication technologies tend to enable organizational networks to be more dispersed and to contain more members. In competitive environments where there are benefits to be gained from information sharing, collaboration, and perhaps economies of scale, managers use these technologies to enlarge the size of their networks. This results in both more members of the network and more dispersion in the network. Thus, the *use of computer-aided communication technologies enable the development of larger, more dispersed networks of organizations.* This important fact has been elaborated on by Dutton and by Monge and Fulk.

The previous discussion makes no explicit assumptions about the type or nature of organizations in the network, only that there are more of them and more in different locations. However, it is also helpful to examine the effects of advanced information technologies on the diversity of organizations in a network. Di-

verse organizations typically require more elaborate, more rapid, and more frequent communication to achieve the purposes of their relationship. Computer-aided communication technologies facilitate these relationships by providing additional communication channels that enable more rapid exchange of information among the participants. This, in turn, makes it more feasible for diverse organizations to successfully exchange information and collaborate. In competitive environments, where having diverse information sources or complementary partner competencies is important or where economies of scope are beneficial, managers build networks composed of more diverse organizations. In summary, the *use of computer-aided communication technologies enables the development of networks of more diverse organizations.*

To further elaborate on the effects of information technology on networks of organizations, two very different configurations of collaborating organizations—infinitely flat organizations and virtual organizations—are now introduced for expositional purposes.

An *infinitely flat organization* is a network of organizations consisting of two layers, a headquarters and a tier of nearly identical operating units. The headquarters unit, as the central member of the network, develops and distributes operations-related knowledge and directives to the operating units, monitors their individual activities and performance, and takes corrective action when appropriate. The dispersed operating units (e.g., franchisees) are the nodes of the network. They leverage and exploit the resources and competencies of the headquarters of the network (e.g., fast-food chain headquarters). In addition, their idiosyncratic circumstances and varied performance cause these operating units to serve as "natural experiments," experiments from which the headquarters unit can learn and thereby acquire even more operations-related knowledge. The headquarters unit makes large investments in knowledge assets, procedures, or systems that can be utilized at little or no additional cost as operating units are added to the network, i.e., the value of the infinitely flat organization grows rather linearly with the size of the network. While there may be little communication among the operating units in the network, there is a great deal of communication between the headquarters unit and the operating units. Knowledge assets and directives are communicated downward and performance results and requests for resources and exemptions from directives are communicated upward. Clearly, the positive properties of computer-aided communication technologies increase the viability of infinitely flat organizations.

Very different from an infinitely flat organization, with its hierarchy and intended permanence, is the *virtual organization.* A virtual organization is an assemblage of independent organizations coordinating their efforts toward achieving some purpose. Examples include construction consortia, military alliances, and organizations cooperating in a disaster relief effort. Communication in the virtual organization frequently occurs from participating organization to participating organization (node to node); there is often no headquarters or other coordinating unit. Individual members possess and contribute specialized knowledge and other resources. Members must be able to access information about the location and nature of such resources in the network and must be able to coordinate their joint efforts with other members. It is clear that the advantageous properties of computer-aided communication technologies cause their use to increase the viability of virtual organizations.

Infinitely flat organizations and virtual organizations share some common features. While both types existed in some form prior to the introduction of current information technologies, both are more feasible and prevalent when using computer-aided communication technologies. Also, both are organized to exploit the resources of their networks, albeit in different ways. In the infinitely flat organization, network members leverage the same knowledge extensively through its repeated use. Information technologies are instrumental in developing, sharing, and monitoring the use of this knowledge. In the virtual organization, members combine their unique knowledge or resources to create or apply the collective capability necessary to achieve their purpose.

Examined together, these very different organizational configurations suggest a variety of ways that organizations can use computer-aided communication technologies to organize their actions and better utilize their resources. *Use of computer-aided communication technologies permits a larger variety of multi-organization configurations.*

VI. EFFECTS ON ORGANIZATIONAL INTELLIGENCE AND DECISION MAKING

This section summarizes the effects of information technologies on organizational intelligence and decision making.

A. Organizational Intelligence

Organizational intelligence is the output or product of an organization's efforts to acquire, process, and

interpret information from outside of the organization. It is an input to the organization's decision makers. Because interpretation of external information involves assessing the implications of the information for the organization, organizational intelligence often draws on information in the organization's memory banks, e.g., information in the minds of its members or in the documents of its files.

To some degree, all organizations scan their external and internal environments for information about problems or opportunities. In many instances, an alerting message is delayed or distorted as it moves through the sequential nodes in the communication network. In other instances, incumbents of adjacent nodes in the communication network have difficulty connecting across time, as in "telephone tag." As a result of these problems, managers sometimes do not learn about problems or opportunities in time to act with maximum effectiveness. Use of computer-aided communication technologies facilitates the "rifle-shooting" of messages and ultimately leads to fewer, sometimes zero, intermediary nodes in the network associated with any particular message making its way from a source to a recipient. This idea, in combination with the fact that the probability and duration of message delay and the probability and extent of message distortion are both positively related to the number of sequential links in communication chains, suggests that use of computer-aided communication technologies facilitates rapid and accurate identification of problems and opportunities—whether inside or outside the organization. (At the same time, since an important role of many network nodes is to screen, package, and interpret messages, the use of the technologies and the consequent elimination of nodes can result in an overload of irrelevant, poorly packaged, or uninterpretable messages. Fortunately, social norms and management practices tend to develop to reduce the problem to a level below what might be imagined.) Use of these technologies aids not only in the identification of problems and opportunities, but also facilitates a wide variety of more focused probes and data acquisitions for the purpose of analysis and interpretation. Thus, the *use of computer-aided communication technologies leads to more rapid and more accurate identification and interpretation of problems and opportunities.*

Technologically advanced systems for the acquisition of external information include, of course, not only computer-aided communication technologies but also environmental and performance-monitoring technologies such as point-of-sale inventory monitoring systems and media monitoring technologies. These systems and the development of computer-enhanced organizational memory banks together enable organizations to increase the range of information sources that the producers and users of organizational intelligence can draw upon. Thus, in summary, *use of computer-aided information storage and acquisition technologies leads to organizational intelligence that is more accurate, comprehensive, timely, and available.*

B. Decision Making and Decision Authorization

The quality of an organizational decision is largely a function of the quality of the organizational intelligence and the quality of the decision-making processes. Once a decision situation has been identified, several activities are undertaken that are more effective when undertaken using information technology. As seen earlier, by facilitating the sharing of information, computer-aided communication technologies increase the information and expertise available to decision makers. For example, electronic mail and video- or teleconferencing help decision makers obtain clarification and consensus without the delays imposed by the temporary nonavailability, in terms of physical presence, of key participants. And, of course, management information systems and electronic mail enable decision makers to more immediately obtain information they might need when deciding what to do about problems and opportunities, while decision-support systems enable decision makers to analyze this information quickly (at least for some types of problems). Thus overall, *use of computer-aided communication and decision-support technologies leads to decisions that are more informed and more timely.* Further, because reducing the number of levels involved in authorizing an action reduces the number of times a proposal must be handled (activities of a logistical, rather than a judgmental nature), *use of computer-mediated communication and decision-support technologies reduces the time required to authorize proposed organizational actions.*

VII. SUMMARY AND OPPORTUNITIES FOR FUTURE RESEARCH

Information technologies in use today have properties different from more traditional communication and decision-aiding technologies. The availability of computer-aided communication technologies extends the range of communication technologies from which organizational designers can choose. When these technologies are chosen wisely, their use leads to more available, accessible, and more quickly retrieved information, including external information, internal

information, and previously encountered information stored in organizational memory banks. Increased information accessibility permits more options in organizational design. Increased information accessibility and those changes in organizational design that increase the speed and effectiveness with which information can be converted into intelligence or intelligence into decisions, lead to organizational intelligence that is more accurate, comprehensive, timely, and available. This higher quality intelligence combined with computer-aided decision-support systems and with communication technologies that enhance decision unit functioning, enable organizations to make higher quality and more timely decisions, decisions that lead to improved organizational performance.

This article sets forth the effects that computer-aided communication and decision-support technologies have on organizational design, intelligence, and decision making. Some caveats and opportunities for future research should be noted, however.

The original ideas espoused by Huber in 1990 were based on observations of organizational practices and on extrapolations from research both in other fields and on older communication and decision-support technologies. More recent research and observations were drawn upon here to expand the original work. Additional research is required to clarify and validate some of the relationships described. Managerial intent, organizational characteristics, and environmental factors especially need to be considered when testing the relationships.

Of course, there are exceptions to the relationships explicated in the summary statements made in this article. The intention is to state that across a large number of cases, *ceteris paribus,* there is a tendency for the stated relationship to evolve. Further research will, eventually, identify any systematic contingencies or exceptions to these relationships. For example, regarding increased participation in decision making, research is needed to determine the circumstances when (1) the potential increase in participation actually occurs; (2) the decision process becomes more effective, and (3) the increase in participation leads to higher quality decisions or better acceptance of decisions.

As another example, authorization as a particular step in the decision-making process has received little attention from organizational scientists, and the time required for organizations to authorize action also has received little attention. These topics are worthy candidates for study in general, and the potential effects of information technology seem to be especially in need of examination, given their probable importance and the total absence of systematic research on their effect on decision authorization. Unfortunately, few studies have systematically examined the circumstances where a causal relationship might exist between the use of information technology and a reduction in the number of organizational levels involved in decision authorization. Research may determine that while fewer levels are needed for information purposes, changes in the costs for involving multiple layers do not lead to changes in this area.

Finally, different forms of information technology were sometimes discussed by name (e.g., electronic mail), yet summary conclusions were made in more general terms. This latter fact should not obscure the need to specify more precisely the particular technology of interest when developing hypotheses to be tested. As more is learned about the effects of computer-aided communication and decision-support technologies, subtle differences in the properties or attributes of the technologies may become more important. Even in areas where these differences are proven benign, it behooves researchers and managers to be clear and precise about what it is that they are discussing.

ACKNOWLEDGMENTS

This paper incorporates, updates, and extends earlier work by Huber (1990). The authors appreciate the helpful comments of Pamsy P. Hui and the reviewers on an earlier draft.

SEE ALSO THE FOLLOWING ARTICLES

Computer-Supported Cooperative Work • Decision-Making Approaches • Decision Support Systems • Human Side of Information, Managing the Systems • People, Information Systems Impact on • Psychology • Resistance to Change, Managing • Sociology • Virtual Organization

BIBLIOGRAPHY

Daft, R. L., and Lengel, R. H. (1986). Organizational information requirements, media richness and structural design. *Management Science,* 32, 554–571.

DeSanctis, G., and Poole, M. S. (1994). Capturing the complexity in advanced technology use: Adaptive structuration theory. *Organization Science,* 5, 2, 121–147.

DeSanctis, G., and Poole, M. S. (1997). Transitions in teamwork in new organizational forms. *Advances in Group Processes,* 14, 157–176.

Dutton, W. H. (1999). The virtual organization: Tele-access in business and industry. *Shaping organization form: Communication, connection, and community* (G. De Sanctis and J. Fulk, eds.). Newbury Park, CA: Sage.

Fulk, J., and DeSanctis, G. (1995). Electronic communication and changing organizational forms. *Organization Science,* 6, 4, 337–349.

George, J. F., and King, J. L. (1991). Examining the computing and centralization debate. *Communications of the ACM,* 34, 7, 63–72.

Huber, G. P. (1990). A theory of the effects of advanced information technologies on organizational design, intelligence, and decision making. *Academy of Management Review,* 15, 1, 47–71.

Huber, G. P. (1991). Organizational learning: The contributing processes and the literatures. *Organization Science,* 2, 1, 88–115.

Huber, G. P., and McDaniel, R. R. (1986). The decision-making paradigm of organizational design. *Management Science,* 32, 5, 572–589.

McPhee, R. D., and Poole, M. S. (2001). Organizational structures and configurations. *The new handbook of organizational communication* (F. M. Jablin and L. L. Putnam, eds.). Thousand Oaks, CA: Sage.

Monge, P., and Fulk, J. (1999). Communication technology for global network organizations. *Shaping organization form: Communication, connection, and community* (G. DeSanctis and J. Fulk, eds.). Newbury Park, CA: Sage.

Orlikowski, W. J. (1992). The duality of technology: Rethinking the concept of technology in organization. *Organization Science,* 3, 3, 398–427.

Pickering, J. M., & King, J. L. (1999). Hardwiring weak ties: Inter-organizational computer-mediated communication, occupational communities, and organizational change. *Shaping organization form: Communication, connection, and community* (G. DeSanctis and J. Fulk, eds.). Newbury Park, CA: Sage.

Quinn, J. B., Anderson, P., and Finkelstein, S. (1996). Leveraging intellect. *Academy of Management Executive,* 10, 3, 7–27.

Sutcliffe, K. M. (2001) Organizational environments and organizational information processing. *The new handbook of organizational communication* (F. M. Jablin and L. L. Putnam, eds.). Thousand Oaks, CA: Sage.

Outsourcing

Eduardo Gelbstein
International Computing Centre, United Nations

GLOSSARY

enterprise resource planning systems (ERP) Integrated applications covering finance, human resources, procurement, logistics, and other structured functions. Available as commercially available packages requiring customization to meet an individual client's needs.

offshore Another country, usually distant and with lower unit costs.

outsourcing The use of outside resources to perform specific functions. The buyer of services turns over the control of the processes to the supplier. The supplier is responsible for deciding how to accomplish the contractually defined results.

service level agreement (SLA) The contractual formulation of service levels. Through these SLAs agreement the buyer may have the option of claiming penalties if performance does not meet the specified targets or have the legal basis for the termination of the contract.

service levels The formal definition of the metrics and targets through which the supplier's performance will be monitored throughout the outsourcing contract.

I. INTRODUCTION

Outsourcing is the mechanism through which technology services are bought from an external specialist party on a contractual basis instead of performing those tasks with internal resources. The factors driving the decision whether or not to outsource are discussed later in this article.

Information technology (IT) outsourcing started over 30 years ago and was referred to as facilities management (FM). Under FM arrangements, data processing hardware, software, and people were transferred to a service provider as a package that was easily quantifiable and priced, and the service provider (outsourcer) delivered economies of scale.

By the end of the year 2000, global IT outsourcing had become a robust industry reaching an annual volume of business in the order of 100 billion dollars. The choice of services has expanded to include, in addition to FM and infrastructure operations,

- Network management and operations
- Maintenance of legacy applications
- Development of client server applications
- Customisation and implementation of ERP systems
- Development of e-commerce and e-business applications
- Application service providers
- Web and e-commerce hosting

In addition, there is a marked growth in business process outsourcing, such as order fulfillment and warehousing to support e-businesses. Business process outsourcing falls outside the scope of this article.

From the above list of services, it is possible to distinguish two distinct types of outsourcing, the

outsourcing of process work and the outsourcing of project work. These are discussed in the section that follows.

II. SCOPE OF OUTSOURCING SERVICES

A. Process Work

This includes all those activities that are, or can be, standardized, systematized, documented in detail, and automated. Typical examples of these are data center operations and web hosting.

These activities are supported by metrics, such as, availability, number of unscheduled service interruptions, which are relatively easy to monitor and analyze. After the transition to a vendor, these activities become "business as usual."

This section will briefly examine the main features of the following services:

- Day-to-day data center operations (mainframe and other large systems)
- Day-to-day operations of distributed systems, including desktops, local area networks, end-user support, etc.
- Supply and day-to-day operations of wide area networks including network management
- Applications service providers (ASP)
- Services for e-commerce and e-business

The paragraphs below describe the main features of these outsourcing services. The sections that follow discuss in more detail the main issues relating to these services when they are outsourced.

1. Data Center Operations

This is the outsourcing business with the longest track record. Here, the hardware, software, staff, and all other components of a data center operation are transferred to a specialist third party. In most cases, the services will be provided from the vendor's premises.

The vendor undertakes to deliver a range of services to a level of quality specified in the contractual agreement.

2. Distributed Systems and Desktop Operations and Support

The main differences between this scenario and that of data center operations are that the range of technology and software, as well as the number of items to manage is usually greater, that the vendor needs to have a physical presence at the client's premises and may also need to have a considerable, if not total, say in technology choices, the implementation of moves, additions, and changes (MAC), system and network management tools, etc.

3. Wide Area Networks

Changes in the telecommunication industry such as deregulation, mergers and acquisitions and joint ventures have led to lower charges and a growth in the availability of global connectivity.

This has created a large market for the provision of managed wide area network (WAN) services as well as for the provision of end-to-end connectivity and support services.

4. Application Service Providers (ASP)

These represent a relatively new business, initially targeted at small and medium-size enterprises and start-up ventures, but increasingly of interest to large organizations. An ASP deploys, hosts, and provides all the activities and expertise required by a set of applications from a facility other than the client's premises, mainly through the Internet.

Application service providers are distinctly different from the provision of hosting services, where the client owns the application and also from business process outsourcing.

In dealing with an ASP, the client buys access to a managed application. The ASP owns or otherwise acquires the necessary software licenses as well as the entire infrastructure to host, operate, and support these applications.

The main distinguishing characteristic of the current ASP market is that the offerings are standardized or have, at best, minimal customization. This presents a barrier to large and established companies and organizations that have developed their own procedures and practices, but not to small and medium companies, which find it easier to accept no customization. The same applies to start-up ventures.

Industry observers expect that the ASP industry will undergo a period of instability where small players with poor business models, poor execution, or an inadequate client base will fail to survive. It can also be expected that successful ASPs will, as in the case of the other outsourcing models, undergo market consolidation and often a wider range of services such as hosting and security management.

5. E-Commerce and Other Internet Age Outsourcing Models and Practices

The dynamics of e-commerce have strengthened the importance of a number of issues concerning outsourcing.

Many new vendors—There are many new entrants to the field previously dominated by established companies in both process and project outsourcing. Many of these are themselves major operators, particularly in telecommunications, and are making major investments to position themselves in new markets such as that of web hosting and supporting e-commerce.

Information security—The need to guarantee to clients and consumers that data such as proprietary processes, payroll information, customer details, and credit card numbers will remain confidential and misused or modified without due authorization. An additional security concern in this area is that of being targeted by a denial of service attack.

Service level delivery—In an environment characterized by unpredictable workloads, fluctuating capacity requirements and variable response times are necessary.

Business process outsourcing—Although this is outside the scope of this article, this represents a major component of e-business and includes activities such as warehousing, order fulfillment, and payment collection. The decision-making process, contractual issues, and success factors for this kind of outsourcing are essentially the same as for the other cases, but likely to be of a considerably more critical nature and subject to higher risks.

B. Project Work

These activities are unique to every project, nonstandard, and frequently not very structured. They also require considerable creative input and their metrics are more complex to manage.

The maintenance of legacy applications and the customization of enterprise resource planning (ERP) packages both fall in this category. After completion, the outcome of these projects leads to a change in the status quo of the client organization.

The outsourcing of software work is primarily driven by a continuing shortage of skilled and experienced personnel. It started with the provision of qualified personnel to complement the staff assigned to such projects and this practice is often referred to as "body-shopping."

While this practice continues, there are many companies across the world that have evolved from this approach to build large software factories employing 500 or more employees, using Rapid Application Development (RAD) tools, certified to ISO 9000 or equivalent total quality management (TQM) qualification. There are many countries where salaries are considerably lower than in North America and Europe and this competitive edge has strengthened their role in this market.

This section will briefly examine the main features of the following services:

- Outsourcing the maintenance of legacy applications
- Outsourcing development and maintenance of client/server applications

The outsourcing of work related to ERP systems customization, integration, and operations is treated separately as its complexities make it a hybrid of both process and project outsourcing.

Section III discusses the main issues to be considered when outsourcing project work.

1. Legacy Applications Maintenance

A substantial amount of this work is carried out internationally and it covers both mainframe applications written in COBOL and similar languages and database conversion from, for example, flat files to relational.

The driver for outsourcing the maintenance of legacy application software is the need to free up staff for new projects. It has to be recognized that most of such software is poorly structured and that the knowledge relating to this software is not fully documented. This will require numerous and frequent exchanges between the client and the vendor's staff. This could become a step toward a replacement of these systems which itself becomes an outsourcing project.

2. Client/Server Applications Development

Many countries, such as India, have good skills in environments such as UNIX and this has allowed them to build a strong position in this market.

For most large new systems, it is likely that the requirements defined at the outset of the project are neither complete nor final. This will require intensive interaction between client and vendor, possibly necessitating a physical presence at the client's premises and an infrastructure that enables the simple validation of prototypes.

3. An Outsourcing Hybrid: ERP System Customization, Integration, and Operations

These systems have been adopted in considerable numbers during the last few years, usually based on packages from a relatively small number of companies such as SAP, Oracle, and PeopleSoft.

These are complex and highly integrated distributed systems, which are frequently implemented by third parties specializing in this field. While there is considerable demand for the outsourcing of the operations and support of these systems, as shown below, this cannot be done without a project culture to manage the numerous and frequent changes required by these applications.

In early 2001 the marketplace is preparing to meet these requirements.

The outsourcing of ERP operations and support differs considerably from that of infrastructure or the traditional mainframe facilities management for several reasons:

- The constant and rapid evolution of ERP implementations, where business changes drive the need for new functionality to reflect changes to business rules
- The rapid dynamics of ERP products or packages which result in frequent software bug fixes and new version releases
- ERP operations and support are highly specific to the ERP package used and are very knowledge-intensive
- For large corporations or organizations, large-scale changes, e.g., the implementation of new modules, implementation at additional sites, synchronized version upgrades, etc., may be cross-border projects that can only be handled as major projects coordinated by an ERP management Center
- Performance tuning needs to be carried out virtually all the time both at the technical level and at the level of business process definition; these must be dealt with in a coordinated and process oriented manner
- The provision of disaster-recovery capabilities for ERP implementations is a highly complex matter which must continue to evolve as these systems become increasingly critical

If the decision to outsource ERP operations and support is made prior to the implementation of the system, many additional tasks related to populating the database (data conversion and validation, data transfer, validation of all system documentation, integration testing, etc.) need to be added to the scope of the outsourcing and the relevant contracts.

End-user support covers two separate areas, a first level to evaluate and assign end-user calls and a second level to deal with the support of the ERP application and how it has been customized. It is recommended that both these levels of support be integrated in a single problem management system.

III. MAKING THE DECISION TO OUTSOURCE

The decision to outsource is never simple and the answer is not always obvious.

The consideration of outsourcing is usually driven by such factors as a perception of high cost, poor quality of service, staff shortages/inability to recruit, and retain and/or a policy to concentrate or core activities.

The initiative to consider outsourcing may come from the Chief Executive, the Board of Management or the Chief Information Officer (CIO), depending on circumstances.

Companies and organizations also need to consider alternatives to outsourcing such as:

- Maintaining an in-house IT organization, either centralized or distributed; this organization may be treated as a cost center or as a profit center
- Creating a fully-owned subsidiary company to be their IT service organization with the objective of seeking additional markets for their services, thus increasing its economies of scale

A. Impact of Outsourcing on the Role of the CIO

In every successful outsourcing case study, the transfer of responsibilities to a third party removes many time-consuming activities from the daily agenda of the CIO, in particular, participation in technology-buying decisions. This has released time to deal with strategic information systems issues, policies, and strategies. The later should not be outsourced.

Outsourcing or not, the CIO remains accountable whenever something goes wrong. The CIO must therefore retain appropriate decision-making powers. In a situation where outsourcing does not work well, the CIO needs to find out quickly what the reasons for this are, and act accordingly, before a problem becomes a crisis that can paralyze a company or organization.

The main success factors for outsourcing are discussed in the last Section VII.

B. Benefits, Disbenefits and Risks

The potential benefits of outsourcing include the following.

1. Cost Reductions

In process outsourcing, these are normally delivered through:

- The outsourcer's economies of scale and ability to leverage the procurement of hardware and software
- The transfer of staff to the outsourcer, leading to lower overhead costs

Accounting restructuring, for example, through moving assets off the balance sheet and avoiding capital expenditure, can show changes in return on investment, etc.

In project outsourcing, these cost reductions can arise from:

- the vendor's ability to reuse code
- Software development on an assembly-line basis
- Use of RAD and other specialized tools
- Lower salaries, particularly in the case of international outsourcing

2. Improvements in Quality of Service

These are achieved as a result of having access to the industry's best practices and other intellectual capital as well as through access to skills that are in short supply.

This is particularly true when in-house service provision can neither meet service delivery demands nor be able to adapt to and adopt new technologies.

3. Ability to Obtain Resources and Capacity on Demand

The client transfers the responsibility for providing additional technical capacity as and when required to the vendor, as well as that of finding qualified personnel to cover the unavailability of a person assigned to a process or project, or to meet changing needs.

4. Reduced Managerial Distractions

The outsourcer takes over responsibility for many managerial and administrative tasks that include:

- Staff recruitment, administration, training, and retention
- Technology assessment
- Individual IT procurement activities

Furthermore, the contractual nature of the relationship with the outsourcer simplifies the IT budgeting process.

This article discusses what steps can be taken to enable these benefits to be achieved.

However, every outsourcing project also has potential disbenefits.

5. Higher Costs

Despite the statement above that a vendor can deliver services at lower costs, the client will have to incur costs that did not arise prior to deciding to outsource. Examples of these are

- Legal costs of contracting—This is discussed in more detail in the sections that follow.
- Higher transaction costs—Every change of requirements, scope of services, technology, charges, etc., needs to be treated as an amendment to the contract(s) defining the relationship between client and vendor.
- Cost of monitoring the vendor's performance—This activity is needed to validate that the vendor delivers its services to the agreed quality level, that issues arising in the execution of the contract are logged and raised, etc. In situations where the services are provided by an in-house organization this activity is frequently not performed in a formal manner.

6. Potential Lock-in to a Vendor

Once a client has transferred its information technology assets and intellectual property to a vendor, and has divested itself of staff and their expertise, the decision to outsource is hard to reverse or change.

7. Underperformance by Vendor

A considerable amount of senior management time may be needed to deal with a vendor that consistently underperforms. Situations that would qualify as "underperformance" include:

- Failure to deliver the agreed service levels
- Replacing staff originally assigned to the client by less qualified or less experienced personnel
- Giving a lower priority to the client than that given to a larger client

8. Vendor Abuse

Although clearly not a good business strategy, there have been situations where the interest of the vendor took precedence over the interest of the client. These situations typically include:

- Improper billing, particularly in project work, where more man hours are charged than are actually incurred
- Exploitation of the client's intellectual property in software for the vendor's gain without the consent or participation of the client, as in the case of reusing code in a project for a different client
- Premium level repricing of contract extensions and modifications, thus taking unfair advantage of a lock-in situation

The following risks should also be considered:

- The risk that a small vendor will be taken over by a big vendor or that it will go out of business
- Selecting the "right" vendor at a time when the outsourcing industry has become complex, with a growing number of players, domestic and international
- The complexity of service level management in the Internet Age and the many parameters that an outsourcer cannot control
- Potential loss of data confidentiality and other security issues

C. Human Factors

Whichever outsourcing activity is selected, be it a process or a project, it is people who drive the outsourcing activity from the outset. Outsourcing needs to bring together, by its very nature, more people from more parties than an activity resourced internally.

Success in outsourcing is therefore a joint venture where there are no winners and losers; either everyone succeeds or everyone fails. Should, for example, an outsourcer exploit favorable terms and conditions in a contract this may turn out to be a short-term win, as the contract may not be renewed with a consequential loss of reputation to the outsourcer?

There are many variants of "people issues" in outsourcing. This section illustrates those best known for being potentially able to cause outsourcing to fail.

1. Staff

The following statement is attributed to an anonymous IT person involved in an outsourcing transfer: "I feel like a slave being auctioned off—all that's missing is the chains."

To the staff impacted by a decision to outsource, this is invariably a political and emotional issue, as it is likely to change their employer, their terms and conditions, the location where they work, and many other factors.

As the language associated with process outsourcing often uses expressions such as "noncore activities," "zero-added value," and "lean and mean" when referring to jobs to be transferred to the outsourcing vendor, the people doing these jobs will find it difficult to avoid making value judgments and being critical of the whole endeavor.

There is a risk that some unrecognized or undervalued skills will only become apparent if they are withdrawn, such as in the case of unique knowledge relating to a "legacy" application.

B. Complications of International Outsourcing

Software maintenance and development, the usual target for international outsourcing, requires good communications. English is the generally accepted business language for international software projects.

Even if the vendor is from an English-speaking country, communications will not always be straightforward. When English is a foreign language for the vendor, the challenge becomes that much greater.

Add to these cross-cultural differences, and the subject becomes truly complex. Cross-cultural differences include body language, approaches to decision making, negotiating, conflict resolution, respect for hierarchy and age, the need to save face, and many others that may be totally unknown and/or misunderstood by the client.

Examples of situations commonly occurring in this environment include:

- People from another culture who would prefer to stay silent rather than "lose face" by asking for clarification
- People from another culture who will not volunteer information about problems they have encountered
- People from another culture who have a different concept of time ("by close of business today" is more precise than "as soon as possible")

There are of course many more issues in human and cultural factors, which are the subject of extensive research and publication.

IV. SELECTING A VENDOR

Because of the critical role that information systems and technology play in the success or failure of a company or organization, whenever a decision is made to outsource part or all of the requirements in this area, the process of selecting one or more vendors must be approached as a search for a partner—a vendor who takes responsibility for success.

A mistake in the selection of an outsourcing vendor can have a major impact, and not just in financial terms. In an extreme case, it can paralyze the client. The best defense against such a situation is good preparation and careful planning.

This preparation must include at least three components:

1. The preparation of a request for information and a request for proposals
2. The definition of the criteria and techniques that will be used to select a vendor
3. The decision whether to employ external advisors or to rely exclusively on in-house staff to perform the selection

A. Preparing a Request for Information

Once the objectives of the planned outsourcing have been clearly identified and the costs, benefits, risks, etc., of alternative approaches have been established, it is good practice to issue a request for information (RFI).

Through this mechanism, an outline of what is intended to be outsourced is distributed to potential vendors and, through their replies, information is collected about vendors' capabilities, current client base, and activities. An RFI can and should include a section where potential vendors are invited to put forward their views on what other value-added services they could provide if awarded the contract.

The RFI should make it clear that a vendor's reply is non-binding.

In parallel with this process it is good practice to obtain additional information about potential vendors such as their companies' annual report and independent financial assessments.

B. Preparing a Request for Proposals

A formal request for proposals (RFP) should be a detailed record of the client's requirements. Good preparation is essential, as it is complex, expensive, and frustrating to discover during the evaluation phase that a critical requirement was omitted. Before releasing a proposal, it is prudent to consider obtaining the assistance of consultants specializing in this area, as well as seeking a legal review by a lawyer familiar with the outsourcing industry.

During the preparation of the RFP, it is also good practice to prepare a framework or checklist against which to tabulate the various responses.

The RFP should also require prospective vendors to provide certifiable information, capacity, availability, etc.

C. Proposal Evaluation

It cannot be assumed that all the proposals received will match exactly the scope of services described in the RFP. Particular attention needs to be given to:

- Services and/or features which are excluded
- Performance levels that differ from those requested
- Schedules that do not meet the clients' requirements
- All other differences from the client's original requirements
- Cost by skills set
- Fixed price versus time and materials offers

The next section, preparing for an outsourcing contract, highlights other important areas that need to be examined in detail when evaluating the responses to the RFP.

D. Vendor Selection Criteria

These normally include what comes easily to mind; such as the vendors' technological capabilities, their track record and reputation, the existence of prior working relationships, vendor viability and stability, price, knowledge of the client's business, ISO 9000 and equivalent qualifications, and location (including language and culture), price.

However, many of these criteria have to be described as "soft" and comparisons between vendors may not be easy. Because they are subjective, they can be influenced by the evaluator's knowledge or familiarity with a particular vendor.

Moreover, to demonstrate their track record, vendors will always provide as a reference those clients that are their greatest successes. The evaluators should also ask vendors to provide information about their

dissatisfied clients, and these references must be followed up.

An independent assessment of a vendor's financial viability is particularly valuable when dealing with smaller or more recently established vendors; it can highlight potential financial problems that may inhibit their ability to deliver.

Price will seldom be a determining factor unless (1) it is significantly out of line with the other offerings or (2) the most important objective of the client is to obtain the lowest possible cost.

Several techniques have been developed to minimize or remove bias from complex decisions involving many people and many factors. One such technique is that of Weighted Ranking by Levels (WRBL), and is particularly suitable when:

- There is a need to choose among several alternatives
- The decision needs to be objective
- The decision should be agreed upon by a group

This technique consists of examining alternatives against what the selection group considers to be the most important qualities, which then become decision criteria, and the relative importance (weight) assigned to each. The total of the assigned weights should equal 100.

A table listing the decision criteria and the weight assigned to them is then produced, preferably as individual sheets for each alternative and each member of the evaluation group.

Each alternative is examined against each decision criterion and rated on a scale of 0 = low to 10 = high.

When all the alternatives have been subjected to this analysis, each table is completed by multiplying the weight assigned to each decision criterion by the rating given. The sum of these results becomes the overall rating of each alternative.

The alternative that has scored the highest total is, by definition, the one that best matches the combined decision criteria.

V. PREPARING FOR AN OUTSOURCING CONTRACT

Preparing an outsourcing contract is a complex and specialized activity. The biggest risk in outsourcing functions that have a significant impact on the client's activities or business is that of being contractually committed to an unsatisfactory arrangement.

The overall framework for a formal agreement, which may consist of several contracts, must address many issues over and above the detailed definition of the scope of services and the service agreements (see next section) that apply to each of these services.

One of the major lessons learned by the outsourcing industry is that if an *operation* is not working properly prior to outsourcing, it must be fixed before handing over to a (new) service provider, or be outsourced with the understanding that the outsourcer will have the freedom to implement whatever changes are necessary to put it right.

Negotiating a proper contract requires both parties to deal with the complexities of an outsourcing relationship. This is hard to do when the buyer may be negotiating his first outsourcing contract with a company that has hundreds, if not thousands, of contracts in place.

Good preparation is absolutely vital, as the outsourcing contract will become the foundation upon which the relationship between of the two parties will be based.

Three separate activities need to be considered in contractual terms:

1. The process of transferring from the buyer to the outsourcer
2. Ongoing operations after completion of the transfer, and subsequent changes to the original agreement
3. The process of retuning assets, staff intellectual property, etc., to the buyer should the contract be terminated or other circumstances so require

In addition to the definition of scope of services and service level details, the following should be addressed in sufficient detail.

A. Transfer and Managing the Transition

- Definition of the ownership and valuation of all assets to be transferred
- Ownership and reassignment of leases, licences, etc.
- Proprietary software: who retains ownership, who has access to the source code, exploitation rights of the outsourcer, terms of supply of documentation
- Knowledge transfer (what, when and how)
- Identification of the staff to be transferred
- Involvement of third parties (vendors, lessors and other financial institutions, disaster recovery operators, etc.)
- Disaster recovery services, documentation, test results, etc.
- Issues related to premises

Unless the selected outsourcer is thoroughly familiar with the activities and organizational culture of the client, the transition period should be used to refine and validate such matters as objectives, measurement procedures, and reporting arrangements.

The transfer process should be approached as a major project by both parties. High-level project management must be in place to deal with all supplier contracts and documentation, and appropriate controls must be in place. Milestones and the schedule of payments should be clearly defined.

B. Ongoing Activities

- Start date
- What can be shared with other clients of the outsourcer and what needs to be dedicated to the buyer?
- Disaster recovery arrangements (keep or change)
- Pricing arrangements (fixed cost, cost plus, benchmarks, benefits sharing, etc.)
- Copyright, intellectual property, confidentiality, and data protection (licensing patents and trademarks to the outsourcer) need to have clear communications to the staff involved and need to comply with national/international legislation such as data protection
- Change control procedures
- Client rights to audit the services, charges, etc., of the vendor
- Warranties and indemnities (service levels, consequential losses, extra costs)
- Management of the relationship (appointed representatives, regular meetings, right to audit)
- Schedule of payments

Vendors will invariably propose their own standard terms and conditions, these are designed to be biased in their favor. A well-negotiated variant of such contracts or an individually tailored contract would be the preferred option to maintain a balance between the parties. This, however, is a major undertaking involving lawyers from both parties, and will increase the cost of the arrangement.

While all discussions relating to the contract should be treated as confidential and subject to incorporation in the final contract, promises made by sales people should be obtained in writing to avoid subsequent disputes.

An important part of these negotiations must focus on liabilities. Vendors will strive to keep their liabilities to a minimum. This is not unreasonable given that the price of the contract must include the cost of insurance and/or the risk of failure, which could be significant in unusual or complex situations. Other clauses concerning liabilities are those of consequential loss (for example, of profit or of goodwill), which are particularly hard to quantify. The possibility of obtaining liability insurance should be investigated.

National legislation should be consulted for topics such as injury, death, fraud, and breach of third-party intellectual property, as these may be the subjects of unlimited liabilities.

Copies of the contract should be distributed to all parties who may need them, and the production of a plain language summary may be a valuable supplementary document.

The contract should also contain detailed procedures for amendment (change of scope or deliverables) and termination including definitions of the period of agreement, notice of termination, and special instances (insolvency/breach of contract).

Additionally, the contract should specify mechanisms for the resolution of disputes and arbitration, including definition of legal framework that will apply (country or place of resolution).

VI. SERVICE AGREEMENTS

Outsourced processes are invariably the subjects of a Service Level Agreement (SLA). This defines the terms under which a vendor will provide a service. As the main purpose of an SLA is to protect the technology services that are important to a company or organization, it is essential that it hold the vendor responsible for delivering services at the appropriate level of quality.

A good SLA is best prepared by a team consisting of business unit managers, legal advisors, and information technologists.

For the outsourcing of existing processes, the client should be able to define current service levels in the appropriate metrics, and notify the selected vendor, who would normally require time to validate and verify these.

Performance measurement criteria for processes usually include:

- Service availability and how it is defined
- Response time and where it is measured
- Definition of maintenance windows
- Problem resolution targets

When outsourcing processes that the client does not currently operate, as would be the case of outsourcing e-business activities from the outset, the client needs

to acquire a clear understanding of the vendor's proposed service levels and how these are defined.

It is essential that there should be no ambiguity in any of these definitions. For example, there are fundamental differences in the specification of a service availability of 99.8% under the following conditions:

- Monday to Friday, prime time (08:00–18:00) at the vendor's time zone
- Seven days a week, over two consecutive working shifts (08:00–24:00) at the vendor's time zone, as would be the case in a global operation
- Seven days a week, twenty-four hours a day

In the case of projects, such as software maintenance and development, performance metrics are considerably more complex. Typical metrics for projects include:

- On-budget and on-time delivery of agreed work, management of the relationship (appointed representatives, regular meetings, right to audit)
- Definition of individual work products in terms of quantity of work content
- Definition of measurable quality items for individual work products

Service Level Agreements constitute a major topic in their own right and two books on SLAs are suggested in the bibliography.

VII. SUCCESS FACTORS

Whenever outsourcing is decided upon, it is vital that it should be successful. One of the lessons that has been learned by the outsourcing industry and its clients is that once a contract is in place, there are frequently few alternatives available to the client but to continue with the vendor originally selected. The quotation that . . . "it is easier to divorce your spouse than to separate from your outsourcer" . . . may be unattributed but is frequently encountered.

There is general agreement in the outsourcing industry that the main success factors are viability of the outsourcing option, thorough understanding of the scope of outsourcing, and realistic expectations, legal support, and relationship management.

A. Viability of the Outsourcing Option

The objectives of the client, whether they are cost reduction, the achievement of accounting benefits, offloading of noncore activities, releasing staff for other work/projects, or sharing risk and/or rewards with the vendor, must be realistic and achievable, as well as understood by all parties.

Throughout this process, the client must develop a clear understanding of the consequences of outsourcing contracts on such matters as its impact on staff as well as its financial, taxation, and legal implications.

Outsourcing something that does not work does not make things better. It is essential that the client should fix the shortcomings of present processes before outsourcing or that a third party, possibly the chosen vendor, should create a project to address the shortcomings before the step to outsourcing is taken.

In the case of outsourcing software projects, the most successful invariably involved clients who had the capability of developing the software themselves and opted instead to outsource in order to use staff for other projects with higher business value.

B. Thorough Understanding of the Scope of Outsourcing, and Realistic Expectations.

In taking these steps, the buyer must ensure that in preparing to outsource, certain client's responsibilities are not ignored. These include:

- Definition of the scope of services to be provided
- For process work, a clear and true definition of the current service levels (which may be validated by the selected outsourcer)
- Definition of the target service levels expected from the outsourcer
- Definition of, and agreement to, the division of responsibilities between buyer and outsourcer
- Definition of the management controls to be put in place, including the right to audit the outsourcer
- Identification of the staff that will remain with the buyer and of the staff that will be transferred to the outsourcer
- Management of the organizational and other changes required as a result of outsourcing
- For project work, definition of performance metrics
- Definition of all the measures to be taken in order to protect the client's intellectual property, confidential data integrity, etc.

C. Legal Support

It is strongly recommended that a lawyer with knowledge and experience of outsourcing contracts should be involved from the preparation of the RFP until the

contract is signed. The same applies to all amendments to the contract.

D. Relationship Management

The client should strive to identify a vendor well-suited to meet the overall objectives of the particular outsourcing requirements, and in particular, an honorable and well managed vendor whose prime objective is to maintain its reputation.

The relationship between client and vendor begins at the time of contract negotiation, and will be a key component of service delivery for the duration of the contract. Success in this area requires the existence of good communications arrangements, frequent contact, and the goodwill necessary to build an effective relationship.

Formal review meetings should be held at least on a quarterly basis during the first year of a relationship.

Litigation should be avoided until such time as all other measures have been exhausted. Litigation is expensive and time consuming. The outcome of litigation never has winners, only losers.

SEE ALSO THE FOLLOWING ARTICLES

Cost/Benefit Analysis • Data, Information, and Knowledge • End-User Computing Tools • Group Support Systems and Electronic Meeting Systems • Organizations, Information Systems Impact on • Project Management Techniques • Staffing the Information Systems Department • Strategic Planning • Telecommuting • Virtual Organizations

BIBLIOGRAPHY

Chang, R. Y., and Niedzwiecki, M. E. (1995). *Continuous improvement tools.* Volume 1. Kogan Page Ltd: London. http://www.outsourcing-experts.com

Cutter IT Journal: Special issue on Outsourcing. (July 1998). Vol. 11, No. 7.

Cutter IT Journal: Special issue on Outsourcing. (October 1999). Vol. 12, No. 10

Hiles, A. (1999). *Complete guide to IT service level agreements: Matching service quality to business needs.* Rothstein Associates.

Hiles, A. (2000). *Service level agreement: Winning a competitive edge for support and supply services.* Rothstein Associates.

Mead, R. (1998). *International management: Cross-cultural dimensions.* Oxford: Blackwell.

Pascal

Bill M. Catambay
ExcaliburWorld Software

GLOSSARY

compiler A program which translates a source text written in a specific programming language into a suitable instruction sequence which can be processed by a computer.
extended Pascal Defined by ISO 10206, a version of the Pascal programming language which incorporates several extensions that go beyond the original unextended Pascal.
International Organization for Standardization (ISO) A nongovernmental organization is responsible for establishing and approving technical standards, resulting in international agreements which are published as International Standards.
object Pascal An implementation of Pascal that supports object-oriented programming (OOP), as defined in the "Object-Oriented Extensions to Pascal" X3J9 report.
rapid application development (RAD) A software development process that allows usable systems to be built in as little as 60–90 days, often with some compromises.
unextended Pascal The original Pascal implementation, as defined in ISO 7185.

I. ORIGINS OF PASCAL

The Pascal language was named for Blaise Pascal, a French mathematician who was a pioneer in computer development history. In 1641, at the age of 18, Pascal constructed the first arithmetical machine, arguably the first computer. He would improve upon the instrument 8 years later. In 1650, Pascal left the world of geometry and physics and shifted his focus toward religious studies or, as Pascal wrote, to "contemplate the greatness and the misery of man." Pascal died in Paris on August 19, 1662.

The earliest computers were programmed in machine code and assembly. This type of programming is tedious and error prone, as well as extremely difficult to understand and modify. Programming is a time-consuming and expensive process. High-level languages were developed to resolve this problem. High-level languages provide a set of instructions that read like English, but can be translated by a program called a compiler into machine code. Pascal is one such language.

Other high-level languages developed in the early years of the computer were FORTRAN (1957), COBOL (1959), ALGOL (1960), APL (1962), BASIC (1964), C (1972), and Ada (1983), to name a few. One problem with many of the early languages (e.g., FORTRAN and BASIC) was the heavy dependency on the use of "goto" instructions. "Goto" instructions tell the computer to jump from one step to another, enabling the computer to skip steps or to go back to repeat earlier steps. This type of sporadic branching increases the difficulty of debugging code. Additionally, languages such as COBOL were designed with overelaborate definitions, weak data structures support, and a lack of flexibility, making programs tedious to code and difficult to enhance.

Niklaus Wirth completed development of the original Pascal programming language in 1970. He based

it upon the block structured style of the Algol programming language. There were two original goals for Pascal. According to the Pascal standard, these goals were to (1) make available a language suitable for teaching programming as a systematic discipline based on fundamental concepts clearly and naturally reflected by the language and (2) to define a language whose implementations could be both reliable and efficient on then-available computers.

Pascal went far beyond its original design goals, with commercial use of the language often exceeding academic interest. Pascal provides rich data structures, including both the enumerated and the record data types, and defined with a pleasing and powerful clarity. It provided an orthogonal and recursive approach to data structures, with arrays of arrays, arrays of records, records containing arrays, files of records, files of arrays, files of records containing arrays of records, and so on. Pascal's popularity exploded in the 1970s, as it was used in writing both system and application software. For this reason, the International Standards Organization (ISO) committee decided that a formal standard was needed to promote the stability of the Pascal language (the ISO 7185 Pascal standard was originally published in 1983). By the end of the 1970s, more than 80 computer systems had Pascal implementations in use.

One of the more popular Pascals of the 1970s and early 1980s was UCSD Pascal on the UCSD P-System operating system. The UCSD P-System was developed at the Institute for Information Studies at the University of California at San Diego, under the direction of Kenneth Bowles. In fact, the P-System operating system itself was written in UCSD Pascal. As Wirth writes in "From Programming Language Design To Computer Construction," his 1985 Turing Award Lecture, "But Pascal gained truly widespread recognition only after Ken Bowles in San Diego recognized that the P-system could well be implemented on the novel microcomputers. His efforts to develop a suitable environment with integrated compiler, filer, editor, and debugger caused a breakthrough: Pascal became available to thousands of new computer users who were not burdened with acquired habits or stifled by the urge to stay compatible with software of the past."

In 1978, Richard Gleaves and Mark Allen, working on-campus in San Diego, used UCSD Pascal to develop the 6502 interpreter which became the basis for Apple Pascal. By the 1980s, Pascal was used by most universities to teach programming, while still invading the commercial markets. It became so popular that even FORTRAN began to change, taking advantage of Pascal's innovations.

Due to the strong popularity of the Pascal language in system and application software development, and in response to the many cited drawbacks of the original Pascal implementation, an Extended Pascal evolved to address the needs of commercial development. In 1990, the ISO 10206 Extended Pascal standard was published to support this new version of the language.

In addition to Extended Pascal, in 1986 Apple Computer released the first Object Pascal implementation, a version of its Apple Pascal that supported object-oriented programming. In 1993, the Pascal standards committee published an "Object-Oriented Extensions to Pascal" technical report (Technical Committee X3J9, 1993) which was based upon Apple's Object Pascal implementation.

II. THE PASCAL ARCHITECTURE

Pascal is a strongly typed, block-structured programming language. The "type" of a Pascal variable consists of its semantic nature and its range of values and can be expressed by a type name, an explicit value range, or a combination thereof. The range of values for a type is defined by the language itself for built-in types or by the programmer for programmer-defined types. Programmer-defined types are unique data types defined within the Pascal `TYPE` declaration section and can consist of enumerated types, arrays, records, pointers, sets, and more, as well as combinations thereof. When variables are declared as one type, the compiler can assume that the variable will be used as that type throughout the life of the variable (whether it is global to the program or local to a function or procedure). This consistent usage of variables makes the code easier to maintain. The compiler detects type inconsistency errors at compile time, catching many errors and reducing the need to run the code through a debugger. Additionally, it allows an optimizer to make assumptions during compilation, thereby providing more efficient executables. As John Reagan, the architect of Compaq Pascal, writes, "it was easy to write Pascal programs that would generate better code than their C equivalents" because the compiler was able to optimize based on the strict typing.

Declaring variables in Pascal is straightforward. The Pascal `VAR` declaration section gives the programmer the ability to declare strings, integers, real numbers, and booleans (to name a few built-in types), as well as to declare variables as records or other programmer-defined types. A variable defined as a `RECORD` allows a single variable to track several data components (or fields) (Fig.1).

```
Type
    Employee_type =   (Hourly, Salary, SalaryExempt);
    InputRec   =   RECORD
        emp_name: packed array[1..30 ] of char;
        social:   packed array[1..9] of char;
        salary:   real;
        emp_type: Employee_type;
        end;

Var
    index:     integer;
    ratio:     real;
    found:     boolean;
    inpf:      file of InputRec;
```

Figure 1 Sample code—types and records.

Pascal also supports recursion, a powerful computing tool that allows a function or procedure within a program to make calls to itself. This allows for elegant and efficient coding solutions, eliminating the need for tedious loops. A good example of recursion is the Pascal solution to the classic Towers of Hanoi puzzle (Fig. 2). The puzzle is to take a stack of disks in increasing sizes from top to bottom, and move them from the first peg to the second peg, with the rule that a disk must always be placed on a disk larger than itself, and only one disk can be moved at a time.

The solution to the Towers of Hanoi puzzle involves moving all but one disk from peg to peg, repeatedly, until the entire stack has moved from one peg to the other peg. The elegance of recursion is that this solution is illustrated clearly, without mundane loops and logic checks. The three steps of the solution are depicted by three lines of code. The movement of a stack, regardless of size, is always done by a call to Hanoi, thus ensuring that the rules are adhered to.

Pascal eliminates the need for clumsy goto statements by supporting `REPEAT/UNTIL`, `WHILE/DO`, and `FOR` loops; by providing an intelligent `CASE` statement; and by providing a means to consolidate common lines of code into `PROCEDURE`s and `FUNCTION`s. Using the words `BEGIN` and `END` to delimit blocks of code within a clause, enforcing strong typing, providing ordinal-based arrays, and other useful linguistic features, Pascal facilitates the production of correct, reliable, and maintainable code. Any language can be commented and indented for better readability, but Pascal, by the nature of its syntax and architecture, encourages structured programming practices and allows the programmer to focus on developing solutions. It is important to emphasize this element of Pascal. Even programmers with the most sophisticated and disciplined of programming styles will find themselves in a time crunch. With deadlines quickly approaching, it is likely that a programmer will focus more on achieving a result and less on making the code understandable for future maintenance. The key to Pascal is that a programmer tasked with maintaining Pascal code will be able to make the same assumptions that the compiler makes about program flow and data usage. This gives the maintenance programmer a fighting chance of figuring out the behavior, purpose, and operating conditions of the code, even if it is poorly written.

```
Program TowersOfHanoi(input,output);

Var
   disks: integer;

Procedure Hanoi(source, temp, destination: char;  n: integer);

   begin
   if n > 0 then
      begin
      Hanoi(source, destination, temp, n - 1);
      writeln('Move disk ',n:1,' from peg ',source,
                    ' to peg ',destination);
      Hanoi(temp, source, destination, n - 1);
      end;
   end;

begin
write('Enter the number of disks: ');
readln(disks);
writeln('Solution:');
Hanoi('A','B','C',disks);
end.
```

Figure 2 Sample recursive code—Towers of Hanoi solution.

A. Block Structure

In *An Introduction to Programming and Problem Solving with Pascal,* the author write, "A block is a sequence of declarations, a begin, a sequence of statements that describes actions to be performed on the data structures described in the declarations, and an end." A Pascal program consists of a PROGRAM heading, which names the program, followed by a block. Within that main program block, there exist subprograms, each of which have their own heading followed by a block. Within each block, there can be inner blocks, and within each inner block, there can exist further inner blocks. In essence, a Pascal program is a hierarchical construction of blocks; hence, Pascal is a block-structured programming language.

All data values declared at the beginning of a block are accessible to the code within the block, including inner blocks, but not to any others. The usefulness of a block-structured language, therefore, is not only the modularization of the program, but also the protection of data that is exclusive to one set of modules within a program from compromise by other modules.

B. Style

The flow of Pascal code often reads like plain English, with code indentation playing a crucial role in visualizing conditional clauses. The BEGIN and END statements are key elements of Pascal's architecture. These clause delimiters are often misunderstood and misused, leading even the most avid Pascal programmers to question their usefulness. Used properly, however, they are a vital to visualizing code clauses within a program. Take, for example, the Towers of Hanoi solution (Fig. 3).

The BEGIN and END statements are bold in Fig. 3 to illustrate a point. While these words are not typically bold in an editor, when a programmer is trained on these words, they are just as visually apparent. Further, when setting the indentation of the clause to match the BEGIN and END statement, a virtual line of sight is established. When clauses are imbedded within other clauses, sometimes several layers deep, these lines of sight become very helpful in deciphering the hierarchy of clauses and flow of execution. Less work is required to understand and rediscover the flow of a program, and the programmer can therefore focus on the logic and algorithms.

In a small program with very few lines of code, this advantage is not quite as apparent. The usefulness, however, increases exponentially as complexity and size increase. For example, take an extensive program of several pages with long clauses that may pass through several page breaks (Fig. 4).

The virtual lines of sight are illustrated in Fig. 4 with vertical lines, and the code itself is purposely blurred to focus attention on the indentation. In a large program, determining the program flow is required each time the programmer attacks the code. The quicker that flow becomes apparent, the sooner the programmer can get to work at updating the code.

```
Program TowersOfHanoi(input,output);

Var
   disks: integer;

Procedure Hanoi(source, temp, destination: char;  n: integer);

   begin
   if n > 0 then
      begin
      Hanoi(source, destination, temp, n - 1);
      writeln('Move disk ',n:1,' from peg ',source,
                  ' to peg ',destination);
      Hanoi(temp, source, destination, n - 1);
      end;
   end;

begin
write('Enter the number of disks: ');
readln(disks);
writeln('Solution:');
Hanoi('A','B','C',disks);
end.
```

Figure 3 Indentation style and block-structured code.

Figure 4 Indentation of block-structured code at a glance.

C. Manageability

Programmers at the beginning of a project face creating a solution to a real-world problem using computer code. Over time, however, programmers face the ongoing problem of maintaining and enhancing the computer code as the users' needs change and grow. Increasing the manageability of code, both for initial implementation and for long-term maintenance, decreases the amount of effort required to work the problem.

Pascal increases manageability of code by enforcing strong typing, supporting block-structured programming, and providing a syntax which is easy to read. These aspects of Pascal provide both immediate and long-term benefits to the programmer. Strong typing removes much of the guesswork from interpreting data structures. Block-structured programming breaks down a program into a hierarchy of tasks. The decomposition of a programming problem into a hierarchy of tasks enhances the manageability of the problem. Finally, the more visually apparent the blocks are within a program, and the more the code reads like English, the easier it is to interpret the program flow, thereby further increasing manageability.

These aspects of Pascal provide the programmer tools for long-term manageability in supporting legacy code as well. Each time a programmer attacks legacy code, the logic flow and data structures must be understood before any work can be done efficiently. Guessing incorrectly at what the code or data structures were designed to do can lead to costly bugs in the software. The sooner the programmer can understand the code clearly and with confidence, the quicker the programmer can get "into the groove" of debugging or enhancing the code. Each block, variable, and line of code may represent only seconds of time saved using Pascal, but these seconds add up. A penny saved may not seem like much, but a penny saved every second, 60 hours a week, 50 weeks a year is over $100,000. Moreover, when it takes a very long time to find the groove, the effort to understand the code becomes so frustrating that it impedes creativity and productivity.

III. PASCAL STANDARDS

Language standards were developed so that when code is written on one type of computer, or with one vendor's compiler, the code can be ported to another computer or compiler and still compile and run correctly. This is not a foolproof plan, however, as there are many unique behaviors of different computers. A program that invokes the unique behavior of a specific computer will have to be changed to work on

another computer. Further, most computer environments offer libraries for performing commonly used functions. If the program makes several calls to the system libraries, those calls will most likely need to be replaced when porting the code to another computer, even if the compiler is fully compliant with its languages standards.

Although standards do not resolve these issues, they do provide a certain amount of consistency within a language construct. The same code may not function, or perhaps even compile, when ported directly to another computer or compiler, but because the standards exist, a programmer familiar with that language will have a basic understanding of what the code is doing. Given an understanding of the libraries and unique properties of the computer to which the code is being ported, the task of porting the code is easier than if there were no language standards at all.

The first standard written for Pascal was developed in 1983, covering what is known as Unextended Pascal (ISO 7185). In 1990, the same year that the Unextended Pascal standard was updated, the Extended Pascal standard (ISO 10206) was established. The Unextended Pascal standard incorporated basic functionality of the original Pascal, while the Extended Pascal standard was introduced to bring Pascal more in line with modern programming needs, thus providing the programmer a more powerful programming tool without sacrificing the elegance of Pascal.

To further meet the demands of the growing technology in computer programming, certain Pascal compilers were established to support object-oriented programming. Although an official standard for Object Pascal has not been established at this writing, in 1993, the Pascal standards committee published an "Object-Oriented Extensions to Pascal" technical report (Technical Committee X3J9, 1993) which provides proposed standards. The members of the committee that assembled this report came from a variety of organizations—from Pace University and the U.S. Air Force to Apple Computer, Microsoft, and Digital Equipment Corporation.

Finally, in 1995, John Reagan, a member of the ISO Pascal standards committee and compiler architect for Digital Equipment Corporation (now Compaq), composed a Pascal standards frequently asked questions (FAQs) list. The FAQ addressed questions such as:

- What are the different Pascal standards?
- Who creates the standards?
- What are the required interfaces to Extended Pascal?
- What is the history of Pascal standards?

All of the standards and reports mentioned above, including the Pascal standards FAQ, are available from the Pascal Central Web site at http://pascal-central.com.

Direct links to specific documents are as follows:

- Unextended and Extended Pascal Standards: http://pascal-central.com/standards.html
- Object Pascal Report: http://pascal-central.com/ooe-stds.html
- Pascal Standards FAQ: http://pascal-central.com/extpascal.html

A. Unextended Pascal

The unextended Pascal standard is incorporated in ISO 7185. The material covered in the standard is too extensive to discuss here, but the following section provides a summary of ISO 7185.

Unextended Pascal token symbols are reserved symbols used by the compiler to perform operations and calculations (Table I).

Additionally, the following tokens are Pascal comment delimiters. When placed around text, they indicate text that is not meant for compilation.

```
{ and }, (* and *)
```

Unextended Pascal token words are reserved words used by the compiler to perform operations and calculations (Table II).

Unextended Pascal supports three categories of data types: simple, structured, and pointer.

The simple data type consists of predefined, enumerated, and subrange types.

The predefined simple types are:

- `INTEGER`: integer value
- `REAL`: real number value
- `CHAR`: single character
- `BOOLEAN` takes values true or false

In unextended Pascal, the enumerated type is defined to allow programmers to establish types based upon a unique and finite list of values. For example,

```
Type
  ShirtColor = (green, blue, red);
  ShirtSize = (small, medium, large);
```

Likewise, unextended Pascal supports subrange types, such as 1..100, −10 . . . +10, and "0". . ."9".

The structured data type consists of array, file, record, and set types. All four can be optionally de-

Table I Unextended Pascal Special Symbol Tokens

+	(PLUS)	Binary arithmetic addition; unary arithmetic identity; set union
-	(MINUS)	Binary arithmetic subtraction; unary arithmetic negation; set difference
*	(ASTERISK)	Arithmetic multiplication; set intersection
/	(SLASH)	Floating point division
=	(EQUAL)	Equality test
<	(LESS THAN)	Less than test
>	(GREATER THAN)	Greater than test
[	(LEFT BRACKET)	Delimits sets and array indices
]	(RIGHT BRACKET)	Delimits sets and array indices
.	(PERIOD)	Used for selecting an individual field of a record variable; follows the final END of a program
,	(COMMA)	Separates arguments, variable declarations, and indices of multi-dimensional arrays
:	(COLON)	Separates a function declaration with the function type; separates variable declaration with the variable type
;	(SEMI-COLON)	Separates Pascal statements
^	(POINTER)	Used to declare pointer types and variables; used to access the contents of pointer typed variables/file buffer variables
(	(LEFT PARENTHESIS)	Group mathematical or boolean expression, or function and procedure arguments
)	(RIGHT PARENTHESIS)	Group mathematical or boolean expression, or function and procedure arguments
<>	(LESS THAN/GREATER THAN)	Nonequality test
<=	(LESS THAN/EQUAL)	Less than or equal to test; subset of test
>=	(GREATER THAN/EQUAL)	Greater than or equal to test; superset of test
:=	(COLON/EQUAL)	Variable assignment
..	(PERIOD/PERIOD)	Range delimiter

clared as PACKED for cases where the programmer wants to minimize storage requirements albeit with the potential cost of greater access time to the individual components.

An array in unextended Pascal consists of a fixed number of components (i.e., a linear vector) which are all the same type. An array's index type establishes the number of components the array contains. Since any ordinal type can be used for the index type, arrays indexed by character, enumerated, or subrange types can be used for a more natural fit to the problem. Because a component type can be an array type in itself, multidimensional arrays are possible. For multidimensional arrays, a consolidated list of dimension indices can be used as an abbreviation alternative. For example,

```
TYPE
  ShirtStock = array[ShirtColor] of
     array[ShirtSize] of integer;
  ShirtPrice = packed array[ShirtColor,
     ShirtSize] of real;
```

The file type consists of a linear sequence of components all of the same type. The number of components is not fixed. Pascal predefines TEXT as a file type. The TEXT file type consists of a sequence of characters subdivided into variable length lines. A special end-of-line marker is used for line subdivision. Example file types are:

```
TYPE
  Document = file of integer;
  Data = packed file of real;
```

Unextended Pascal supports the RECORD type, which allows the programmer to establish a type containing several components (or fields). In addition to being able to declare fields of varying types, variant records can also be declared such that different record layouts exist based upon the value for the variant field (Fig. 5).

A set type consists of the powerset of the set of values of the set's base type. Every value in the set's base type can be represented as an element in the set. Example set types are:

Table II Unextended Pascal Word Symbol Tokens

AND	Boolean conjunction operator
ARRAY	Array type
BEGIN	Starts a compound statement
CASE	Starts a CASE statement
CONST	Declares a constant
DIV	Integer division
DO	Follows WHILE and FOR clause, preceding action to take
DOWNTO	In a FOR loop, indicates that FOR variable is decremented at each pass
ELSE	If the boolean in the IF is false, the action following ELSE is executed
END	Ends a compound statement, a case statement, or a record declaration
FILE	Declares a variable as a file
FOR	Executes line(s) of code while FOR loop variable in within range
FUNCTION	Declares a Pascal function
GOTO	Branches to a specified label
IF	Examine a boolean condition and execute code if true
IN	Boolean evaluated to true if value is in a specified set
LABEL	Indicates code to branch to in a GOTO statement
MOD	Modular integer evaluation
NIL	Null value for a pointer
NOT	Negates the value of a boolean expression
OF	Used in CASE statement after case variable
OR	Boolean disjunction operator
PACKED	Used with ARRAY, FILE, RECORD, and SET to pack data storage
PROCEDURE	Declares a Pascal procedure
PROGRAM	Designates the program heading
RECORD	Declares a record type
REPEAT	Starts a REPEAT/UNTIL loop
SET	Declares a set
THEN	Follows the boolean expression after an IF statement
TO	In a FOR loop, indicates that FOR variable is incremented at each pass
TYPE	Defines a variable type
UNTIL	Ends a REPEAT/UNTIL loop
VAR	Declares a program variable
WHILE	Executes block of code until WHILE condition is false
WITH	Specifies record variable to use for a block of code

```
TYPE
  CharSet = set of char;
  SizeSet = packed set of ShirtSize;
```

Unextended Pascal also supports pointer types, a method of referencing a variable using its address. Pointers are often used for referencing record structures, supporting linked lists, an array of data which is connected by means of pointers rather than a sequential index (Fig. 6).

Refer to the ISO 7185 (ISO, 1990a) for a complete list of unextended Pascal predefined functions, types, and other language constructs.

B. Extended Pascal

The Extended Pascal standard is incorporated in ISO 10206. The material covered in this document is summarized below.

```
Type
    ShirtOrder = record
        color:        ShirtColor;
        size:         ShirtSize;
        customer:     string;
        city:         string;
        case USCust: boolean of
            true:     (USState: packed array[1..2] of char);
            false:    (state: string;
                       country: string);
        end;
```

Figure 5 Example of a variant record.

In addition to the token symbols reserved for unextended Pascal, Extended Pascal supports three additional tokens (see Table III).

In addition to the token words reserved for unextended Pascal, Extended Pascal requires support for several new token words (Table IV).

The following, from ISO 10206, is a summary of features in Extended Pascal that are not found in unextended Pascal.

1. Modularity and separate compilation. Modularity provides for separately compilable program components, while maintaining type security.
- Each module exports one or more interfaces containing entities (values, types, schemata, variables, procedures, and functions) from that module, thereby controlling visibility into the module.
- A variable may be protected on export, so that an importer may use it but not alter its value. A type may be restricted, so that its structure is not visible.
- The form of a module clearly separates its interfaces from its internal details.
- Any block may import one or more interfaces. Each interface may be used in whole or in part.
- Entities may be accessed with or without interface-name qualification.
- Entities may be renamed on export or import.
- Initialization and finalization actions may be specified for each module.

```
Type
    Person        =  ^Persondetails;
    Persondetails =  record
        name:         String;
        firstname:    String;
        age:          Integer;
        married:      boolean;
        nextPerson:   Person;
        prevPerson:   Person;
        end;
```

Figure 6 Example of a linked list record structure.

- Modules provide a framework for implementation of libraries and non-Pascal program components.

2. Schemata. A schema determines a collection of similar types. Types may be selected statically or dynamically from schemata.
- Statically selected types are used as any other types are used.
- Dynamically selected types subsume all the functionality of, and provide functional capability beyond, conformant arrays.
- The allocation procedure NEW may dynamically select the type (and thus the size) of the allocated variable.
- A schematic formal parameter adjusts to the bounds of its actual parameters.
- The declaration of a local variable may dynamically select the type (and thus the size) of the variable.
- The with statement is extended to work with schemata.
- Formal schema discriminants can be used as variant selectors.

3. String capabilities. The comprehensive string facilities unify fixed-length strings and character values with variable-length strings.
- All string and character values are compatible.
- The concatenation operator (+) combines all string and character values.
- Variable-length strings have programmer-specified maximum lengths.
- Strings may be compared using blank padding via the relational operators or using no padding via the functions `EQ`, `LT`, `GT`, `NE`, `LE`, and `GE`.
- The functions length, index, substr, and trim provide information about, or manipulate, strings.
- The substring-variable notation makes accessible, as a variable, a fixed-length portion of a string variable.
- The transfer procedures readstr and writestr process strings in the same manner that read and write process text files.

Table III Extended Pascal Special Symbol Tokens

**	(ASTERISK/ASTERISK)	To the real power of
><	(GREATER THAN/LESS THAN)	Set symmetric difference
=>	(EQUAL/GREATER THAN)	Renames identifiers on import and/or export

- The procedure read has been extended to read strings from text files.

4. Binding of variables. A variable may optionally be declared to be bindable. Bindable variables may be bound to external entities (file storage, real-time clock, command lines, etc.). Only bindable variables may be so bound.

- The procedures bind and unbind, together with the related type BindingType, provide capabilities for connection and disconnection of bindable internal (file and nonfile) variables to external entities.
- The function binding returns current or default binding information.

5. Direct access file handling. The declaration of a direct access file indicates an index by which individual file elements may be accessed.

- The procedures SeekRead, SeekWrite, and SeekUpdate position the file.
- The functions position, LastPosition, and empty report the current position and size of the file.
- The update file mode and its associated procedure update provide in-place modification.

6. File extend procedure. The procedure extend prepares an existing file for writing at its end.
7. Constant expressions. A constant expression may occur in any context needing a constant value.
8. Structured value constructors. An expression may represent the value of an array, record, or set in terms of its components. This is particularly valuable for defining structured constants.
9. Generalized function results. The result of a function may have any assignable type. A function result variable may be specified, which is especially useful for functions returning structures.
10. Initial variable state. The initial state specifier of a type can specify the value with which variables are to be created.
11. Relaxation of ordering of declarations. There may be any number of declaration parts (labels, constants, types, variables, procedures, and functions) in any order. The prohibition of forward references in declarations is retained.
12. Type inquiry. A variable or parameter may be declared to have the type of another parameter or another variable.
13. Implementation characteristics. The constant maxchar is the largest value of type char. The constants minreal, maxreal, and epsreal describe

Table IV Extended Pascal Word Symbol Tokens

AND_THEN	Boolean operator w/short circuiting
BINDABLE	Bindable to some entity external to the program
EXPORT	Exporting to/from modules
IMPORT	Importing to/from modules
MODULE	Can be compiled, but not executed, by itself
ONLY	Selective import option
OR_ELSE	Boolean operator w/shortcircuiting
OTHERWISE	Default condition handler in CASE statement
POW	To the integer power of
PROTECTED	Protects exported variables from being altered by importors; protects function/procedure parameter from being altered by function/procedure
QUALIFIED	Qualified import specification
RESTRICTED	Creation of opaque data types
VALUE	Specifies initial state for component

the range of magnitude and the precision of real arithmetic.
14. Case statement and variant record enhancements. Each case-constant list may contain ranges of values. An otherwise clause represents all values not listed in the case-constant lists.
15. Set extension.
- An operator (><) computes the set symmetric difference.
- The function card yields the number of members in a set.
- A form of the for statement iterates through the members of a set.

16. Date and time. The procedure GetTimeStamp and the functions date and time, together with the related type TimeStamp, provide numeric representations of the current date and time and convert the numeric representations to strings.
17. Inverse ord. A generalization of succ and pred provides an inverse ord capability.
18. Standard numeric input. The definition of acceptable character sequences read from a text file includes all standard numeric representations defined by ISO 6093.
19. Nondecimal representation of numbers. Integer numeric constants may be expressed using bases 2–36.
20. Underscore in identifiers. The underscore character (_) may occur within identifiers and is significant to their spelling.
21. Zero field widths. The total field width and fraction digits expressions in write parameters may be zero.
22. Halt. The procedure halt causes termination of the program.
23. Complex numbers.
- The simple-type complex allows complex numbers to be expressed in either Cartesian or polar notation.
- The monadic operators + and – and dyadic operators `+, -, *, /, =, <>` operate on complex values.
- The functions cmplx, polar, re, im, and arg construct or provide information about complex values.
- The functions abs, sqr, sqrt, exp, ln, sin, cos, and arctan operate on complex values.

24. Short circuit boolean evaluation. The operators AND_THEN and OR_ELSE are logically equivalent to AND and OR, except that evaluation order is defined as left to right, and the right operand is not evaluated if the value of the expression can be determined solely from the value of the left operand.
25. Protected parameters. A parameter of a procedure or a function can be protected from modification within the procedure or function.
26. Exponentiation. The operators ** and pow provide exponentiation of integer, real, and complex numbers to real and integer powers.
27. Subrange bounds. A general expression can be used to specify the value of either bound in a subrange.
28. Tag fields of dynamic variables. Any tag field specified by a parameter to the procedure new is given the specified value.

Additionally, Extended Pascal incorporates the following feature at level 1 of this standard:
29. Conformant arrays. Conformant arrays provide upward compatibility with level 1 of ISO 7185, programming languages—PASCAL.

C. Object Pascal

There is no official Object Pascal standard, but a technical report was published in 1993 outlining the recommended standards for Object Pascal. A draft of this report is available on the Web at http://pascal-central.com/ooe-stds.html. An outline of the report is shown in Fig. 7. Further information can be obtained at the previously mentioned Web page.

IV. MYTHS UNCOVERED

The fast-moving pace of technology and the variety of platforms different compilers support make it easy for multiple dialects of the same programming language to evolve. Although standards exist, compiler vendors have not chosen to comply fully with them. The result is a variety of Pascal dialects with significant differences in performance.

Most Pascal compilers on the market support the Unextended Pascal standard. Most support an expanded subset of the Extended Pascal standard. Programmers often pass judgment on the Pascal language based on their experience with one particular Pascal dialect, without knowing whether that dialect complies with the standards or how it compares to other available implementations. In several of the dialogs over the past 10 years on the USENET newsgroups, negative views of Pascal have been based upon preconceptions

I. Introduction
II. Scope
III. References
IV. Definitions
V. Definitional Conventions
VI. Compliance
VII. Object Extensions
1 Class Definition
Extension of the Type System, Restrictions on Class Definitions,
Contents and Syntax of Class Definitions
(Kinds, Inheritance, Fields, Methods, Constructors, Destructors)
Scope of Entities Defined in a Class, Class Definitions
2 Kinds of Classes
(Concrete, Abstract, Property, Type Model, Views)
3 Inheritance
(The Root Class, Multiple Inheritance, Name Conflicts, Overriding, Abstract Methods, Constructors, and Destructors)
4 Syntax
5 Object Access
The Object Model, Implicit Parameter Self
Polymorphism during Construction and Destruction
Implicit References, Field References, Inherited, Reference Type Coercion
Operations
(Compatibility, Methods Activation, Constructors & Destructors, Assignment, Comparison, Parameter Passing, Membership)
6 Predefined Entities
Null, Copy, Root
(Create, Destroy, Clone, Equal)
TextWritable
(ReadObj and WriteObj)
7 Signatures
8 With Statement
9 Procedure, Function, Constructor, and Destructor Declarations
10 Changes to Export Clause
11 Visibility
12 Extended Pascal Features
13 Suggested Changes to Extended Pascal

Figure 7 Object Pascal report outline.

about the language or bad experiences with a specific compiler. In most cases, these notions were based upon myth rather than fact.

A. Myth 1: C and Pascal Are Basically the Same Language

This comparison has been made quite frequently over the years, yet nothing is further from the truth. From the compiler's perspective, the Pascal architecture is much more straightforward. It adheres to stronger type definitions, making optimizations easier to accomplish. From the programmer's perspective, the nature of Pascal is inherently different from C in convention, syntax, structure, and mindset.

A variety of coding styles can be seen when converting several C source code projects and a few C++ projects to Pascal and Object Pascal. The process of converting a C program to a Pascal program provides first-hand experience in the differences between the two languages. In Pascal, declarations must be moved to the top of a block, and often, a lot of investigation is required to decipher many of the C data structures. During the conversion process, programmers reorganize the code, changing and indenting it to make it more readable, so that the actual translation process is much easier. It is a significant amount of work, however, and in some cases the difficulty is so extreme that it is not worth the trouble.

The point is that while C code can be made readable, it provides loopholes that give programmers the ability to create chaos. Given the weak typing of the C language, inconsistency of data use becomes not only a possibility, but a probability. Pascal's strong typing makes inconsistency of data use far less likely to occur.

Another difference between C and Pascal is the underlying design of the language and the intent of that design. C shares APL's penchant for being able to pack a lot of action into a single expression or line of

code. This provides a certain "cool" factor that many programmers appreciate, but it also leads to a bloated and unreadable code base which can be difficult, at best, to maintain. The extensive operator overloading and weak-typing architecture increases the odds that a programmer will get caught up in the "passion" of the moment, indulge every shortcut, and end up with a maintenance nightmare.

Cultures have sprung up around both languages reflecting different attitudes toward getting work done. The culture of Pascal is oriented unapologetically toward readability in style, elegance of algorithm, and its expression as code. The culture of C is self-consciously ambitious, yet obfuscatory. This is illustrated quite well by C. A. R. Hoare in an introduction to the paper "An Axiomatic Definition of the Programming Language Pascal" for the book, *Great Papers in Computer Science*. In reference to his goal of designing a better programming language—one which makes it easier to write correct programs and harder to write incorrect ones—Hoare writes,

> It is a matter of continuing regret that so few languages have ever been designed to meet that goal, or even to make significant concessions towards it. For example, the programming language C was designed to assist in writing a small single-user operating system (UNIX) for a real-time minicomputer (PDP 11), now thankfully obsolete. For this purpose, its low level of abstraction and plethora of machine-oriented features are entirely appropriate. For all other purposes, they are a nuisance. The successful propagation of the language can be explained by accidental, commercial, historical, and political factors; it is hardly due to any inherent quality as a tool for the reliable creation of sophisticated programs.

B. Myth 2: Pascal Is Limited in Power

The biggest myth about Pascal is that it is a language without power. Nothing can be further from the truth. Pascal is not only a language designed to encourage well-written and manageable code, but it has evolved into a powerful language that supports industrial and commercial needs.

One problem with perception is that many look no further than the original Unextended Pascal standard. While a strong language in itself, it does fall short of supporting more industrial strength needs. Extended Pascal evolved from Pascal to support industrial, scientific, and commercial needs. Further extensions to Pascal were developed by Borland and Apple Computer, providing the language with object-oriented capabilities.

The so-called "limited power" of Pascal may also refer to the lack of low-level and machine-oriented features which are inherent in C. It is true that Pascal's design is not conducive to poking around the machine at a low level, but there are constructs available within the language that support performance of low-level operations. Low-level hacking may not be encouraged by the language, but it is not prevented either. Programmers have access to every allowed memory location via the use of memory pointers. In addition, there are low-level libraries on each platform that support the access and control of memory and devices. Any language can access these libraries as long as the call is properly defined. These low-level libraries are inherently machine dependent anyway, so direct language support is inappropriate.

Finally, some have the misguided idea that C's inherent low-level nature makes executables more efficient than ones generated in Pascal. That, too, is a fallacy. The efficiency of the executable is only as good as the compiler. The truth is that Pascal's architecture lends itself to easy optimization by a compiler. In a recent interview, John Reagan, architect of the Compaq Pascal compiler and member of the X3J9 Pascal Standards Committee, writes, "The strong-typing of Pascal makes it easier for an optimizer to understand the program and provide better generated code. That isn't to say it can't be done for C, but it just takes more work.

"The point is that Pascal can produce efficient code and provide the additional benefit of strong-typing and optional run-time checks. You need not switch from Pascal to C just to get performance (at least on Compaq's OpenVMS or Tru64 UNIX platforms where our compiler runs)."

This last statement makes an important point regarding an efficient code compiler: the performance of a compiler is highly dependent upon the compiler architect. Unfortunately, while Pascal's design makes optimizations easier to implement in a compiler, that does not mean that all compiler architects take advantage of it.

To summarize, while the original Unextended Pascal may have lacked certain functionality, fully supported Extended Pascal provides industrial strength power, and Object Pascal advances the language to support object-oriented programming. Furthermore, the nature of Pascal lends itself to easy optimization by a compiler to produce efficient run-time executables. Ingemar Ragnemalm, noted shareware author and co-author of the book *Tricks of the Mac Game*

Programming Gurus, wrote, "I can do everything in Pascal that can be done in C, but in a more elegant manner."

C. Myth 3: Pascal Has Weak String-Handling Capabilities

The nature of myths makes it ironic that Pascal should be saddled with the same criticism made of C regarding poor string-handling capabilities. Nothing could be easier than working with strings in Pascal. String-handling capabilities are built into the language, using the predefined `STRING` schema type for variable-length strings and `PACKED ARRAY[1..n] OF CHAR` for fixed-length strings.

The elegance of Pascal strings is that they are so simple to use that there is really no need to understand how they are handled internally. A string is assigned using single-quoted text, such as

```
myString := 'Strings are simple in
  Pascal';
```

Checking for null strings is also simple, with both of the following examples working equally as well on variable-length strings:

```
1) if myString = '' then
       myString := 'Go 49ers!';
2) if length(myString) = 0 then
       myString := 'Go 49ers!';
```

For fixed-length strings, the length is always the fixed length regardless of the value, so the first option above would be the way to perform the comparison. The smaller string is padded with blanks to match the length of the fixed string it is being compared with.

Locating a string within a string is done simply with an `INDEX` function:

```
theIndex := index(myString,'9');
```

Using the value set for `myString`, the results of this `INDEX` call would result in a value of 5 for `theIndex`. Pascal's position of characters in a string starts at 1 rather than 0 (i.e., the third character is position 3, the fourth character is position 4, etc.). In some other languages, the first character is position 0, the second is position 1, etc. In this regard, Pascal is more intuitive.

Extended Pascal also supports the "+" operator for string concatenation. Therefore, a string may be constructed as follows:

```
output_String := 'Employee #' +
  emp_no + '(' + emp_name +
  ') prefers to program in ' +
    emp_favorite_language;
```

Extended Pascal also provides support for reading and writing from strings just like one would read and write from a file. For example,

```
Var
  myAge:    string(10);
  age:      integer;

  myAge := '29';
  readStr(myAge, age);
```

The above `READSTR` call would result with a value of 29 in the integer variable `Age`. Likewise,

```
Var
   outputString: string(100);
   empName:      string(30);
   boxes:        integer;

empName := 'Jane Doe';
boxes := 298;
writeStr(outputString,'Employee ',
   empName,' sold ',boxes:1,
   ' boxes of cookies.');
```

This would result with the following value for `outputString`:

```
Employee Jane Doe sold 298 boxes of
cookies.
```

There are also the predefined string functions of `SUBSTR` and `TRIM` and other useful string support, all described in the Extended Pascal standard.

D. Myth 4: Pascal Does Not Support Object-Oriented Programming

Initially designed and released by Apple Computer in 1986, Object Pascal was developed as an extension to Pascal to support object-oriented programming. Object support is incorporated in THINK Pascal, CodeWarrior Pascal, Borland Pascal, and various open source Pascals. As an example, Fig. 8 illustrates Object Pascal under the CodeWarrior Pascal dialect.

E. Myth 5: Pascal Is Only an Instructional Language

There is no argument with the fact that Pascal is an excellent teaching language. Its design encourages good programming habits and supports the creation of com-

```
Program OOP_Sample;

Type
   Employee = object
      firstName:    string;
      lastName:     string;
      hourlyWage:   real;
      Function      Pay(hoursWorked: integer): real;
      end;
   ExemptEmployee = object(Employee)
      Function      Pay(hoursWorked: integer): real; override;
      end;

Var
   anyEmp:    Employee;
   exEmp:     ExemptEmployee;

Function Employee.Pay(hoursWorked: integer): real;

   begin (* Pay with deduction for benefits *)
   Pay := hourlyWage * hoursWorked - 100;
   end;

Function ExemptEmployee.Pay(hoursWorked: integer): real;

   begin (* Pay with no deductions *)
   Pay := hourlyWage * hoursWorked;
   end;

begin
new(exEmp);
exEmp.Hire('John','Smith',15);
writeln('As exempt, pay ',exEmp.lastName,' $',exEmp.pay(40):8:2);
new(anyEmp);
anyEmp.Hire('Jane','Doe',15);
writeln('As a regular employee, pay ', anyEmp.lastName,
    ' $', anyEmp.pay(40):8);
end.
```

Figure 8 Object Pascal sample.

plex data structures from simple, well-defined types. However, it is far more than just an instructional language. Pascal is used in commercial applications as well as in industrial and scientific environments. Most if not all of what a programmer would want to do in C and C++ can be done in Extended Pascal and Object Pascal. There are entire systems built upon Compaq Pascal in industrial manufacturing shops, and there are also several commercial desktop applications which have been written in Pascal. Delphi, a Pascal development system available on Windows, is extended and object oriented and is one of the most popular rapid application development (RAD) systems on the platform. On the Macintosh platform, CodeWarrior Pascal and

1) - InterArchy – *Full featured and award winning FTP client*
2) - Ingemar's Skiing Game – *Action game*
3) - Klondike – *One of the most popular Solitaire games*
4) - Autoshare – *A full feature listserver and auto-responder*
5) - Scripter – *Top selling development environment for Applescripting*
6) - SuperLock – *File security program*
7) - FlightMath – *Flight analysis program*
8) - JacqCAD Master– *CAD program for Jacquard textile design*
9) - NIH Image– *Image Analysis program for Biomedicine*

Figure 9 Commercial-quality Pascal applications.

THINK Pascal are the most popular. Figure 9 is an example of some of the commercial Macintosh applications written in Pascal.

F. Myth 6: Pascal Is Not for Serious Programmers

In 1981, Brian W. Kernighan posted an article on the Web, "Why Pascal Is Not My Favorite Programming Language," that criticized Pascal as not being suitable for real programming tasks and referred to it as a "toy language." One of Kernighan's introductory comments regarding Pascal said, "Because the language is so impotent, it must be extended." The article then exposed the shortcomings of the original Unextended Pascal without mentioning the extensions.

It is relevant to note that the original intent of Pascal was to provide a solid language suitable for teaching programming—one whose implementations could be both reliable and efficient on then-available computers. Unextended Pascal was exactly that language. Early adaptors of Pascal, however, recognized the strengths and promise of the language and began to use it far beyond its original intent. The Extended Pascal standard was created to refine the language, to better support these commercial needs, and to establish Pascal as a language suitable for serious programmers.

The criticisms in Kernighan's article have become outdated and mostly irrelevant with the implementation of Extended Pascal. The article would not have been mentioned at all, except that the criticisms contained within the article are used by many in the field today against current implementations of Pascal.

The following is a summary of those points, as well as the Pascal extensions which invalidate them.

1. "The size of all arrays are part of its type; therefore, it is not possible to write general-purpose routines to deal with strings of different sizes." With the introduction of Extended Pascal, variable-length strings were implemented, along with several other powerful string capabilities (see Item 3. String Capabilities, Section III. B).
2. "The lack of static variables and variable initialization destroy the locality of a program." Variable initialization is included in the Extended Pascal standard (see Item 10. Initial Variable State, Section III. B).
3. "The lack of separate compilation impedes the development of large programs and makes the use of libraries impossible." Modularity and separate compilation is a part of the Extended Pascal standard (see Item 1. Modularity and Separate Compilation, Section III. B).
4. "The order of logical expression evaluation cannot be controlled, which leads to convoluted code and extraneous variables." Short circuit boolean operators are a part of the Extended Pascal standards (see Item 24. Short Circuit Boolean Evaluation, Section III. B).
5. "There is no flow control due to the lack of `RETURN` and `BREAK` statements." Although not in the standards, most current Pascals support function and control loop exits. In both Compaq and CodeWarrior Pascals, `RETURN` will exit a function. In Compaq Pascal, `CONTINUE` and `BREAK` are supported in control loops, and in CodeWarrior Pascal, `CYCLE` and `LEAVE` are supported.
6. "The CASE statement is emasculated because there is no default clause." The Extended Pascal standard now includes an `OTHERWISE` clause in `CASE` statements (see item 14. Case Statement and Variant Record Enhancements, Section III. B).
7. "The language lacks most of the tools needed for assembling large programs, most notably file inclusion." Extended Pascal's modules provide much more sophisticated support for managing and assembling programs than primitive file inclusion methods (see Item 1. Modularity and Separate Compilation, Section III. B). With modules, file inclusion is superfluous.
8. "There is no escape from Pascal's strong typing controls." Although not stated in the standards, type casting is supported in modern Pascal implementations. In Compaq Pascal, `"::"` is the casting operator (e.g., `myVar::myType`). In CodeWarrior Pascal, casting is implemented using parenthesis (e.g., `myType(myVar)`). The language has evolved without sacrificing the tremendous benefits of strong typing (see Section II).

V. PASCAL TODAY

Pascal is still used today, both as a powerful educational tool for programming and as a viable language for industrial, commercial, scientific, shareware, and freeware applications. The Pascal language is available on a number of different platforms, both commercially and through open source.

A. Platforms

Pascal is available for

- OpenVMS VAX
- OpenVMS Alpha
- Tru64 Unix

Table V Available Pascal Compilers

Complier and vendor	Platforms	Supports
Compaq Pascal by Compaq Computer http://www.openvms.compaq.com/commercial/pascal	Open VMS, Tru64 Unix	PAS, full EPAS, nearly full
Prospero Extended Pascal by Prospero Software http://www.prosperosoftware.com/e32iw.html	Windows	PAS, full EPAS, full OP
Delphi by Borland http://www.borland.com/delphi/	Windows	Rapid development[a]
THINK Pascal by Symantec http://www.think-pascal.com/	Macintosh	PAS, full EPAS, partial OP
CodeWarrior Pascal by Metrowerks	Macintosh, Windows	PAS, full EPAS, partial OP
Borland Pascal by Borland	Windows	PAS, nearly full EPAS, partial OP (Borland)
FreePascal http://www.freepascal.org/	Linux, OS/2, FreeBSD, Windows	PAS, nearly full EPAS, partial OP (Borland)
GNU Pascal by Free Software Foundation http://agnes.dida.physik.uni-essen.de/~gnu-pascal/	Linux, OS/2, FreeBSD, Windows	PAS, full EPAS, partial OP (Borland)
Dr. Pascal by Visible Software http://www.visible-software.com.	Windows, Macintosh, Open VMS	PAS, Partial EPAS

[a]Borland's Delphi is a RAD environment. It is based upon Pascal, but uses menus and windows for application building, hiding the code for the most part.

- Windows 95/98/ME
- Windows NT
- Linux
- FreeBSD Unix
- OS/2
- Macintosh

See Table V for a list of available compilers.

B. Compilers

Commercial versions of Pascal include Borland Pascal (aka, Turbo Pascal), Delphi, Compaq Pascal, THINK Pascal, and CodeWarrior Pascal. Turbo Pascal and THINK Pascal have not been updated for quite some time; likewise, CodeWarrior Pascal's final update is in progress at the time of this writing. Delphi and Compaq Pascal are both still commercially viable products, with yearly updates and full vendor support. Finally, FreePascal and GNU Pascal are ongoing open source projects, providing versions of Pascal freely available to the public.

Table V provides a list of the more popular Pascal compilers available. In the "Supports" column, each compiler is rated for how well it adheres to the unextended Pascal standard (PS) and the Extended Pascal standard (EPS), and for whether it supports the Object Pascal model (OP). For each standard, the compiler is also rated to what extent it supports that standard: full, nearly full, or partial. In addition to the Object Pascal as defined in the Object Pascal technical report, Borland designed a variation of Object Pascal. Compilers which adhere to Borland's Object Pascal model are designated as such.

C. Internet

There is a number of Internet resources which provide information, tools, and support for Pascal programmers. Table VI provides a few of these resources. For a more complete list, visit http://pascal-central.com/plinks.html, the Pascal Links and Forums Web page, at Pascal Central.

In addition to these Web sites, there are Pascal mailing lists that can be useful to both novice and ex-

Table VI Pascal Web Sites

Web site	Description
Learn Pascal http://www.taoyue.com/tutorials/pascal/	A very complete and comprehensive Pascal tutorial. The tutorial is searchable and serves as a useful Pascal reference, even after one has completed it.
Pascal Central http://pascal-central.com/	Provides the Pascal community one place to obtain Pascal technical information, Pascal source code, and Pascal-related Internet links.
Prospero Pascal http://www.prosperosoftware.com/e32iw/html	Prospero fully supports both Pascal standards and the Pascal Committee's object-oriented extensions.
Introduction to Pascal http://www.cit.ac.nz/smac/pascal/	Courses to help teach Pascal programming.
FreePascal http://www.freepascal.org/	Open Source Freeware Pascal Compiler supporting a variety of platforms.
GNU Pascal http://agnes.dida.physik.uni-essen.de/~gnu-pascal/	Open Source Freeware Pascal Compiler supporting a variety of platforms.
MacTech Macintosh Pascal http://www.mactech.com/macintosh-pascal/	MacTech hosts the Macintosh Pascal Hobbyist Guide page.
THINK Pascal Guide http://www.think-pascal.com/	Ingemar's Guide to Think Pascal, a free Mac development environment.
Ingemar's Corner http://pascal-central.com/ingemars-corner.html	A descriptive list of very useful Pascal sample code at Ingemar's FTP site
How to Code Pascal http://www.allegro.com/papers/htpp.html	A paper focused on how to write quality Pascal code.
Khaan's Place http://www.algonet.se/~khaan/	A site devoted to Borland Pascal Programming, where the programmer can learn Pascal and find lots of source code.
Pascal Tools http://pascal-central.com/tools.html	Free Pascal tools for Macintosh, Windows, and Linux.

pert Pascal programmers. Use the mailing lists to ask questions, trouble-shoot problems with code, and share expertise. A list of these Pascal mailing lists, including instructions on how to subscribe, can be found at the Web site http://pascal-central.com/plinks.html#anchor-lists

VI. SUMMARY

Pascal has evolved since its origin in the 1970s. A full demonstration of just how much the language has evolved would require a study of the Extended Pascal standard, the Object Pascal report, and the Compaq Pascal Language Reference manual. My own work in Pascal has spanned the last 15 years, and I've witnessed the evolution of the language first hand.

While the evolved Pascal goes far beyond what Wirth originally put forth, it does not sacrifice the original design concept: an elegant programming language that makes it easier to write correct programs and harder to write incorrect ones.

Programmers should avoid seduction by languages that offer a lot of power while sacrificing readable and manageable code. Extended Pascal provides all the features needed to perform any task, while enforcing enough rules to establish a manageable code base. With its strict type casting and block-structured architecture, Pascal allows programmers to make the same assumptions that the compiler makes about program flow and data usage. It eliminates much of the guesswork from interpreting data structures, and it provides a robust architecture for breaking down programs into a hierarchy of tasks. When a programmer needs to establish a strong and manageable code base in a timely manner, the Extended Pascal language is one of the best tools to use.

SEE ALSO THE FOLLOWING ARTICLES

C and Ctt • COBOL • Compilers • FORTRAN • Java • Object-Oriented Programming • Programming Languages Classification • Visual Basic

BIBLIOGRAPHY

Clancy, M. (1996). "Designing Pascal Solutions: Case Studies Using Data Structures."
Cooper, D. (1987). "Condensed Pascal."
Cooper, D., and Clancy, M. (1993). "Oh! Pascal!"
Digital Equipment Corporation. (1993). "DEC Pascal Language Reference Manual."
Drake, R., and Catambay, B. (2000). "Pascal From the Quality Perspective."
Findlay, W., and Watt, D. (1978). "Pascal: An Introduction to Methodical Programming."
Hoare, C. A. R., and Wirth, N. (1973). An axiomatic definition of the programming language Pascal. *Acta Informatica,* Vol. 2, 335–355.
Horowitz, E. (1999). "Fundamentals of Data Structures in Pascal."
ISO. (1990). ISO 7185: Unextended Pascal Standards.
ISO. (1990). ISO 10206: Extended Pascal Standards.
Koffman, E. (1997). "Blaise Pascal: Reasons of the Heart."
Laplante, P. (1996). "Great Papers in Computer Science."
Reagan, J. (1995). "Pascal Standards FAQ."
Reagan, J., and Catambay, B. (2000). "Interview With a Pascal Architect."
Schneider, G. M., Weingart, S. W., and Perlman, D. M. (1982). "An Introduction to Programming and Problem Solving with Pascal."
Technical Committee X3J9 (1993). "Object-Oriented Extensions to Pascal."

Pattern Recognition

Sergios Theodoridis
University of Athens

GLOSSARY

classifier The part in a pattern recognition system whose goal is to realize the discriminant functions and place a pattern, with unknown class label, into one of the regions into which the feature space has been divided by the discriminant functions. The pattern is then assigned to the class associated with the region. The design of the classifier is based on the training data set.

decision surfaces The surfaces in the feature space separating contiguous regions of different classes. The decision surfaces are realized by the discriminant functions.

discriminant functions Functions whose goal is to divide the space of feature vectors into regions, each region being associated with a single class.

features Measured quantities, which characterize each pattern, and used for the classification of the patterns into one of the classes. A number of features together form the feature vector.

generalization The capability of a classification system, designed using training patterns of known class labels, to perform well with patterns of unknown class labels.

pattern recognition Deals with the automatic classification of an object (pattern) into one from a number of different categories or classes.

supervised pattern recognition The pattern recognition task where the number of classes is known *a priori* and a set of training patterns is available; that is, patterns with known class labels.

unsupervised pattern recognition or clustering The Pattern Recognition task where the number of classes is not known *a priori* and the goal is to unravel the underlying similarities among the data and group (cluster) "similar" patterns together.

I. INTRODUCTION

Pattern recognition (PR) is a fast growing field with applications in many diverse areas such as optical character recognition (OCR), computer-aided medical diagnosis, speech recognition, computer vision, and data mining, to name but a few. A PR system is an integral part in most *machine intelligence systems* built for decision making.

Pattern recognition deals with the classification of a given object (*pattern*) into one from a number of different *categories or classes*. The goal of PR is the development of an *automatic classification system,* which will be subsequently used for the classification of patterns presented to it.

In order to illustrate the basic philosophy behind the development of a classification system, let us consider a

computer-aided diagnosis system for, say, X-ray images. The purpose of such a system is to classify a "suspect" region in an image (the pattern in this case) in either of two classes, depending on whether this image region corresponds to a benign or malignant lesion (cancer). Given the specific region of the image as input, the classification system will measure a number, say l, of prespecified quantities, x_i, $i = 1,2, \ldots, l$, related to the image region (such as its perimeter, its area, etc.), known as *features*. In the sequel, each region is represented by the vector formed by the above features, i.e., $\mathbf{x} = [x_1, x_2, \ldots, x_l]^T$ which is called *feature vector*. Each pattern is represented by a single feature vector which is then used for the classification of the corresponding image region either to the "benign" or to the "malignant" class.

The vector space into which the feature vectors lie is known as *feature space*. The dimension of the space depends on the number of the selected features, l, and it can be either continuous or discrete valued. Thus, if in an application the features used are discrete valued (i.e., they may take values only in, usually, a finite data set C), the feature space is C^l. If, on the other hand, the features can take values in a continuous real number interval, the feature space is R^l. In the case where the first l_1 features are continuous valued and the remaining $l - l_1$ are discrete valued, taking values in C, the feature space is $R^{l_1} \times C^{l-l_1}$. In the sequel, we will assume that the feature vector consists of real valued coordinates and the feature space is R^l.

Having given the basic definitions, let us now focus on the major design stages of a classification system.

- *Feature generation:* This stage deals with the generation of the features that will be used to represent the patterns. The choice is application dependent and they are usually chosen in cooperation with an expert in the field of application.
- *Feature selection:* This stage deals with the selection of features, from a larger number of generated ones, that are most representative for the problem. That is, they are rich in information concerning the classification task. Basically, their values must exhibit large variation as we move from one class to another. Furthermore, the selection of features must be done in a way that guarantees minimum information redundancy. Feature selection is an important task. If the selected features are poor in classification related information, the overall performance of the system will be poor. If, on the other hand, information-rich features are mutually highly correlated the system complexity is increased without much gain in performance.
- *Classifier design:* This stage can be considered as the heart of the classification system. The design of the classifier is usually carried out through the optimization of an optimality criterion. Speaking in geometrical terms, the task of a classifier is to divide the feature space into regions so that each one of them corresponds to only one of the posible classes. Thus, for a given feature vector, the classifier identifies the region, in the feature space, where it belongs and assigns the vector to the corresponding class. The design of the classifier is based on a set of features vectors, which are known to which class they belong, and it is known as the *training data set*.
- *System evaluation:* The final stage consists on the performance evaluation of the designed classification system, that is, the estimation of the *classification error probability*. This is also an important task, since in the majority of cases this estimation is based on a limited number of test data. If the estimated error probability turns to be higher than a prespecified threshold, the designer may have to redesign some or all of the previous stages.

Figure 1 shows the various stages followed for the design of a classification system. As it is apparent from the feedback arrows, these stages are not independent. On the contrary, they are interelated and, depending on the results, one may come back to redesign earlier stages in order to improve the overall performance. Furthermore, there are some methods which combine stages, i.e., the feature selection and the classifier design stage, under a common optimization task.

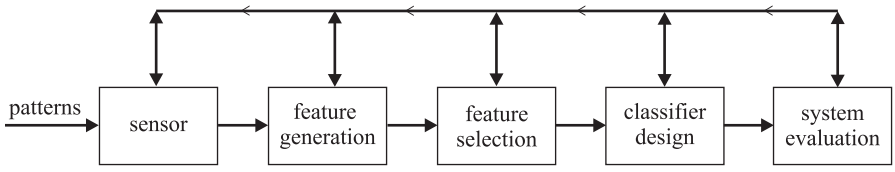


Figure 1 The basic stages of a PR system.

In our discussion, so far, we have assumed that a set of training data was available, and the classifier was designed by exploiting this *a priori* known information. This is known as *supervised pattern recognition*. However, this is not always the case, and there is another type of PR tasks for which training data of known class labels are not available. In this type of problem, we are given a set of feature vectors X and the goal is to unravel the underlying *similarities and cluster* (group) "similar" vectors together. This is known as *unsupervised pattern recognition* or *clustering*. Such tasks arise in many applications in social sciences and engineering, such as remote sensing, image segmentation, image and speech coding, etc.

In the sequel, we will first focus on the supervised PR task and consider the various design stages. Sections II-X are related to the classifier design stage. Feature generation and feature selection is treated in Section XI and the system evaluation stage in Section XII. Unsupervised PR is treated in Section XIII.

II. BAYES DECISION THEORY

Consider an M-class classification task, i.e., a problem where an object may belong to one out of M possible classes. Let ω_i denote the ith class. The problem that has to be solved by the classifier may be stated as: "Given a feature vector **x,** which is the most appropriate class to assign it to?" The most reasonable solution is to assign it to the *most probable* class. Here is the point where probability theory enters into the scene. In this context, the feature vector **x** will be treated as a random vector, i.e., a vector whose coordinates (features) are treated as random variables.

Let $P(\omega_i|\mathbf{x})$ be the probability that **x** stems from the ω_i class. This is also known as *a posteriori probability*. In mathematical terms the notion of "most probable" is stated as:

Assign **x** *to class* ω_i *if*

$$P(\omega_i|\mathbf{x}) = \max_{j=1,\ldots,M} P(\omega_j|\mathbf{x}) \tag{1}$$

This is the well known *Bayes classification rule.*[1]

Let $P(\omega_i)$ be the *a priori* probability for class ω_i, i.e., the probability a pattern to belong class ω_i, and $p(\mathbf{x}|\omega_i)$ be the class conditional probability density function (pdf) for class ω_i, which describes the distribution of the feature vectors in the ith class, $i = 1, \ldots, M$. Finally, let $p(\mathbf{x})$ denote the pdf of **x**. Recalling that

$$P(\omega_i|\mathbf{x}) = \frac{P(\omega_i)\,p(\mathbf{x}|\omega_i)}{p(\mathbf{x})} \tag{2}$$

the Bayes classification rule may be stated as

Assign **x** *to class* ω_i *if*

$$P(\omega_i)\,p(\mathbf{x}|\omega_i) = \max_{j=1,\ldots,M} P(\omega_j)\,p(\mathbf{x}|\omega_j) \tag{3}$$

$p(\mathbf{x})$ has been omitted since it is the same for all classes, i.e., independent of ω_i.

In the special case of equiprobable classes, i.e., if $P(\omega_j) = 1/M$, $j = 1, \ldots, M$, the Bayes rule can simply be stated as

Assign **x** *to class* ω_i *if*

$$p(\mathbf{x}|\omega_i) = \max_{j=1,\ldots,M} p(\mathbf{x}|\omega_j) \tag{4}$$

That is, in this special case, the Bayes decision rule rests on the values of the class conditional probability densities evaluated at **x**.

Let us consider now Fig. 2, where the class conditional pdfs of a one-dimensional two-class problem are depicted. The classes are assumed equiprobable. The point x_0 is the threshold that partitions the feature space into two regions. Thus, according to the third version of the Bayes decision rule, all x that belong to R_1 are assigned to class ω_1, and all $x \in R_2$ are assigned to class ω_2.

This example, also, indicates that classification errors are unavoidable. Speaking in terms of Fig. 2, there is a finite probability for some x to stem from ω_2 and lie in R_1. However, the classifier will assign it to class ω_1. It may be shown that *the Bayes decision rule*

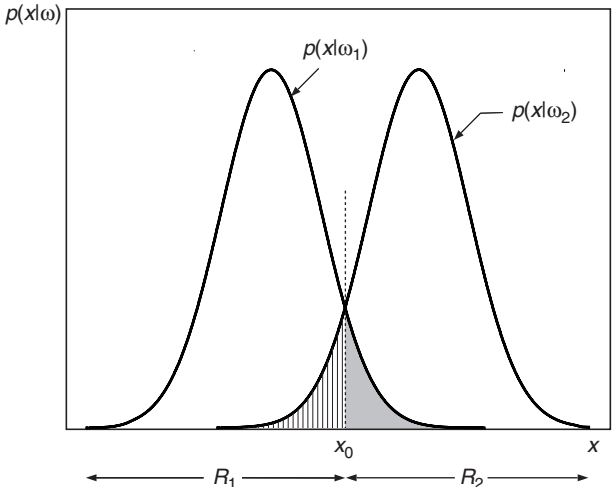


Figure 2 The class conditional pdfs of a one-dimensional two-class problem. Region R_1 is assigned to class ω_1, and R_2 to class ω_2.

[1]In the case where two classes satisfy Eq. (1) we may choose arbitrarily one of the classes.

minimizes the total probability of error. Thus, the Bayesian classifier is the optimal one, since minimizing the error probability is the ultimate goal of any classification system designer.

III. DISCRIMINANT FUNCTIONS AND DECISION SURFACES

In some cases, instead of working directly with $P(\omega_i|\mathbf{x})$ it is more convenient to use $g_i(\mathbf{x}) = f(P(\omega_i|\mathbf{x}))$, where $f(\cdot)$ is a monotonically increasing function. The $g_i(\cdot)$s are known as *discriminant functions*. In this case, the Bayes decision rule becomes

Assign $\mathbf{x}$ *to class* ω_i *if*

$$g_i(\mathbf{x}) = \max_{j=1,\ldots,M} g_j(\mathbf{x}) \tag{5}$$

Monotonicity does not change the points where maxima occur, and the resulting partition of the feature space remains the same. The contiguous regions in the feature space, R^l, that correspond to different classes are separated by continuous surfaces, known as *decision surfaces*. If regions R_i and R_j, associated with classes ω_i and ω_j, respectively, are contiguous then, it is easy to see from the respective definitions that, the corresponding decision surface is described by:

$$g_{ij}(\mathbf{x}) = g_i(\mathbf{x}) - g_j(\mathbf{x}) = 0 \tag{6}$$

So far, we have approached the classification problem via Bayesian probabilistic arguments and the goal was to minimize the classification error probability. However, as we will soon see, not all problems are well suited for such an approach. For example, in many cases the involved pdfs are complicated and their estimation is not an easy task. In these cases it may be more preferable to compute decision surfaces *directly by means of alternative costs* and this will be our focus in Sections V–IX. Such approaches give rise to discriminant functions and decision surfaces, which are entities with no (necessary) relation to Bayesian classification and they are, in general, suboptimal with respect to Bayesian classifiers.

In the sequel, we will focus on a particular family of decision surfaces associated with the Bayesian classification, for the specific case of Gaussian density functions $p(\mathbf{x}|\omega_i)$, i.e.,

$$p(\mathbf{x}|\omega_i) \equiv \mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i) = \frac{1}{(2\pi)^{l/2}|\Sigma_i|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\Sigma_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right) \tag{7}$$

$i = 1,2,\ldots,M$, where $\boldsymbol{\mu}_i = E[\mathbf{x}|\mathbf{x} \in \omega_i]$ is the mean vector for the class ω_i, $\Sigma_i = E[(\mathbf{x}-\boldsymbol{\mu}_i)(\mathbf{x}-\boldsymbol{\mu}_i)^T]$ is the $l \times l$ covariance matrix for class ω_i and $|\Sigma_i|$ is the determinant of Σ_i. It is clear that only the vectors $\mathbf{x} \in \omega_i$ contribute in Σ_i. For the special one dimensional case, $l = 1$, the above becomes our familiar Gaussian pdf, i.e.,

$$p(x|\omega_i) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

where σ^2 is the variance around the mean μ. Our objective is to design a Bayesian classifier taking into account that each $p(\mathbf{x}|\omega_i)$ is a normal distribution. Having in mind the Bayesian rule given in Eq. (3), we define the discriminant functions

$$g_i(\mathbf{x}) = \ln(P(\omega_i)p(\mathbf{x}|\omega_i)) \tag{8}$$

Substituting Eq. (7) into Eq. (8) we obtain

$$g_i(x) = -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\Sigma_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) + \ln P(\omega_i) + c_i \tag{9}$$

or, after some manipulations,

$$g_i(x) = -\frac{1}{2}\mathbf{x}^T\Sigma_i^{-1}\mathbf{x} + \frac{1}{2}\mathbf{x}^T\Sigma_i^{-1}\boldsymbol{\mu}_i - \frac{1}{2}\boldsymbol{\mu}_i^T\Sigma_i^{-1}\boldsymbol{\mu}_i + \frac{1}{2}\boldsymbol{\mu}_i^T\Sigma_i^{-1}\boldsymbol{x} + ln\ \mathrm{P}(\omega_i) + \mathrm{c}_i \tag{10}$$

where c_i is a constant equal to $-\frac{l}{2}\ln(2\pi) - \frac{1}{2}\ln|\Sigma_i|$.

Clearly, the decision surfaces defined as $g_i(x) - g_j(\mathbf{x}) = 0$ are *quadrics* (hyperellipsoids, hyperparaboloids, pairs of hyperplanes, etc.) and the resulting Bayesian classifier is a *quadric classifier*.

The special cases of linear classifiers:

1. Assuming that the covariance matrices for all classes are equal to each other, i.e., $\Sigma_i = \Sigma$, $i = 1,2,\ldots,M$, the terms $-\frac{1}{2}\mathbf{x}^T\Sigma_i^{-1}\mathbf{x}$ and c_i are the same in all g_i's for all classes, thus they can be omitted. In this case, $g_i(\mathbf{x})$ becomes a linear function of $\mathbf{x}$, and it can be rewritten as:

$$g_i(\mathbf{x}) = \mathbf{w}_i^T\mathbf{x} + w_{i0}, \tag{11}$$

where

$$\mathbf{w}_i = \Sigma^{-1}\boldsymbol{\mu}_i \tag{12}$$

and

$$w_{i0} = -\frac{1}{2}\boldsymbol{\mu}_i^T\Sigma^{-1}\boldsymbol{\mu}_i + \ln P(\omega_i). \tag{13}$$

In this case, the decision surfaces $g_i(\mathbf{x}) - g_j(\mathbf{x}) = 0$ are hyperplanes and the Bayesian classifier is a *linear classifier*.

2. Assume, in addition, that all classes are equiprobable, i.e., $P(\omega_i) = 1/M$, $i = 1,2,\ldots,M$. Then Eq. (9) becomes

$$g_i(\mathbf{x}) = -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^T\Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu}_i) \equiv -\frac{1}{2}d_m^2 \quad (14)$$

The quantity d_m on the right-hand side of the above equation, without the minus sign, is called *Mahalanobis distance*. Thus, in this case, instead of searching for the class with the maximum $g_i(\mathbf{x})$, we can equivalently search for the class for which the Mahalanobis distance between the respective mean vector and the input vector $\mathbf{x}$ is minimum.

3. In addition to the above assumptions, assume that $\Sigma = \sigma^2 I$, where I is the $l \times l$ identity matrix. In this case, Eq. (14) becomes

$$g_i(\mathbf{x}) = -\frac{1}{2\sigma^2}(\mathbf{x} - \boldsymbol{\mu}_i)^T(\mathbf{x} - \boldsymbol{\mu}_i), \quad (15)$$

or, eliminating the factor $\frac{1}{2\sigma^2}$, which is common for all g_i's, we obtain:

$$g_i(\mathbf{x}) = -(\mathbf{x} - \boldsymbol{\mu}_i)^T(\mathbf{x} - \boldsymbol{\mu}_i) \equiv -d_\epsilon^2 \quad (16)$$

Clearly, searching for the class with the maximum $g_i(\mathbf{x})$ is equivalent to searching for the class for which the Euclidean distance, d_ϵ, between the respective mean vector and the input vector $\mathbf{x}$ becomes minimum.

In summary, in the case where each $p(\mathbf{x}|\omega_i)$ is a normal distribution, $g_i(\mathbf{x})$'s are quadratic functions of $\mathbf{x}$ and the Bayesian classifier partitions the feature space via decision surfaces which are quadrics. If, in addition, all the classes have equal covariance matrices, $g_i(\mathbf{x})$'s become linear functions of $\mathbf{x}$ and the decision surfaces are hyperplanes. In this latter case, the Bayesian classifier becomes very simple. *A pattern (represented by* $\mathbf{x}$*) is assigned to the class whose mean vector* $\boldsymbol{\mu}_i$ *is closest to* $\mathbf{x}$. The distance measure, in the feature space, can be either the Mahalanobis or Euclidean distance, depending on the form of the covariance matrix.

IV. ESTIMATION OF UNKNOWN PROBABILITY DENSITY FUNCTIONS

A. Parzen Windows

As we saw above, the Bayes decision rule requires the knowledge of $p(\mathbf{x}|\omega_i)$'s and $P(\omega_i)$'s, $i = 1,2,\ldots,M$. However, in practice, this is rarely the case. In the majority of the cases, all we have at our disposal is a finite training set of N feature vectors, i.e.,

$$X = \{(\mathbf{x}_i, y_i), \mathbf{x}_i \in R^l, y_i \in \{1,2,\ldots,M\},\ i = 1,2,\ldots,N\}, \quad (17)$$

where y_i is the index indicating the class to which $\mathbf{x}_i$ belongs.

A commonly used estimate for $P(\omega_i)$ is

$$P(\omega_i) = \frac{n_i}{N}, \quad (18)$$

where n_i is the number of vectors in the training data set that belong to class ω_i.

However, the estimation of $p(\mathbf{x}|\omega_i)$ is not so straightforward. One way to attack the problem of estimating $p(\mathbf{x}|\omega_i)$ is to assume that it has a given parametric form and to adopt certain optimality criteria, whose optimization determines the unknown parameters of the distribution, using only the feature vectors available for class ω_i. If, for example, we assume that $p(\mathbf{x}|\omega_i)$ is a normal distribution, the unknown parameters that have to be estimated, may be the mean vector and/or the covariance matrix.

Parametric methods that follow this philosophy are the Maximum Likelihood Parameter Estimation, the Maximum *A Posteriori* Probability Estimation, the Bayesian Inference, and the Mixture models.

An alternative philosophy for estimating $p(\mathbf{x}|\omega_i)$ requires no assumptions of parametric models for $p(\mathbf{x}|\omega_i)$. The basic idea relies on the approximation of the unknown pdf using the histogram method. Let us consider the one-dimensional case ($l = 1$). In this case the data space is divided into intervals (bins) of size h. Then, the probability of x lying in a specific bin is approximated by the frequency ratio, i.e.,

$$P \simeq \frac{k_N}{N} \quad (19)$$

where k_N is the number of data points lying in this bin. As $N \to +\infty$, the frequency ratio converges to the true P. The corresponding pdf value is assumed to be constant throughout the bin and is approximated by

$$\hat{p}(x) = \frac{1}{h}\frac{k_N}{N},\ |x - \hat{x}| \le \frac{h}{2} \quad (20)$$

where $\hat{x}$ is the center of the bin, see Fig. 3.

Generalizing to the l dimensional case, the bins become l dimensional hypercubes of edge h and $\hat{p}(x)$ becomes

$$\hat{p}(x) = \frac{1}{h^l}\frac{k_N}{N} \quad (21)$$

where now k_N is the number of feature vectors lying in the l dimensional hypercube where $\mathbf{x}$ belongs, and h^l is the hypercube volume. In practice, we can make $\mathbf{x}$ the center of the hypercube and measure how many of the training points fall inside it. The above can be

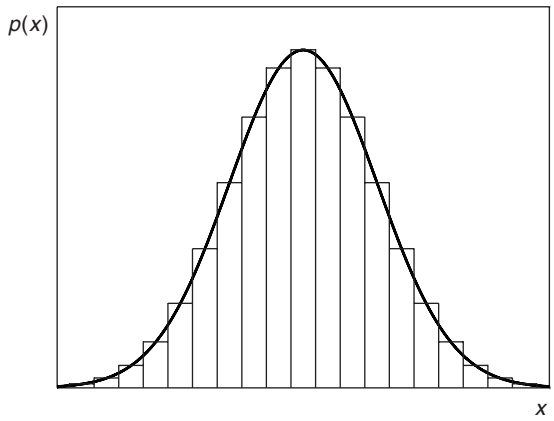


Figure 3 Probability density function approximation by the histogram method.

written in a more elegant form that leads to useful generalizations.

Let us define $\phi(\boldsymbol{x}_i)$ as

$$\phi(\mathbf{x}_i) = \begin{cases} 1, & \text{if } |\mathbf{x}_{ij}| < \frac{1}{2}, \text{ for } j = 1, \ldots, l \\ 0, & \text{otherwise} \end{cases} \quad (22)$$

where x_{ij} is the jth coordinate of the ith vector. In words, $\phi(\mathbf{x}_i)$ equals to 1 if $\mathbf{x}_i$ lies in the l-dimensional unit hypercube centered at the origin $\mathcal{O}$ and 0 otherwise. Then, Eq. (21) may be rewritten in the following more general form

$$\hat{p}(\mathbf{x}) = \frac{1}{h^l}\left(\frac{1}{N}\sum_{i=1}^{N}\phi\left(\frac{\mathbf{x}i - \mathbf{x}}{h}\right)\right) \quad (23)$$

The above can be seen as an effort to expand the unknown $p(\mathbf{x})$ into a set of N functions centered around the training points. However, we try to approximate a continuous function ($p(\mathbf{x})$) in terms of a linear combination of instances of a discontinuous one ($\phi(\cdot)$). Parzen generalized Eq. (23) by using smooth functions ϕ that satisfy

$$\phi(\mathbf{x}) \geq 0,$$

and

$$\int_{\mathbf{x}} \phi(\mathbf{x})\, d\mathbf{x} = 1, \quad (24)$$

and showed that such choices of ϕ lead to legitimate estimates of pdf. A widely used form of ϕ is the Gaussian.

If we employ the Parzen windows method for estimating pdfs, the Bayesian classifier, Eq. (3), takes the form (for the simple two class case)

$$\text{assign } \mathbf{x} \text{ to } \omega_1 \ (\omega_2) \text{ if } \left(\frac{\frac{1}{N_1 h^l}\sum_{i=1}^{N_1}\phi\left(\frac{\mathbf{x}_i - \mathbf{x}}{h}\right)}{\frac{1}{N_2 h^l}\sum_{i=1}^{N_2}\phi\left(\frac{\mathbf{x}_i - \mathbf{x}}{h}\right)}\right) > (<)\ \frac{P(\omega_2)}{P(\omega_1)} \quad (25)$$

where N_1, N_2 are the training vectors in class ω_1, ω_2 respectively. For large N_1, N_2 the above computation is a very demanding job, in both processing time and memory requirements.

B. k Nearest Neighbor Density Estimation

In the Parzen method for the estimation of the pdf, the volume around the points $\mathbf{x}$ was considered fixed (h^l) and the number of points k_N, falling inside the volume, was left to vary randomly from point to point, depending on the local density. Here we will reverse the roles. The number of points $k_N = k$ will be fixed and the size of the volume around $\mathbf{x}$ will be adjusted each time, to include k points. Thus, *in low density areas the volume will be large and in high density areas it will be small.* We can also consider more general types of regions, besides the hypercube, e.g., hyperspheres. The estimator can now be written as

$$\hat{p}(\mathbf{x}) = \frac{k}{NV(\mathbf{x})} \quad (26)$$

where the dependence of the volume $V(\mathbf{x})$ on $\mathbf{x}$ is explicitely shown.

From a practical point of view, at the reception of an unknown feature vector $\mathbf{x}$, we compute its distance d, for example Euclidean, from *all* the training vectors of the various classes, i.e., ω_1, ω_2. Let r_1 be the radius of the hypersphere, centered at $\mathbf{x}$, which contains k points from ω_1 and r_2 the corresponding radius of the hypershare containing k points from class ω_2 (k may not necessarily be the same for all classes). If we denote by V_1, V_2 the respective hypersphere volumes, the Bayesian rule becomes

$$\text{assign } \mathbf{x} \text{ to } \omega_1 \ (\omega_2) \text{ if } \frac{kN_1V_1}{kN_2V_2} > (<)\ \frac{P(\omega_2)}{P(\omega_1)}$$

$$\frac{V_2}{V_1} > (<)\ \frac{N_1}{N_2}\,\frac{P(\omega_2)}{P(\omega_1)} \quad (27)$$

V. NEAREST NEIGHBOR RULES

So far, our kick off point was the Bayesian classifier. A different kind of rule that is suboptimal, in the sense

that it leads to classification error P_e greater than that of the optimal Bayesian classifier, are the k-nearest neighbor (k-NN) rules. The general idea is the following: for a given vector $\mathbf{x}$, we determine the subset A, of the k closest neighbors of $\mathbf{x}$ among the feature vectors of the training set, irrespective of the class where they belong. Let $k_j(A)$, $j = 1, 2, \ldots, M$, be the number of feature vectors in A that belong to class ω_j. We determine, among the $k_j(A)$'s the largest one, say $k_i(A)$, and we assign $\mathbf{x}$ to class ω_i. The Euclidean is the most frequently used distance between vectors, although other distances may also be used. Note that, although this method has an affinity with the k-nearest neighbor pdf estimation, it is a different procedure and its philosophy is different from the Bayesian one.

A popular rule that belongs to this category is the nearest neighbor (NN) rule, which results from the above general scheme for $k = 1$. According to this rule, a vector $\mathbf{x}$ is assigned to the class where its closest vector in the training set X belongs! Although this rule is so simple, it can be shown that its corresponding classification error probability can never be greater than twice the probability of error of the Bayesian classifier, P_B. Upper bounds for the probability of error are, also, available for the general case where $k > 1$. It is worth pointing out that as $k \to +\infty$, the probability of error of the k-NN classifier approaches the optimal error, P_B, of the Bayesian classifier.

Note that all the above schemes require a number of operations that increase with N, which in cases where large data sets are to be utilized require excessive computational effort. One way to avoid this problem, at the cost of loss of information, is to cluster feature vectors of the same class that are close to each other and to use the mean vectors of the resulted clusters as data set. Such an approach reduces significantly the size of the data set.

VI. LINEAR CLASSIFIERS

In this and subsequent sections, we will emancipate from the notion of probability densities and our goal will be to design classifiers that partition the feature space into class regions, via appropriately chosen discriminant functions. These do not have, necessarily, any relation to probability densities and the Bayesian rule. The aim is to estimate a set of unknown parameters of the discriminant functions in a way that places the equivalent decision surfaces optimally in the feature space, according to an optimality criterion. The simple two-class classification task will be considered.

Our first priority, in this section, will be the design of linear classifiers, that is, classifiers that implement linear decision surfaces, i.e., hyperplanes. A hyperplane is described as

$$g(\mathbf{x}) = \sum_{i=1}^{l} w_i x_i + w_0 = \mathbf{w}^T\mathbf{x} + w_0 = 0 \qquad (28)$$

where $\mathbf{w} = [w_1, w_2, \ldots, w_l]^T$ and w_0 are the parameters describing the hyperplane. Note that $(\mathbf{w}, w_0)$ define uniquely a single hyperplane. For the compactness of notation, we can embed w_0 into $\mathbf{w}$ and we augment the feature vector $\mathbf{x}$ with one more coordinate which is equal to 1, i.e., $\boldsymbol{w} = [w_0, w_1, w_2, \ldots, w_l]^T$ and $\mathbf{x} = [1, x_1, x_2, \ldots, x_l]^T$. Thus, the last equation becomes

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = 0 \qquad (29)$$

which is valid for all points lying on the hyperplane. As we know from our analytic geometry basics, all points lying on one side of the hyperplane correspond to $g(\mathbf{x}) > 0$ and all the points lying on the other side correspond to $g(\boldsymbol{x}) < 0$.

A. The Perceptron

Our goal now is to determine the optimal hyperplane that separates the two classes. We will assume that the training feature vectors in the two classes (denoted by $+1$ and -1) are linearly separable, i.e., there exists a hyperplane $\mathbf{w}^*$ that leaves all feature vectors of X that belong to class 1 (-1) in its positive (negative) halfspace. Let us now define

$$\delta_{\mathbf{x}} = \begin{cases} 1, & \text{if } \mathbf{x} \in \omega_2 \\ -1, & \text{if } \mathbf{x} \in \omega_1 \end{cases} \qquad (30)$$

Then a vector $\mathbf{x}$ is *misclassified (correctly classified)* by the hyperplane $\mathbf{w}$ if

$$\delta_{\mathbf{x}}(\mathbf{w}^T\mathbf{x}) > (<)0 \qquad (31)$$

An algorithm that converges to the parameters defining a hyperplane that classifies correctly all feature vectors of X, is the *perceptron algorithm.* This may be viewed as an optimization task as described below.

Let us define the following optimization function

$$J(\mathbf{w}) = \sum_{\mathbf{x} \in \mathcal{Y}_{\mathbf{w}}} \delta_{\mathbf{x}}(\mathbf{w}^T\mathbf{x}) \qquad (32)$$

where $\mathcal{Y}_{\mathbf{w}}$ is the set of all misclassified feature vectors of X, if the hyperplane used is $\mathbf{w}$. Clearly, $J(\mathbf{w})$ is a non-negative function taking its minimum value when

$\mathcal{Y}_{\mathbf{w}} = \varnothing$ (empty set).[2] An iterative algorithmic scheme that minimizes the above cost function is the perceptron algorithm and is given by

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \rho_t \sum_{\mathbf{x} \in \mathcal{Y}_{\mathbf{w}(t)}} \delta_{\mathbf{x}} \mathbf{x} \qquad (33)$$

where t denotes the current iteration step. Typical choices for the weight sequence, ρ_t, are $\rho_t = c/t$, where c is a constant or $\rho_t = \rho < 2$ (to guarantee convergence).

It can be shown that *if the training feature vectors of the two classes are linearly separable, then the perceptron algorithm converges in a finite number of steps to a solution* $\mathbf{w}^*$ *that separates perfectly these vectors.* Note that such a hyperplane is not unique. However, if the two classes are not linearly separable then the perceptron algorithm *does not converge.* In this latter case, a variant of the perceptron algorithm exists that converges (under certain conditions) to an optimal solution (i.e., to a solution with the minimum possible number of misclassified feature vectors) and it is known as the *pocket algorithm.*

Once the hyperplane parameters have been computed, the classifier of Fig. 4 results. The parameters $w_0, w_1, \ldots, w_l$ are those obtained from the perceptron algorithm and $f(z)$ is the *hard limiter function defined* as

$$f(z) = \begin{cases} +1, & \text{if } z > 0 \\ -1, & \text{if } z < 0 \end{cases} \qquad (34)$$

The operation of this element is the following: Once the features $x_1, x_2, \ldots, x_l$ are applied to the input nodes, they are weighted by $w_1, w_2, \ldots, w_l$, respectively, and the results are summed, i.e.,

$$z = \sum_{i=1}^{l} w_i x_i + w_0 = \mathbf{w}^T \mathbf{x} + w_0 = g(\mathbf{x}) \qquad (35)$$

Then, the output of this element is $+1$ if $z > 0$ (points on the "positive" side) and -1 if $z < 0$ (points on the "negative" side). The w_i's are known as *synaptic weights* or simply *weights*, and the w_0 as the *threshold.* Also, $\mathbf{w}$ is known as the *weight vector.* Finally, the function f is called *activation function.* Sometimes, the levels of the hard limiter activation function are taken to be 1 and 0, instead of 1 and -1.

The above structuring element is known as *perceptron.* Clearly, *it implements the separation of the feature space by a hyperplane H defined by the parameters* w_i, $i = 0, \ldots, l$. That is, the perceptron will output $+1$ (-1) if the vector applied to its input nodes lies in the positive (negative) side of H.

The perceptron, with a generalized activation function f, will be the basic structuring element for a class of *artificial neural networks* known as *multilayer perceptrons,* that will be discussed later on. In this context, the simple perceptron element of Fig. 4 is also known as *neuron* or *node.*

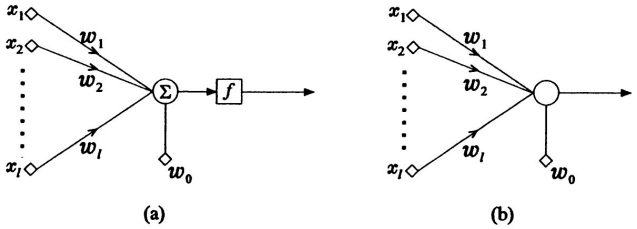


Figure 4 The basic perceptron architecture. In (b) the actions of summation and of the activation function have been combined.

B. Least-Squares Estimation

To determine the hyperplane defining a linear classifier via the perceptron algorithm presupposes that the two classes are linearly separable. However, due to the simplicity of a linear classifier it may be still desirable to design such a classifier even though we know that the feature vectors in the classes cannot be separated linearly. The goal now is to estimate the weights of the hyperplane so that an appropriately adopted *cost function* is optimized. The resulting linear classifier, although is suboptimal with respect to the classification error probability, is optimal with respect to the adopted criterion.

The *sun of error squares* cost function is a popular criterion usually encountered in practice and the weights of the hyperplane are estimated so that the cost function

$$J(\mathbf{w}) = \sum_{i=1}^{N} (y_i - \mathbf{x}_i^T \mathbf{w} - w_0)^2 = \sum_{i=1}^{N} e_i^2 \qquad (36)$$

becomes minimum. The class labels y_i are $+1$ if $\mathbf{x}_i$ originates from class ω_1 and -1 if belongs to ω_2. Ideally, one would like to get $\mathbf{x}^T\mathbf{w} + w_0 = +1$ for all points from class ω_1 and -1 for the points of class ω_2. However, this is not possible. In practice, all we can expect is to compute the weights so that to minimize the deviations from this ideal case, i.e., to compute the weights so that $J(\mathbf{w})$ is minimum. This is a typical least-squares problem, leading to the solution of a set of linear equations.

C. Support Vector Machines

We are now going to follow an alternative rationale for designing linear classifiers. We will focus on the two class linearly separable task, in order to demon-

[2]The trivial case where $\mathbf{w} = \mathbf{0}$, is not considered.

strate the philosophy of the technique. Generalizations for nonlinearly separable problems have also been proposed and used in practice.

Let $\mathbf{x}_i$, $i = 1,2,\ldots,N$, be the feature vectors of the training set, X. These belong to either of two classes, ω_1, ω_2, which are assumed to be linearly separable. The goal, once more, is to design a hyperplane

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 = 0 \tag{37}$$

that classifies correctly all the training vectors. Such a hyperplane is not unique and there is an infinite number of planes that can separate two linearly separable classes. The perceptron algorithm converges to any one of the possible solutions. Figure 5a illustrates the classification task with two possible hyperplane solutions. Both hyperplanes do the job for the training set. However, which one of the two, any sensible engineer would choose as the classifier for operation in practice, where data outside the training set would be fed to it. No doubt, the answer is: the full line one. The reason is that this hyperplane leaves more "room" on either side, so that data in both classes can move a bit more freely, with less risk of causing an error. Thus, such a hyperplane can be trusted more, when it is faced with the challenge of operating with unknown data and it is expected to lead to better *generalization performance of the classifier*. This refers to the capability of a classifier to operate satisfactorily when (unknown) data outside the training set are presented to it.

After the above brief discussion, we are ready to accept that a very sensible choice for the hyperplane classifier would be the one that leaves the maximum margin from both classes. This very sensible choice has a deeper justification, springing from the elegant Vapnik-Chernovenkis theory.

Let us now quantify the term "margin" that a hyperplane leaves from both classes. Every hyperplane is characterized by its direction (determined by $\mathbf{w}$) and its exact position in space (determined by w_0). Since we want to give no preference to either of the classes, then it is reasonable, for every direction, to consider only that hyperplane whose distance from the nearest points in ω_1 and ω_2 is the same. This is illustrated in Fig. 5b. The planes shown with a dark line are the selected ones from the infinite set in the respective direction. The margin for direction "1" is $2z_1$ and the margin for direction "2" is $2z_2$. *Our goal is to search for the direction that gives the maximum possible margin.* However, each hyperplane is determined within a scaling factor. We will free ourselves from it, by appropriate scaling of all the candidate hyperplanes. From simple geometry arguments it can be shown that the distance of a point from a hyperplane is given by

$$z = \frac{|g(\mathbf{x})|}{\|\mathbf{w}\|}$$

We can now scale $\mathbf{w}$, w_0 so that the value of $g(x)$, at the nearest points in ω_1, ω_2, is equal to 1 for ω_1 and, thus, equal to -1 for ω_2. This is equivalent with

1. Having a margin of $\frac{1}{\|\mathbf{w}\|} + \frac{1}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|}$
2. Requiring that

$$\mathbf{w}^T\mathbf{x} + w_0 \geq 1, \forall \mathbf{x} \in \omega_1$$
$$\mathbf{w}^T\mathbf{x} + w_0 \leq -1, \forall \mathbf{x} \in \omega_2$$

We have now reached the point where mathematics will take over. For each x_i, we denote the corresponding class indicator by y_i ($+1$ for ω_1, -1 for ω_2). Our task can now be summarized as: compute the parameters $\mathbf{w}$, w_0 of the hyperplane so that to:

$$\text{minimize } J(\mathbf{w}) \equiv \frac{1}{2}\|\mathbf{w}\|^2 \tag{38}$$

$$\text{subject to } y_i(\mathbf{w}^T\mathbf{x}_i + w_0) \geq 1,\ i = 1,2,\ldots,N \tag{39}$$

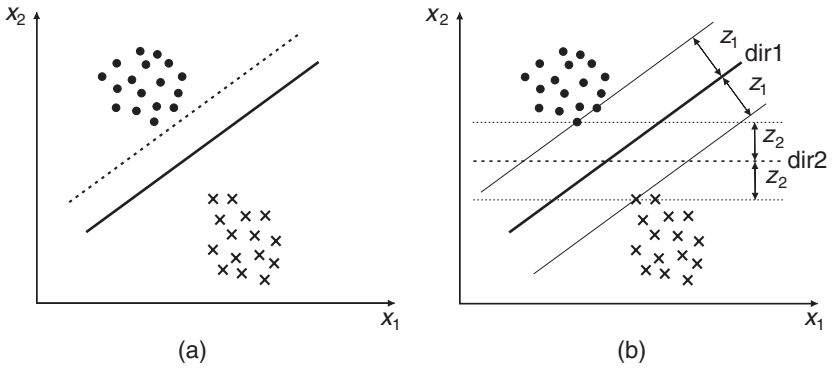


Figure 5 An example of linearly separable classes showing (a) two possible solutions and (b) two hyperplanes with different margins from the closest points in the classes.

Obviously, minimizing the norm results in maximum margin. This is a nonlinear (quadratic) optimization task subject to a set of linear inequality constraints, whose solution results to the optimal hyperplane. It turns out that the solution can be expressed as a linear combination of a subset of the training vectors $\mathbf{x}_i$, which are known as *support vectors*, and they are the training vectors that lie closer to the separating hyperplane.

VII. MULTILAYER PERCEPTRONS

As it was discussed in the previous section, the perceptron structure can deal with two-class problems, provided that the classes are linearly separable. Two well-known linearly separable problems are the *AND* or *OR* problems of the Boolean algebra which can be seen as PR tasks. Indeed, depending on the values of the input binary data, $\mathbf{x} = [x_1, x_2, \ldots, x_l]^T$, the output is either 1 or 0 and $\mathbf{x}$ is classified in either of the two classes (namely 1 or 0.) The respective truth tables are given in Table I. It is now apparent from the geometry of Figs. 6a and b that both problems are linearly separable. The question that arises now is how often linear separability is met in practice. One has not to go very far to see that even well-known simple problems are not linearly separable. Consider for example the *XOR* problem, where (0,0) and (1,1) are assigned to class 0 and (0,1) and (1,0) are assigned to class 1 (see Fig. 6c). Clearly, this problem cannot be solved by a single perceptron, since a single perceptron realizes one hyperplane and there is not a single hyperplane that can separate the vectors of the two classes. In order to solve this problem, two hyperplanes are needed, as shown in Fig. 6c. Each of these hyperplanes can be realized in the feature space by a perceptron. Let y_1, y_2 be the outputs of the two perceptrons, respectively. Then, as we know from the previous section, y_1, y_2, will be either 0 or 1, depending on the position of the input point with respect to the respective hyperplanes, $g_1(\mathbf{x})$, $g_2(\mathbf{x})$. What we have actually achieved is a transformation of the original space (x_1,x_2) into a new one (y_1,y_2) (Table II). The possible values in the new space lie on the *vertices of the unit square (hypercube in higher dimensions)*. Clearly, as it is apparent from Fig. 6d, a single hyperplane now suffices to separate the two classes in the transformed space (y_1,y_2). Speaking in terms of neurons, we need two neurons to realize the two hyperplanes g_1 and g_2 and, in the sequel, a single neuron that combines the outputs of the previous two neurons and realizes a third hyperplane in the (y_1,y_2) space. The structure shown in Fig. 7 is a possible one that solves the XOR problem. This is known as a *two-layer perceptron*. It consists of two layers of neurons. The first layer is known as hidden layer, and the second layer, known as the output layer, consists of a single neuron. The inputs (x_i) are applied to the input nodes, known as the *input layer*, which are nonprocessing nodes. The input layer has as many input nodes as the dimensionality of the input space. This is a special type of a neural network architecture.

Table I Truth Tables for AND and OR Problems

x_1	x_2	AND	Class	OR	Class
0	0	0	B	0	B
0	1	0	B	1	A
1	0	0	B	1	A
1	1	1	A	1	A

So far we have seen how a two-layer perceptron can be used to solve the *XOR* problem. The natural question that now arises is whether there are partitions of the feature space into classes, via hyperplanes, that cannot be solved by a two-layer architecture. Let us consider the example of Fig. 8a, where the shaded regions correspond to class 1 (ω_1) and the rest to class 0 (ω_2). Clearly, a two-layer network that will tackle this problem must have in its first layer as many neurons as the hyperplanes that define the regions in the feature space. Thus, speaking in the spirit of the solution given for the *XOR* problem, each of these neurons realizes one of the hyperplanes and at the same time performs a transformation of the original $\mathbf{x}$ space into the (y_1,y_2,y_3). Each one of the regions is named by the corresponding values of the triple (y_1,y_2,y_3), which now lie on the vertices of the three-dimensional unit cube, depending on the position (1 or 0) of $\mathbf{x}$ with respect to the respective plane (Fig. 8b). However, no single hyperplane, realized by the output neuron in the three-dimensional (y_1,y_2,y_3) space, can separate the vertices of the hypercube that correspond to class 1 from those that correspond to class 0. Thus, this problem cannot be solved by a two-layer network.

However, adopting the same philosophy as before, this problem can be solved by a three-layer network as follows. Having transformed the original problem to that of Fig. 8b, we can use two extra neurons to realize the hyperplanes g_4 and g_5 in the transformed three-dimensional space. g_4 (g_5) leaves the vertex (1,0,0) ((0,1,1)) to its positive side and all the others to its negative side. In other words, the neuron realizing g_4 will signal a 1 for all the points lying in the "100" re-

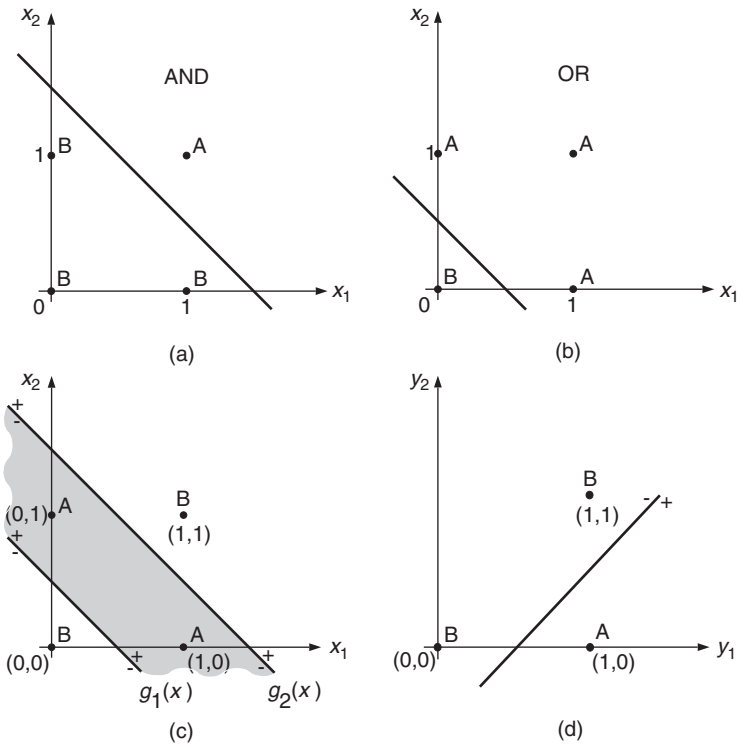


Figure 6 Classes and decision hyperplanes for the basic Boolean operations.

gion and a 0 for all the others. The neuron, realizing g_5, will signal 1 for the points in "011" region and zero for all the others. Thus, all points of class ω_2 signal always a 0 on both neurons ((0,0)), in contrast to the points of class ω_1 that signal 1 in only one of the neurons ((1,0) or (0,1)). This defines a further transformation into the vertices of a unit square, shown in Fig. 8c, where the problem is now linearly separable by g_6, realized by another neuron.

The general structure of the network that implements the given partition is shown in Fig. 9, and it is known as a three-layer perceptron. The first layer of neurons realizes the hyperplanes dividing the feature space, the neurons of the second layer realize hyperplanes in the transformed space, and the output neuron basically realizes the OR Boolean operation, whose output is 1 for all the points of ω_1 and 0 otherwise. In general, a *multilayer perceptron* consists of L layers of neurons (excluding the input layer) and the rth layer consists of k_r neurons. The Lth (final) layer is known as the *output layer* and its neurons *output neurons*. All the other layers are called *hidden layers* and the respective neurons *hidden neurons*.[3]

Table II Input and Transformed Space Values for the XOR Problem

x_1	x_2	y_1	y_2	Class
0	0	0	0	B
0	1	1	0	A
1	0	1	0	A
1	1	1	1	B

The methodology of constructing a three-layer perceptron (which by no means is optimal with respect to the number of neurons used) can be applied to any partition of the input space in two classes, where each class is a union of polyhedral sets. A *polyhedral set* is a union of half-spaces. For example, in Fig. 8a we have three hyerplanes (lines in R^2) that form seven nonintersected polyhedral sets. *Hence, a three-layer perceptron can solve any classification task, where classes consist of unions of polyhedral regions.* Although, so far, we focused on the two-class problem, the generalization to the multiclass problem is straightforward. For an M-class task the output neurons are increased to M, and it is easy to show that such a three-layer

[3]It should be emphasized that this is just one type of neural network architecture. For other types of neural network architectures see Haykin and Zurada.

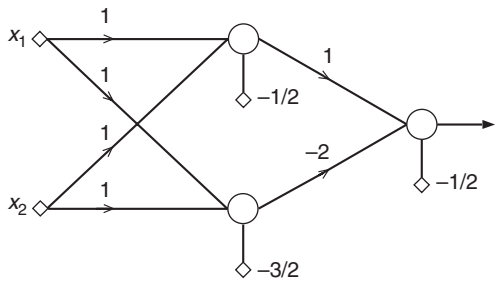


Figure 7 A two-layer perceptron solving the XOR problem.

perceptron can be trained to classify *any union* of polyhedral regions in M classes.

In summary, we can say that *the first layer of neurons forms the hyperplanes, the second layer forms the regions, and the third layer forms the classes.*

A. The Back Propagation Algorithm

In the previous subsection, the focus was on the potential capabilities of a three-layer perceptron to classify unions of polyhedral regions. In practice, these regions are not known and the analytic computation of the hyperplane equations is not, in general, possible. In practice, a procedure for *training the neurons* is required in order to estimate the unknown parameters.

By far the most well-known and widely used learning algorithm to estimate the values of the synaptic weights and the thresholds is the so-called *back propagation (BP) algorithm* and its variants. According to this training method, the size (number of layers and number of nodes in each layer) of the network remains fixed.

The BP algorithm relies on a training data set X. Note that we can use various representations for labeling the respective class of each vector of the training set. One representation that is frequently used in practice is the following:

$$\mathbf{y}(i) = [\overbrace{0, \ldots, 0, \underbrace{1}_{y_k(i)}, 0, \ldots, 0}^{M}] \tag{40}$$

where $y(i)$ is the vector having zeros everywhere except at position k, that corresponds to the class in which the ith training vector belongs, where the value is 1.

The BP algorithm is a gradient descent scheme that determines a local minimum of the cost function which is adopted. The most popular cost function is the following:

$$J = \sum_{i=1}^{N} \mathcal{E}(i) \tag{41}$$

where

$$\mathcal{E}(i) = \frac{1}{2} \sum_{p=1}^{M} e_p^2(i) = \frac{1}{2} \sum_{p=1}^{M} (\hat{y}_p(i) - y_p(i))^2 \tag{42}$$

with $\hat{y}_p(i)$ being the true output (in general different than 1 or 0) of the pth output node of the network when the ith feature vector is fed to the network and M is the number of neurons of the output layer. Note that, in general, y_p (0 or 1) and $\hat{y}_p$ are different and are known as the *desired* and *true outputs*, respectively.

The gradient descent optimization procedure requires the computation of the gradient of the cost function with respect to the unknown parameters. However, differentiation of J is not possible, since $\hat{y}_p(i)$ is expressed via the hard limiter function, which is not a differentiable function. One way to overcome this difficulty is to use a smooth function that approximates the hard limiter function, which will be continuous and differentiable. One such function is the *logistic function*, shown in Fig. 10, which is defined as

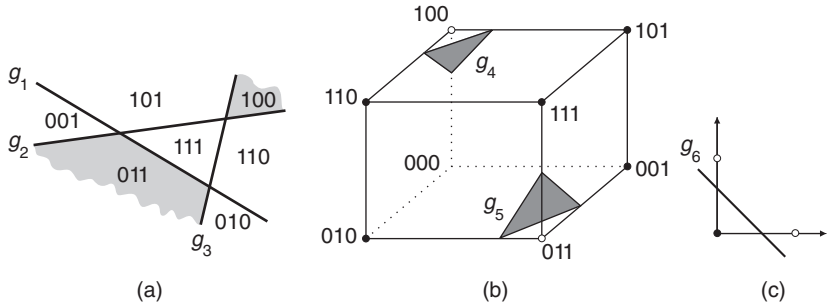


Figure 8 (a) Partition of the feature space by hyperplanes, realized by the neurons of the first hidden layer, (b) hyperplanes realized in the transformed space by the neurons of the second hidden layer, and (c) the hyperplane realized by the output neuron in a three layer perceptron.

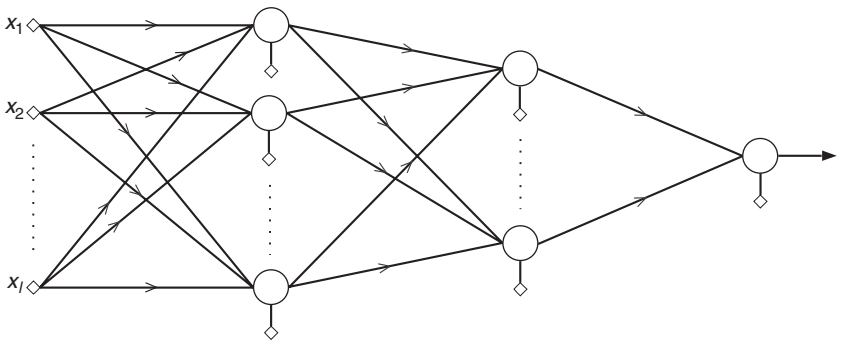


Figure 9 Architecture of a three-layer perceptron with a single output neuron.

$$f(x) = \frac{1}{1 + \exp(-ax)} \tag{43}$$

where a is the slope parameter. The larger the value of a, the more the logistic function approaches the hard limiter. Clearly, this function takes values between 0 and 1. This function belongs to a broader class of functions known as *squashing functions*, whose outputs are limited in a continuous real number interval. Other such functions are also possible.

The basic iteration step of the BP algorithm builds around the update of the weight vector (including the threshold) associated with a single neuron. Assuming $\mathbf{w}_j^r$ to be the weight vector of the jth neuron in the rth layer, the basic iteration step is of the form

$$\mathbf{w}_j^r(new) = \mathbf{w}_j^r(old) + \Delta\mathbf{w}_j^r$$

where

$$\Delta\mathbf{w}_j^r = -\mu \frac{\partial J}{\partial \mathbf{w}_j^r}$$

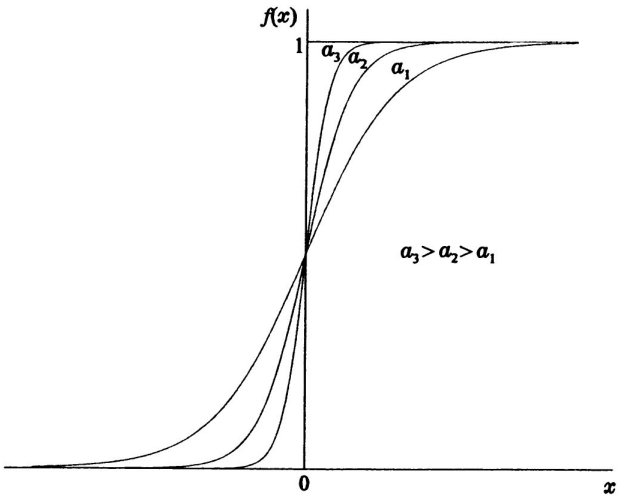


Figure 10 The logistic function.

and $\mathbf{w}_j^r(old)$ is the current estimate of the unknown vector and $\Delta\mathbf{w}_j^r$ the correction vector to obtain the next estimate. r runs over all layers and j over all neurons of each layer. The major difficulty in the BP algorithm is the computation of the involved gradients, due to the nonlinear dependence of the cost function J on $\mathbf{w}_j^r$. The computation of the gradients starts with the neurons of the last layer and then propagates backward, and this is the reason for the specific name of the algorithm.

The literature related to the BP algorithm is very rich and a large number of variants have been proposed. All of them try, in one way or another, to overcome the drawbacks associated with any nonlinear cost function optimization task, namely

- Local minima
- Speed of convergence
- Computational complexity

VIII. GENERALIZED LINEAR CLASSIFIERS

The basic idea on which we chose to build our presentation of the multilayer perceptrons was that of the transformation of the original input space into a new one, where the classification task becomes linearly separable. This methodology has been exploited in various forms in PR. Behind it, lies the very powerful Cover's theorem. It states that as the dimensionality, l, of the feature space goes to infinity, the probability of *any* two groupings of N points being linearly separable approaches unity, provided that $N < 2(l + 1)$.

For the case of a two layer perceptron (XOR case) the adopted mapping was of the form

$$\mathbf{x} \rightarrow \mathbf{y}$$

where

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_k \end{bmatrix} = \begin{bmatrix} f(g_1(\mathbf{x})) \\ \vdots \\ f(g_k(\mathbf{x})) \end{bmatrix} \tag{44}$$

where $f(\cdot)$ was the activation function and $g_i(\mathbf{x})$ the hyperplane realized by the ith neuron. Emancipating from the basic perceptron, one can write

$$\mathbf{x} \in \mathcal{R}^l \rightarrow \mathbf{y} \in \mathcal{R}^k$$

$$\mathbf{y} \equiv [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})]^T \tag{45}$$

where $f(\cdot)$ are appropriate, for each problem, nonlinear functions, and k a large, enough, integer, defining the dimensionality of the transformed space. The general structure of a generalized linear classifier is shown in Fig. 11. The name "generalized linear" classifier indicates that in the new space the task is treated as a linear one, i.e.,

$$w_0 + \mathbf{w}^T\mathbf{y} > 0, \text{ classify } \mathbf{x} \text{ in } \omega_1 \tag{46}$$

$$w_0 + \mathbf{w}^T\mathbf{y} < 0, \text{ classify } \mathbf{x} \text{ in } \omega_2 \tag{47}$$

Depending on the choice of $f_i(\cdot)$, different classifiers result, including, as special cases, the two-layer perceptron, the radial basis function (RBF) networks, the polynomial classifiers, and the support vector machines.

IX. DECISION TREES

In this section we briefly review a large class of nonlinear classifiers known as *decision trees*. They are *multistage* decision systems where classes are sequentially rejected until we reach a finally accepted class. To this end, the feature space is split into unique regions, corresponding to the classes, *in a sequential manner*. Upon the arrival of a feature vector, the searching of the region into which the feature vector will be assigned, is achieved via a sequence of decisions along a path of *nodes* of an appropriately constructed *tree*. Such schemes offer advantages when a large number of classes is involved. The most popular among the decision trees are those, which split the space into hyperectangles, with sides parallel to the axis. The sequence of decisions is applied on *individual* features, and the questions to be answered are of the form *is feature* $x_i \leq \alpha$?, where α is a threshold value. Such trees are known as *ordinary binary classification trees* (OBCT).

The basic idea behind an OBCT is demonstrated via the simplified example of Fig. 12. By a successive sequential splitting of the space we have created regions corresponding to the various classes. Figure 13 shows the respective binary tree with its decision nodes and leaves. Note that it is possible to reach a decision without having tested *all* the available features.

In general, in order to construct a decision tree, a splitting criterion (i.e., optimizing function) may be adopted for each node. For further information and a deeper study of this class of classifiers the interested reader may consult, e.g., Breiman and colleagues and Quinlan.

X. SIZE OF NETWORKS AND GENERALIZATION PROPERTIES

In the previous sections, we assumed the number of free parameters describing a classifier (size of the classifier) to be known and fixed. For example, in a multilayer perceptron the number of free parameters is the total number of neuron synapses and thresholds. The task of how one determines the appropriate number of layers and neurons was not of interest to us. This task will become our major focus now.

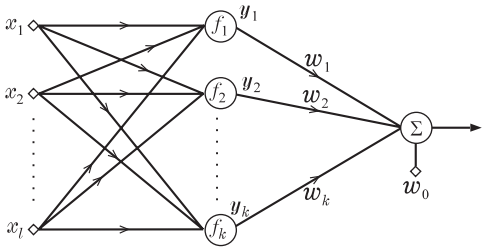


Figure 11 Generalized linear classifier.

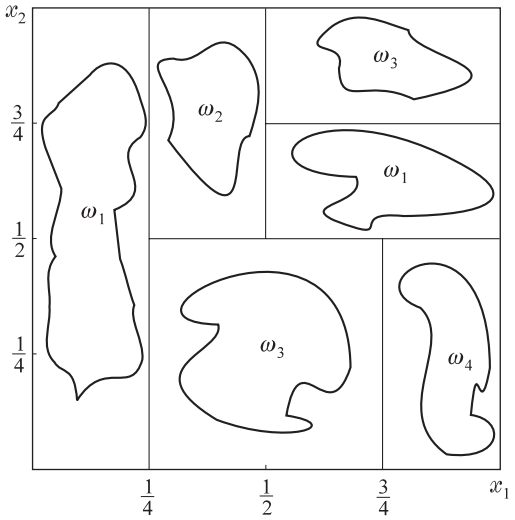


Figure 12 Decision tree partition.

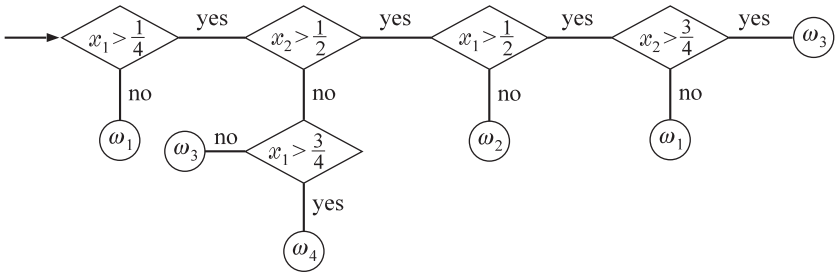


Figure 13 Decision tree classification.

One answer to the problem could be to choose the size of the classifier large enough, and leave the training to decide about the parameters. Little thought reveals that such an approach is rather naive. Besides the associated computational complexity problems, there is a major reason for which the number of free parameters should be kept as small as possible. This is imposed by the *generalization capabilities* which the classifier must possess. The term generalization refers to the capability of any classifier to classify correctly feature vectors, which were not presented to it during the training phase. That is, the capability of a classification system to decide upon unknown, to it, data, based on what it has learned from the training set. Taking for granted the finite (and in many cases small) number N of training samples, the number of free parameters to be estimated should be: (1) large enough, so that to learn whatever common exists among the feature vectors of each class, and at the same time whatever makes one class different from the other, and (2) small enough, with respect to N, so that not to be able to learn the underlying differences among the data of the same class. When the number of free parameters is large, the classifier tends to adapt to the particular details of the specific training data set. This is known as *overfitting* and leads to poor generalization performance, when the classifier is called to operate upon feature vectors unknown to it. In conclusion, the classifier should have the smallest possible size, so that to adjust its weights to the largest regularities in the data and ignore the smaller ones, which might also be the result of noisy measurements.

Although the above comments hold true for most parametric classifiers used in practice, an interesting result comes from the Vapnik-Chernovenkis (VC) learning theory. The crucial factor that determines the generalization capabilities of a classifier is its, so called, VC dimension. Basically this number can be considered as the "intrinsic capacity" of a classifier, that is, its ability to learn any training data set without error. For good generalization performance the number of training samples must exceed the VC dimension *sufficiently*. Machines with high VC dimension (capacity), with respect to the size of the training data, can learn the particular characteristics of each training set, thus they lack the ability to generalize well. For most of the known classifiers used in practice the VC dimension is directly related to the number of free parameters. However, one can construct classifiers which do not conform with this trend.

XI. FEATURE GENERATION AND SELECTION

In the previous sections we assumed that the features presented to the classifier, for a given classification task, were known. This section deals with the stages concerning their generation and the final selection of the most informative ones, from the classification point of view.

A. Feature Generation

This stage is very much dependent on the problem at hand and the nature of signals associated with the patterns, e.g., images, speech, video, music. However, there are certain methodologies which, in one way or another, enter in a number of diverse applications. Feature generation based on the application of a mathematical transform on the available input data obtained from a given pattern is such an approach used in various applications.

The basic reasoning behind transform-based features is that an appropriately chosen transform can exploit and remove information redundancies, which usually exist in the original set of samples. Let us take, for example, an image resulting from a sensing device, e.g., X-rays or a camera. The pixels (i.e., the input samples) at the various positions in the image

have a large degree of correlation, due to the internal morphological consistencies of real-world images which distinguish them from noise. Thus, if one uses the pixels as features there will be a large degree of redundant information. Alternatively, if one obtains the Fourier transform, for example, of a typical real-world image, one immediately realizes that most of the energy lies in the low-frequency components, due to the high correlation between the pixels. Hence, using the Fourier coefficients as features seems a reasonable choice, since the low-energy, high-frequency coefficients can be neglected, with little loss in information. The Fourier transform is just one of the tools from a palette of possible transformations.

Given a vector of input samples, $\mathbf{x}$, a linear transform of it is of the form

$$\mathbf{y} = A\mathbf{x} \tag{48}$$

The choice of matrix A determines the specific transform. The optimal one is the Karhunen-Loève (KL) transform, in the sense that the elements of $\mathbf{y}$ are mutually uncorrelated, i.e.,

$$E[y_i y_j] = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{49}$$

and $E[\cdot]$ denotes expected value. In spite of its optimality, the KL transform is not always useful, since the computation of the elements of A leads to an eigenvalue-eigenvector problem, which usually poses high computational demands. Fortunately, a number of linear transforms exist, with an associated matrix whose elements are not signal dependent but are known *a priori*, and can give close to the optimal results. Examples of such transforms are the Discrete Fourier Transform, Discrete Cosine Transform, Discrete Sine Transform, the Haddamard Transform and the Wavelet transform.

B. Feature Selection

There is more than one reason in support of the necessity to reduce the number of feature parameters to a sufficient minimum. Computational complexity is the obvious one. Another related reason is that although two features may carry good classification information when treated separately, there is little gain if they are combined together as elements in a feature vector, due to a high mutual correlation. Thus, complexity increases without much gain. Another major reason is that imposed by the required generalization properties of the classifier, as already discussed in Section X. According to the discussion there, the higher the ratio of the number of training patterns, N, to the number of free classifier parameters, the better the generalization properties of the resulting classifier. Thus, for a finite and usually limited number N of training patterns, keeping the number of feature parameters as small as possible is in line with our desire to design classifiers of small size, with good generalization capabilities.

The major task of the feature selection stage can now be summarized as follows: *Given a number of features, how can one select the most important of those, so that to reduce their number and at the same time to retain as much as possible from their discriminatory information.* It must be emphasized that this step is very crucial. If we select features with little discrimination power, the subsequent design of a classifier would lead to poor performance. On the other hand, if information-rich features are selected, the design of a classifier can be greatly simplified. In a more quantitative description, we should aim to select features leading to *large between class distance and small within class variance*, in the feature vector space. This means that features should take distant values in the different classes and close located values for the same class.

There are two major directions for feature selection. One of them is to treat each one of the features individually and test its class discriminatory capabilities. This is a first step, usually followed in feature selection, so that to get rid of "information-poor" features. Statistical hypothesis testing is a commonly adopted methodology. The idea is to test if the mean values in the various classes of a specific feature *differ significantly*. If they are not, the feature is discarded and not considered in subsequent stages. This is justified, since if the respective mean values do not differ significantly, it is an indication that the values of the feature in the different classes tend to overlap, thus the feature is equipped with little class discriminatory information.

The second direction is to consider the features, that passed the first test, in groups. To approach this task one needs to have at his/her disposal

- Measures that quantify the class discriminatory capability of vectors.
- Efficient algorithms that result into the "best" combination of l features out of a larger number of candidate features that passed the first test. The resulting feature combination is, usually, suboptimal (with respect to the adopted measure), since the consideration of all possible

combinations of features is prohibitive in computational requirements for most practical applications.

A number of various class separability measures have been suggested such as, the *divergence* and the *Bhattacharyya distance*. A popular set of measures, used in practice, are those built around the *scatter matrices*, which code the way in which the values of the feature vectors in the various classes are scattered in the l-dimensional space, e.g., measuring the between-class and the within-class variances.

XII. SYSTEM EVALUATION

This section deals with the last stage of a classification system, whose goal is to evaluate the performance of the designed system, quantified by the classification error probability, that is, the probability of a pattern being assigned in the wrong class.

Let us assume that the task consists of M classes and we are given a *test data set* of N points, for testing the performance of the classifier, which has been designed on the basis of another (independent) training data set. If N_i points of the test set belong to class ω_i, $i = 1,2, \ldots, M$, and k_i of those are misclassified, then an estimate of the respective error probability is given by

$$\hat{P}_i = \frac{k_i}{N_i} \tag{50}$$

and the total error probability is estimated by

$$\hat{P} = \sum_{i=1}^{M} P(\omega_i) \frac{k_i}{N_i} \tag{51}$$

where $P(\omega_i)$ is the *a priori* probability for ω_i. For equiprobable classes $P(\omega_i) = 1/M$. There is nothing exciting in all these. However, the interesting and not trivial task emerges in the way the test set is chosen. In most of the cases, all we have available is a finite and usually limited number of data points, both for testing and training. The natural question now is how one can use this data set effectively for both tasks. Below are the two most popular alternatives encountered in practice.

1. *Holdout method:* The available data set is divided into two subsets, one for training and one for testing. The major drawback of this technique is that it reduces the size for both the training as well as the testing data. Another problem is to decide how many of these N available data will be allocated to the training set and how many to the test one. Efforts made to optimize the sizes of the two sets have not yet led to practical results.
2. *Leave-one-out method:* This method frees itself from the dilemma associated with the holdout method. The training is performed using $N - 1$ samples, and the test is carried out using the excluded sample. If this is misclassified an error is counted. This is repeated N times, each time excluding a *different* sample. The total number of errors leads to the estimation of the classification error probability. Thus, training is achieved using, basically, all samples and at the same time independence between training and test sets is maintained. The major drawback of the technique is the increased computational complexity. A number of variants of the method have been suggested in the related literature in order to reduce the computational demands.

XIII. CLUSTERING

All sections so far dealt with supervised classification. In this section the focus of interest is turned into the unsupervised case, where, class labeling of the training patterns is not available. Thus, our major concern now becomes to "reveal" the organization of patterns into "sensible" clusters (groups), which will allow us to discover similarities and differences among patterns and to derive useful conclusions about them. This idea is met in many fields, such as, life sciences (biology, zoology), medical sciences (psychiatry, pathology), social sciences (sociology, archeology), earth sciences (geography, geology) and engineering. Clustering may be found under different names in different contexts, such as unsupervised learning and learning without a teacher (in PR), numerical taxonomy (in biology, ecology), typology (in social sciences), partition (in graph theory), etc.

Clustering is one of the most primitive mental activities of humans in order to handle the huge amount of information they receive every day. Processing every piece of information as a single entity would be impossible. Thus, humans tend to categorize these entities (i.e., objects, persons, events) into clusters. Each cluster is then characterized by the common attributes of the entities it contains. For example, most of the humans "possess" a cluster "dog." If someone sees

a dog sleeping on the grass, he/she will identify it as an entity of the cluster dog. Thus, he/she will infer that this entity barks even though he/she has never listened to this specific entity to bark before.

As it was the case with the supervised learning, we will assume that all patterns are represented in terms of *features*, which form l-dimensional feature vectors.

The basic steps which an expert must follow in order to develop a clustering task are the following:

- *Selection of features*—Features must be properly selected so that to encode as much information as possible, concerning the task of interest. Once more, parsimonity and, thus, minimum information redundancy among the features is a major goal.
- *Proximity measure*—This is a measure that quantifies how "similar" or "dissimilar" two feature vectors are.
- *Clustering criterion*—This depends on the interpretation which the expert gives to the term "sensible," based on the type of clusters that are expected to underlie the data set. For example, a compact cluster may be sensible according to one criterion, while an elongated cluster may be sensible according to another. The clustering criterion may be expressed via a cost function or some other types of rules.
- *Clustering algorithms*—Having adopted a proximity measure and a clustering criterion, this step refers to the choice of a specific algorithmic scheme, which unravels the clustering structure of the data set.
- *Validation of the results*—Once the results of the clustering algorithm have been obtained, we have to verify their correctness. This is usually carried out using appropriate tests.
- *Interpretation of the results*—In many cases, the expert of the application field must integrate the results of clustering with other experimental evidence and analysis, in order to draw the right conclusions.

As one may have already suspected, different choices of features, proximity measures, clustering criteria, and clustering algorithms may lead to totally different clustering results. *Subjectivity is a reality we have to live with.* To demonstrate this, let us consider the following example.

Consider Fig. 14. How many sensible clusterings can we obtain for these points? The most "logical" answer seems to be two. The first clustering contains four clusters (surrounded by solid circles). The second clustering contains two clusters (surrounded by dashed lines). Which clustering is correct? It seems that there is no definite answer. Both clusterings are valid. The best thing to do is to give the results to an expert and let him decide about the most sensible one. Thus, the final answer to the above question will be biased to the knowledge of the expert.

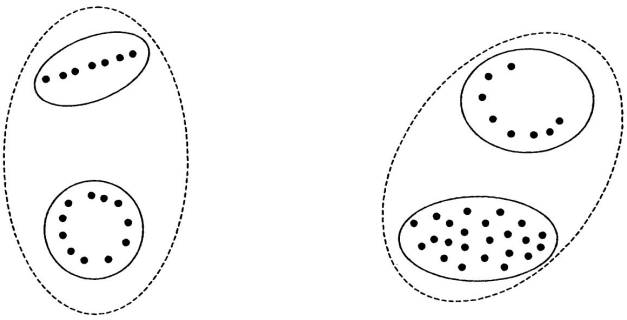


Figure 14 Clustering of data may result either in two or four clusters.

A. Definitions of Clustering

Let X be our data set, i.e.,

$$X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\} \tag{52}$$

We define as an *m-clustering* of X, $\mathcal{R}$, the partition of X into m sets (*clusters*), $C_1, \ldots, C_m$, so that the following three conditions are met:

- $C_i \neq \varnothing$, $i = 1, \ldots, m$,
- $\bigcup_{i=1}^{m} C_i = X$
- $C_i \cap C_j = \varnothing$, $i \neq j$, $i, j = 1, \ldots, m$.

In addition, the feature vectors contained in a cluster C_i are "more similar" to each other and "less similar" to the feature vectors of the other clusters. Quantifying the terms "similar" and "dissimilar" depends very much on the types of clusters involved.

Note that, under the definitions of clustering given above, each vector belongs to a single cluster. This type of clustering is sometimes called *hard* or *crisp*. An alternative definition is in terms of the *fuzzy* sets, introduced by Zadeh. A fuzzy clustering of X into m clusters consists of m functions u_i, where

$$u_i : X \rightarrow [0,1], \; i = 1, \ldots, m \tag{53}$$

and

$$\sum_{i=1}^{m} u_i(\mathbf{x}) = 1, \; \forall \, \mathbf{x} \in X \tag{54}$$

These are called *membership functions*. The value of a fuzzy membership function is a mathematical characterization of a set, i.e., a cluster in our case, which may not be precisely defined. That is, each vector $\mathbf{x}$ belongs

to more than one clusters simultaneously "up to some degree," which is quantified by the corresponding value of u_i in the interval [0,1]. Values close to unity show high "grade of membership" to the corresponding cluster and values close to zero low grade of membership. The values of these membership functions are indicative of the structure of the data set, in the sense that if a membership function has close to unity values for two vectors of $\mathbf{x}$, i.e., $\mathbf{x}_i$, $\mathbf{x}_j$, these are considered similar to each other. The definition of clustering into m distinct sets C_i, given before, can be recovered as a special case of the fuzzy clustering if we define the fuzzy membership functions u_i so that to take values in {0, 1}, that is, to be either 1 or 0. In this sense, each data vector belongs exclusively to one cluster and the membership functions are now called *characteristic functions*.

B. Clustering Algorithms

Given the time and resources, the best way to assign the feature vectors $\mathbf{x}_i$, $i = 1, \ldots N$, of a set X, into clusters would be to identify all possible partitions and to select the most sensible one according to a preselected criterion. However, this is not possible even for moderate values of N. Indeed, let $S(N,m)$ denote the number of all possible clusterings of N vectors into m groups. It is reminded that, by definition, no cluster is empty. It is clear that the following conditions hold:

- $S(N,1) = 1$
- $S(N,N) = 1$
- $S(N,m) = 0$, for $m > N$

It can be shown that

$$S(N,m) = \frac{1}{m!} \sum_{i=0}^{m} (-1)^{m-i} \binom{m}{i} i^N \qquad (55)$$

where

$$\binom{m}{i} = \frac{m!}{(m-i)!\,i!}$$

Some numerical values of (Eq.) 55 are

- $S(15,3) = 2\ 375\ 101$
- $S(25,8) = 690\ 223\ 721\ 118\ 368\ 580$
- $S(100, 5) \simeq 10^{68}$

It is clear that the above calculations are valid for the case where the number of clusters is fixed. If this is not the case, one has to enumerate all possible clusterings for all possible values of m. From the above analysis, it is obvious that the evaluation of all clusterings in order to identify the most sensible one is impractical even for moderate values of N. Indeed, if, for example, one has to evaluate all possible clusterings of 100 objects into 5 clusters, with a computer that evaluates each single clustering in 10^{-12} sec, the most sensible clustering would be available after approximately 10^{48} years.

Clustering algorithms may be viewed as schemes that provide us with sensible clusterings, by considering *only a small fraction of the set containing all possible partitions of X. The result depends on the specific algorithm and the criteria used.* Thus, a clustering algorithm is a learning procedure that tries to identify the specific characteristics of the clusters, underlying the data set. The related literature is rich in various methodologies and approaches for the development of clustering algorithms. The interested reader may refer to Jain and Dubes and Theodoridis and Koutroumbas for a more extensive treatment of the subject. In the sequel, we provide two examples of clustering algorithms that offer the reader a bit of the flavor of the topic.

C. The Isodata or k-Means or c-Means Algorithm

This is one of the most celebrated and widely used clustering algorithms. The number of clusters m, underlying the data, is considered known and fixed. Each cluster is represented by a vector $\boldsymbol{\theta}_j$, $j = 1,2, \ldots, m$, known as *cluster representative*. The goal of the algorithm is to place the cluster representatives optimally in the l-dimensional feature space and then assign each feature vector, $\mathbf{x}_i$, $i = 1,2, \ldots, N$, to the cluster, whose representative is closest (based on the Euclidean distance) to it.

1. *The Algorithm*

- Choose arbitrarily initial estimates $\boldsymbol{\theta}_j(0)$ for the $\boldsymbol{\theta}_j$'s $j = 1, \ldots, m$.
- Repeat
  - For $i = 1$ to N
    - Determine the closest representative, say $\boldsymbol{\theta}_j$, for $\mathbf{x}_i$.
    - Set $b(i) = j$
  - End { For }
  - For $i = 1$ to m
    - *Parameter Updating:* Determine $\boldsymbol{\theta}_j$ as the mean of the vectors $\mathbf{x}_i \in X$ with $b(i) = j$.
  - End { For }
- Until no change in $\boldsymbol{\theta}_j$ occurs between two successive iterations.

An advantage of the above algorithm is its computational simplicity.

D. Agglomerative Algorithms

The isodata algorithm assumes that the number of clusters is fixed and known *a priori*. A different philosophy is adopted in the family of algorithms known as *hierarchical clustering algorithms*. Instead of producing a single clustering they produce a hierarchy of clusterings. This kind of algorithms is usually met in social sciences and biological taxonomy, modern biology, medicine and archeology, computer science, and engineering.

A clustering $\mathcal{R}_1$ containing k clusters is said to be *nested* in the clustering $\mathcal{R}_2$, which contains $r(<k)$ clusters, if *each* cluster in $\mathcal{R}_1$ is a subset of a set in $\mathcal{R}_2$ and at least one cluster of $\mathcal{R}_1$ is a proper subset of $\mathcal{R}_2$. In this case we write $\mathcal{R}_1 \subset \mathcal{R}_2$. For example, the clustering $\mathcal{R}_1 = \{\{x_1,x_3\}, \{x_4\}, \{x_2,x_5\}\}$ is nested in $\mathcal{R}_2 = \{\{x_1,x_3,x_4\}, \{x_2,x_5\}\}$. On the other hand, $\mathcal{R}_1$ is not nested in $\mathcal{R}_3 = \{\{x_1,x_4\}, \{x_3\}, \{x_2,x_5\}\}$. It is clear that a clustering is not nested to itself.

Hierarchical clustering algorithms produce a *hierarchy of nested clusterings*. More specifically, these algorithms involve N steps, as many as the number of data vectors. At each step t, a new clustering is obtained based on the clustering produced at the previous step $t-1$. The most common category of these algorithms are the *agglomerative algorithms*.

The initial clustering $\mathcal{R}_0$ for the agglomerative algorithms consists of N clusters each one containing a single element of X. At the first step, the clustering $\mathcal{R}_1$ is produced. It contains $N-1$ sets, such that $\mathcal{R}_0 \subset \mathcal{R}_1$. This procedure continues until the final clustering, $\mathcal{R}_{N-1}$, is obtained, which contains a single set, that is, the set of data, X. Notice that for the hierarchy of the resulting clusterings we have

$\mathcal{R}_0 \subset \mathcal{R}_1 \subset \ldots \subset \mathcal{R}_{N-1}$

Let $g(C_i,C_j)$ be a function defined for all possible pairs of clusters of X. This function measures the proximity (similarity or dissimilarity) between C_i and C_j. Let t denote the current level of hierarchy. Then the general agglomerative scheme may be stated as follows:

1. Generalized Agglomerative Scheme (GAS)

1. *Initialization:*

- 1.1 Choose $\mathcal{R}_0 = \{C_i = \{\mathbf{x}_i\}, i = 1, \ldots, N\}$ as the initial clustering.
- 1.2 $t = 0$.

2. *Repeat:*

- 2.1 $t = t + 1$
- 2.2 Among all possible pairs of clusters (C_r,C_s) in $\mathcal{R}_{t-1}$ find the one, say (C_i,C_j), such that

$$g(C_i,C_j) = \begin{cases} \min_{r,s} g(C_r,C_s), & \text{if } g \text{ is a dissimilarity function} \\ \max_{r,s} g(C_r,C_s), & \text{if } g \text{ is a similarity function} \end{cases} \quad (56)$$

- 2.3 Define $C_q = C_i \cup C_j$ and produce the new clustering $\mathcal{R}_t = (\mathcal{R}_{t-1} - \{C_i,C_j\}) \cup \{C_q\}$.

3. Until all vectors lie in a single cluster.

It is clear that the above scheme creates a hierarchy of N clusterings so that each one is nested in all successive clusterings, that is, $\mathcal{R}_t \subset \mathcal{R}_s$, $t < s$, $s = 1, \ldots, N-1$. Alternatively we can say that *if two vectors come together into a single cluster at level t of the above hierarchy, they will remain in the same cluster for all subsequent clusterings*. This is another way of viewing the nesting property.

At each level t, there are $N - t$ clusters. Thus, in order to determine the pair of clusters that is going to be merged at the $t + 1$ level

$$\binom{N-t}{2} \equiv (N-t)(N-t-1)/2$$

pairs of clusters have to be considered. Thus, the total number of pairs which have to be examined throughout the whole clustering process is

$$\sum_{t=0}^{N-1} \binom{N-t}{2} = \sum_{k=1}^{N} \binom{k}{2} = \frac{(N-1)N(N+1)}{6}$$

that is, the total number of operations required by an agglomerative scheme is proportional to N^3. Moreover, the exact complexity of the algorithm depends on the definition of g.

XIV. CONCLUSION

The goal of this article was to review the basic concepts and methodologies used in PR. The adopted approach was to focus on the major stages followed for the design of a PR system for the automatic classification of patterns into one from a number of classes. More emphasis was given on the classifier design stage, and a number of commonly used classifiers were reviewed, including the Bayesian, the multilayer perceptrons, and the support vector machines. Finally,

the basic definitions of unsupervised PR, that is, clustering, were discussed and the isodata and aglomerative algorithms were presented.

SEE ALSO THE FOLLOWING ARTICLES

Data Mining • Decision Theory • Expert Systems Construction • Machine Learning • Robotics • Search Techniques • Speech Recognition • Uncertainty

BIBLIOGRAPHY

Bishop, C. M. (1995). *Neural networks for pattern recognition*. London: Oxford University Press.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1993). *Classification and regression trees*. London: Chapman & Hall.

Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, Vol. 2(2), pp. 1–47.

Devijver, P. A., and Kittler, J. (1982). *Pattern recognition: A statistical approach*. Englewood Cliffs, NJ: Prentice Hall.

Devroye, L., Gyorfi, L., and Lugosi, G. A. (1996). *A probabilistic theory of pattern recognition*. Berlin: Springer-Verlag.

Duda, R., Hart, P. E., and Stork, D. G. (2001). *Pattern classification*, Second Edition. New York: John Wiley.

Fukunaga, K. (1990). *Introduction to statistical pattern recognition*, 2nd ed. San Diego, CA: Academic Press.

Gallant, S. I. (1990). Perceptron based learning algorithms. *IEEE Transactions on Neural Networks*, Vol. 1(2), 179–191.

Haykin, S. (1999). *Neural networks: A comprehensive foundation*, 2nd Ed. Englewood Cliffs, NJ: Prentice Hall.

Jain, A. K., and Dubes, R. C. (1998). *Algorithms for clustering data*. Englewood Cliffs, NJ: Prentice Hall.

Minsky, M. L., and Papert, S. A. (1998). *Perceptrons*, expanded edition. MIT Press, Cambridge, MA: MIT Press.

Papoulis, A. (1991). *Probability, random variables and stochastic processes*, 3rd ed., New York: McGraw-Hill.

Parzen, E. (1962). On the estimation of a probability density function and mode. *Ann. Math.m Stat.*, Vol. 33, 1065–1076.

Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. San Mateo, CA: Morgan Kaufmann.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, Vol. 65, 386–408.

Rumelhart, D. E., and McLelland, J. L. (1986). *Parallel distributed processing*. Cambridge, MA: MIT Press.

Theodoridis, S., and Koutroumbas, K. (1998). *Pattern recognition*, San Diego, CA: Academic Press.

Vapnik, N. V. (1998). *Statistical learning theory*. New York: John Wiley & Sons.

Werbos, P. J. (1974). Beyond regression: New tools for prediction and analysis in the behavioral sciences. Ph.D. Thesis, Harvard University, Cambridge, MA.

Zadeh, L. H. (1965). Fuzzy sets. *Information and control*, Vol. 8, 338–353.

Zurada, J. M. (1992). *Introduction to artificial neural systems*. West Publishing Company.

People, Information Systems Impact on

Pedro David Pérez
Cornell University

GLOSSARY

blog (short for Web-log) Frequently updated Web pages consisting of brief, dated entries, with new ones pushing the old ones to the bottom of the page. The entries tend to be either hyperlinks to other Websites (with commentary) or personal diaries.

chatting Synchronous communication between people using the Internet at the same time. While the most used form of chatting is the writing of short messages (as in a written conversation), it is possible to chat in a teleconferencing environment (given the necessary equipment and bandwidth).

Gilder's law Bandwidth grows at least three times faster than computer power. While computer power doubles every eighteen months (Moore's law), communications power doubles every 6 months. (George Gilder, *Telecosm,* 2000).

Kondratieff cycles Long term cycles of economic activity, named after the Russian economist Nikolai Kondratieff (1892–1938). Joseph Schumpeter (1883–1950) observed that they coincide with outbursts of technological innovation. Many economists of technology agree that we are at the beginning of a new Kondratieff cycle, brought by information technology.

last mile problem Refers to the difficulties inherent in connecting every individual on a given set (earth, country, region, city) through a full network.

open source software Software that is publicly available to read, use, redistribute, and modify. The open source philosophy states that such software evolves to become "bigger and better" faster than copyrighted software can. The best example of open source software is the "Linux" operative system.

peer-to-peer networks Internet networks that allow a group of computer users with the same networking program to connect with each other and directly access files from one another's hard drives.

spamming Sending of unsolicited messages of propagandistic or advertising nature to lists of e-mail addresses.

virtual communities Social aggregations that emerge from the Internet when enough people carry on public discussions long enough and with sufficient human feeling to form webs of personal relationships in cyberspace (Howard Rheigold, *The Virtual Community,* 1993).

Web-enabling The replication of the information flows of an organization on the Internet.

In the beginning was the word. Our ancestors realized that communication helped them to survive, improve their lot, and make their interaction richer. So attention was paid to improve it. The results: writing, early post services, printing. Then the Industrial and Scientific Revolution brought things to a head: telegraph,

massive post services, wireless, telephones, cinema, wireless radio, television, satellites, computers. Each one of these devices made the interval between sending and receiving shorter, the potential distance between sender and receiver longer, and the number of people involved in a given communication greater.

We now witness the emergence of a new communication technology with the potential to engulf all previous communication technologies within itself: the Internet and World Wide Web, a worldwide network of computers that interact with each other using standardized communication protocols. This survey attempts to give a wide perspective of the current and likely effects of the emergence of the Internet as the communication substrate of humankind.

Nua Internet Surveys estimate that there were 407.1 million people connected to the Internet in November 2000. They are not uniformly distributed around the world: 167.12 million are connected in Canada and the United States (about a quarter of the population), while Africa shows only 3.1 million people connected, less than 1% of its population. However, the most important datum is the speed of growth. In 1995, Nua estimated 25 million users worldwide. So the number of users worldwide has doubled every 18 months. These numbers are expected to mount until the majority of the earth population is connected.

In January 2001, according to data posted by the Internet Software Consortium there were 109,574,429 servers that held documents and data for users of the Internet to access.[1] There are approximately four users for each server on the Internet. This ratio is bound to decline as more and more client computers become servers in their own right, and a peer-to-peer paradigm evolves.

While the Internet offers the ability to replicate almost all the previous modes of information exchange developed by humankind, its usage has been centered on two capabilities: electronic mail, or e-mail, and web sites. Electronic mail is a method of sending formatted messages and files over the Internet. A web site is a location on the World Wide Web that is identified by a web address. Web sites consist of pages of information and data coded as to be readable by a web browser such as Netscape or Microsoft Explorer.

Web sites can be static or dynamic. Static web sites are like a catalog. If the intent of the web site is to display and share information, static web sites offer ease of construction and retrieval. If organized properly (i.e., appropriate indexes, FAQs, a good site map) the ease of usage can also be an incentive for visitors.

Beyond a certain size (i.e., a library) even a well organized site becomes unwieldy. Also, static web sites do not allow for the exchange of information (orders, payments, instructions for delivery and changes in inventory status) required for commerce to happen over the Web. Dynamic web sites allow for interaction, whether within the web site pages or with other web sites and people (customer service representatives, other users). A dynamic web site can be a powerful tool for both visitors and hosts. And, as web technology becomes more sophisticated and easier to use, the differences in feel and ease of use between both types of site are blurred.

I. COMMUNICATING ON THE INTERNET

Electronic mail, or e-mail, is the capability that launched the Internet into wide acceptance. It has become ubiquitous: An employee in a business or government organization can spend up to 2 hours daily answering an average of 30 e-mails and the trend is for steep increases in the number of messages sent and received.

Some of the salient features of e-mail are:

- Sending and retrieval of messages is almost instantaneous.
- While paperless, it leaves a document trail that can be stored, organized, and retrieved.
- Allows for effortless replication of messages and receivers.
- It is as ubiquitous as the telephone grid, including wireless service.
- The cost per word and message sent is very low.
- Allows for attachment and sending of electronic files (including multimedia).

E-mail is an emerging technology, and it remains relatively unfinished. Some weaknesses are:

- While the technology per se can protect privacy, the nature of e-mail makes personal communications potentially available to the whole world.
- E-mail is the contagion mode for computer viruses.
- E-mail allows for spamming, the sending of unsolicited messages, propaganda, and advertising to lists of e-mail addresses. The equivalent of junk

[1] A server is a combination of hardware (computer machines) and software (computer programs) that controls a network.

mail on the Internet, spam is the more annoying because of the rudimentary e-mail management capabilities currently available to users.

E-mail is mostly used as a personal, one-to-one communication medium. In that sense, it is a substitute for the telephone,[2] as much as the telephone was a substitute for the letter. The Internet also offers ways to communicate with multiple people. The most direct is the multiple address capability of e-mail, which has led to the evolution of newsgroups, discussion groups dedicated to very specific areas of interest. Each question, answer, comment, and news that any member considers noteworthy can be addressed to the whole group. Other manifestations of the same phenomenon are mailing lists (groups that receive messages from specific organizations or interest groups) and electronic bulletin boards.

Reductions in the price of web cams (cameras connected to the Internet) have made teleconferencing an option for individuals and small businesses. Widespread use of these capabilities requires fast connections to the Internet, but the potential is great. For example, use of Internet-based videoconferencing capabilities is already helping with the health care and monitoring of patients with special needs requiring 24-hour supervision. The system allows for automatic access to physicians and other medical personnel, who can respond to a crisis immediately and chart a course of action in case the crisis does not resolve itself. Because of being based on the Internet, the system can also be used to transfer data such as charts, MRIs, and test results. The Internet enables use of the videophone, with the benefit of supporting data transfer. Computers and information technology already have profoundly transformed medicine: think of body imaging applications such as computer-assisted tomography (CAT) scanners or medical lasers. But cybermedicine (medical practice as mediated by the Internet) could be a qualitative step forward for medicine.

- Telecare, as exemplified above, could make prevention, diagnosis, and treatment easier and more personal. It also will allow for decentralization of the hospital environment.
- The availability of complete patient information for downloading by any physician who requires it when she requires it (including capabilities for adequate organization of the data, dynamic updating and display, and descriptive analysis) could make intervention more timely and precise. In addition, teleconferencing could involve specialists at the very moment their participation is needed.
- The availability of accurate and updated medical information on the Internet (i.e., Medscape) will allow for more informed dialog among professionals and between physician and patient.

In addition to being a communication-enabling environment, the Internet/World Wide Web is a unique publishing device:

- It can potentially reach every human being.
- It is a mass medium that allows for self-expression.

The most common medium of self-expression on the Internet is the personal web page, an on-line multimedia text addressing the question: "Who am I?" Because of the nature of the medium, it is easy to collect materials from other web sites and create a "home page" that reflects the image that the author has of him- or herself. Personal web pages tend to be updated often, with additional items (photos or other multimedia material, writings, designs) and with hyperlinks to other web sites. In that way, they become rather like a personal diary, although one that is open to the whole world. This characteristic of personal web pages is emphasized in so-called "blogs" (short for web logs), a frequently updated web page consisting of brief, dated entries, with new ones pushing the old ones to the bottom of the page. These entries tend to be both hyperlinks to other web sites (with commentaries) and personal diaries. Blogging tools make maintenance of the blog easy. (To create and maintain an attractive personal web page the creator has to be proficient in programming languages such as HTML and Java.) Blogging is making personal web pages at the same time more intimate and more attractive to read and follow, so bloggers tend to become "virtual" groups.

II. GATHERING INFORMATION ON THE INTERNET

Besides communication proper, the number of activities that a person can accomplish on the Internet is expanding rapidly, but most of them have to do with

[2] Internet telephony is a growing field. As long as two devices (computers, cell phones, etc.) that can connect with the Internet are turned on at the same time, it is possible to connect the two as if replicating a phone call. A phone call can be seen, therefore, as a *synchronous* e-mail, and e-mail can be seen as an *asynchronous* phone call. It is also possible to mimic a synchronous telephone call by using *chatting* capabilities, which allow the users to write messages to each other in real time, as if in a (written) conversation.

searching for information and *collecting information*. The Internet has redefined these activities.

Consider a search for information (say, about a small city in India). Before the Internet became widely available, the searcher would have started by consulting an encyclopedia such as the *Encyclopedia Britannica*. After finding the basic facts, the searcher would have had the option of going to the local library and hope to find additional information available there. Alternatively, she could write to the relevant authority (in this case, an Indian consulate or the mayor of the town itself), hoping the person would provide some leaflets. Or she could search for other sources within the community (professors, travel agents, Indian citizens) who may have either been there or have access to additional information.

Nowadays, someone wishing to find information on the same small town can start by accessing the Internet and conducting a search on any one of the search engines available to her. As a result, she might find:

- Corporate web sites for the city itself and for its main institutions and businesses.
- Personal web sites of individual citizens both living there and abroad.
- Travel-oriented web sites indicating what is interesting there and how to get there.
- Other sites (including *Encyclopedia Britannica*) detailing its geography and history.

In addition, hyperlinks to other relevant web sites and e-mail addresses of individuals and institutions may be found that are useful in either providing more information or in arranging a stay.

An important additional element is the way in which the Internet lends itself to information sharing. An example of this phenomenon has been the explosion of Internet-based sharing of music. Sites such as Napster.com, gnutella.wego.com, and scour.com have made Web-based sharing of music files easy. In effect, their software converts personal computers into miniservers, retrieving information from them for others to download. While the capabilities of each system differ, the technology has made it possible for the contents of all memory devices in all connected computers to be available to everyone connected to the Internet.

The specific use of this peer-to-peer capability for teenage music lovers (most of the users of these services) is that they are able to load tunes onto their computer hard drive and share them with the whole set of users of Napster or any other of the tools. This has been seen by the music industry as a deadly threat to its current business model, and companies have attempted to close these sites under copyright law (with success in the case of Napster). But, with a customer base of about 200 million users for Napster alone, it is clear that peer-to-peer models of information exchange are here to stay.

III. CONDUCTING BUSINESS ON THE INTERNET

One of the most visible effects of the Internet in everyday life is the appearance of alternatives for buying goods there, or "shopping the Net." People can use the Internet to:

- Obtain information about products (including prices).
- Buy products (and have them delivered to their homes).

The first activity is more prevalent than the second. However, it is expected that e-commerce revenues will reach the $100 billion mark in the United States alone by 2002. The preferred transactions have involved computer hardware and software, travel (air tickets, packages, etc), and collectibles. There has also been an explosion in the use of financial brokerages, leading to the phenomenon of "day trading" (direct stock-market activity by nonprofessional traders).

IV. THE NEW ECONOMICS OF INFORMATION

Suppose a user, sitting through a web-surfing session, is looking for a car. She will visit mostly static sites (such as edmund.com) to gather information about different makes and models. She might stop by a chat room to converse with other people also looking for cars. Eventually, she has to decide whether to take all the information she has collected and go to a dealer, or to just proceed to purchase the vehicle over the Internet. In the second case, she can visit the web sites of dealers (even using web-based phone capabilities to talk to customer representatives) or do a search using web crawlers that scour dealer web sites for prices, performance, or any other desired product or dealer characteristics. Alternatively, she could enter a bid for a car being auctioned at a site such as eBay, or enter a bid with the price she would be willing to pay for a given car in a site such as priceline.com.

The description above emphasizes the two main goods being offered by sites on the Internet: information and commerce. The Internet is about infor-

mation posted on it and developed within it; it is the repository of billions of conversations recorded in chat rooms and formal threads. (A *thread* is the trail of responses to a message posted in any Internet-based forum). Information incorporated into the Internet is quickly and effortlessly copied, swapped, and stored. The amount of information and the speed of retrieval available to people with access to computers and phone lines have shown a qualitative increase in the last 10 years, and the economic value of information has dropped accordingly. *Encyclopedia Britannica* used to cost $2000. Its contents are now freely available on the Internet. (However, to use search and other data organization capabilities, a user needs to subscribe to the site.)

The ease with which information is stored and reproduced within the Internet presents both opportunities and challenges for information providers, entities that generate information in order to obtain a given result. On the one hand, the Internet presents the promise of almost instantaneous access to information from anywhere in the world and of reaction and ongoing conversation about this information. This is a great opportunity for government agencies, nongovernmental organizations, and any other not-for-profit entities that want to reach a maximum audience with their message. If a message is posted on a web site, it is potentially available to anyone anywhere in the world.

On the other hand, information providers who want to obtain a profit from the sale of that information need to conform to a new set of rules. Basically there are two ways to add value to information to make it worth paying for on the Internet: Either provide real-time information that is of great value to a segment of users (i.e., real-time stock market data and analysis) or be the portal to a given set of data (i.e., the archives of *The New York Times*), usefully organized for a given segment of users. A user pays for either timeliness or organization, both value added. If an information provider conforms to neither of these models (i.e., CNN), then it has to rely on enough revenue from advertisers (the TV network model of income).

V. E-COMMERCE

What if the Internet is used as a means to market, sell, and organize the distribution of material goods? This is the realm of e-commerce. From nil just a few years ago, Christmas sales alone for 2000 over the Internet reached $10 billion. While some of the growth is due to the more obvious characteristics of the Internet (easy access, instantaneous shopping and purchase, convenience), it is helped by techniques and possibilities that are unique to the new medium, such as auctions, instant price comparison, dynamic pricing, and customer requirement management.

It is useful to think of e-commerce in terms of the interaction between businesses and consumers, as per the following matrix:

	Business	Consumer
Business	B2B (e.g., Covisint)	B2C (e.g., Amazon)
Consumer	C2B (e.g., Priceline)	C2C (e.g., eBay)

Amazon.com epitomizes the dot.com retailer. It is a simple concept: an electronic storefront/catalog backed up by good logistic capabilities. The details of the implementation of the concept put Amazon.com on the forefront of e-commerce:

- Large investments in warehouse distribution centers allow Amazon.com to react quickly to its customers' orders.
- A streamlined, easy-to-navigate site allows for fast purchases.
- Amazon.com uses its transactions database to generate information useful for customers.
- Amazon.com allows customers to write reviews of any of the articles it sells, and leaves them there: good, bad, and indifferent.

The result is a retail experience like no other, and an appealing one: two-thirds of Amazon's sales come from repeat customers.

EBay started as a site for buyers and sellers of odd items: a fleamarket auction site. However, it has become apparent that the Internet is an ideal location for auction markets. The Internet provides wide access (and therefore a wide market), it allows for communication between buyers and sellers, and it does so over as extended a period of time as the seller may be willing to countenance. The winning characteristics of eBay as an enterprise are:

- The minimal resources it needs to organize its business (just a cut of the transactions it hosts is enough to provide a handsome profit).
- The appeal of its integrated databases of offerings
- The sense of community that it has been able to foster among its users.

Priceline.com has taken a different approach. In its original form, a customer named his price for a

flight, and airlines would either accept or reject the bid. This is known as a *reverse auction,* and it is a way to match unused capacity or excess inventory with unmet market demand *without* changing the revealed price structure for the goods and services in question.

In February 2001, both eBay and Amazon had a bright future ahead of them. The annual revenues of Amazon have climbed to $3 billion, and the company is capping its expansion phase to concentrate on becoming self-financing. EBay has been always self-financing (its revenue comes from a take on every transaction completed in the site), and is able to generate annual profits of $50 million. Priceline.com, on the other hand, seems to have failed the market test, with stock valued at less than $3 per share (as against $162 in April 1999).

VI. ORGANIZING ON THE INTERNET

The Internet, its communication possibilities, and e-commerce have also created new models for organizing. Three of them deserve closer examination:

- Web enabling
- Open source software (and other content) development
- Political activism

A web-enabled company is one that replicates its information flows over the Internet. Doing this, while not straightforward, allows an organization a great deal of flexibility and reliability in its operations. For example, Cisco Corp. claims that most of its sales and customer support is done over the Internet, using its sophisticated web sites and its link with information systems within Cisco.

Another example is e-mail, which allows for fast communication without interrupting the specific work at hand. It also allows for constant streams of information being passed among employees at the corporate site and employees visiting customers, suppliers, or other sites. E-mail has enabled telework, the ability to perform a job from locations other than the worksite itself. A side effect of telework, however, is an increase in hours worked by the employee. Using the Internet to organize work seems to blur professional and personal barriers.

The opportunities for the emergence of self-organized groups opened by the Internet are compelling. A major paradigm is the open sourcing model. Originally evolved from software development, it proposes that a free-flowing input of ideas from a myriad of observers using Internet-based capabilities will yield better results than a closed organization trying to achieve the same goal. This principle was used by Linus Torvald, a Finnish software developer, to create Linux, an open, free Unix-based operating system that is the preferred application for web servers. In the words of Eric Raymond: "Linux is subversive. Who would have thought . . . that a world-class operating system could coalesce as if by magic out of part-time hacking (programming) by several thousand developers scattered all over the planet, connected only by the tenuous strands of the Internet? . . . Linus Torvalds's style of development—release early and often, delegate everything you can, be open to the point of promiscuity—came as a surprise. . . ."

Echoes of the open sourcing paradigm can be heard in the political activism of the swarm of grassroots groups opposing tendencies toward economic globalization and coordination. These groups, using the Internet and wireless telephony, have been able to stage very visible and well-coordinated demonstrations whenever international organizations dealing with international trade and economics gather (most notably, the International Monetary Fund and World Bank meeting in Seattle in 1999). The Internet has helped bring together disparate small groups toward a specific goal, coordinate them, and create opportunities both for effective protest and for visibility. In many cases, these activities have been successful. For example, the World Bank is making active efforts to integrate parts of the activist platforms into its programs.

VII. SATISFYING HUMAN NEEDS AND WANTS IN THE INFORMATION AGE

To understand the impact of the Internet on society at large it helps to think of previous cycles of technological innovation. In the early 20th century, the Russian economist Nikolai Kondratieff (1892–1938) suggested that market capitalist societies undergo long-term cycles of economic activity. Joseph Schumpeter (1883–1950) pointed out that these long-term cycles of economic activity coincide with outbursts of technological invention. Since the Industrial Revolution, there have been four such cycles of major technological innovation activity:

- In the first half of the 19th century, a cycle characterized by the steam engine and the mechanized loom, with cotton as the enabling economic factor
- In the second half of the 19th century, a cycle characterized by the mastery of iron and railroads, with coal as the enabling economic factor

- In the first half of the 20th century, a cycle characterized by the mastery of chemistry and steel, with electricity as the enabling economic factor
- In the second half of the 20th century, a cycle characterized by the mastery of mass production, with cheap energy (notably petroleum) as the enabling economic factor

In the beginning of the 21st century, we are entering a new Kondratieff cycle, characterized by the mastery of information technology, in terms of information processing and distribution capability, with information itself as the enabling economic factor. The progressive increases in production and productivity accrued during the first four Kondratieff cycles have yielded the potential to satisfy basic human needs (food, shelter, and physical well-being). The fifth Kondratieff cycle is about subtler human needs. In terms of Abraham Maslow's hierarchy of needs, the physiological needs of human beings can be satisfied, and are being satisfied, within our current levels of technology and productivity. So it is imperative to satisfy the remaining needs in the hierarchy: safety, belonging, esteem, and self-actualization. But safety needs have always been tied to physiological needs, on the individual level, and to belonging and esteem needs, on the group level. Therefore, the information technology revolution that we are witnessing is the first global attempt of the market capitalist economy to address the belonging and esteem needs of human beings—and this is the promise of the information age.

It will require new forms of productive and social organization. Before the Industrial Revolution, productive activity concentrated either around agriculture and animal tending (the great majority), or it took the form of craftmanship, artisans producing the goods of daily life. After the Industrial Revolution came mass production, in which machines in tightly coordinated organizations put together the goods of daily life on an unprecedented scale. Mass production has served humankind well. From being victims of nature, human beings became masters of it. And a first glimpse of a life beyond "nasty, brutish and short" was achieved.

But mass production has reached its limits. Markets have become saturated. The kind of scientific knowledge that can be turned into commercial technology is becoming more and more expensive, thus requiring organizations (even competitors) to collaborate. And the mass production model, predicated on economies of scale (the higher the production capacity of a plant, the cheaper the products), has reached its limits. This is because hierarchies (the form of organization naturally associated with mass production) are rigid, do not handle information well, and are not empowering. Even the just-in-time and continuous improvement techniques of Japanese industry, while successful in perfecting the model, could not overcome its limitations. Information technology and its Internet and wireless embodiments have effected a revolution that takes productive activity into a post-industrial environment. In terms of strategy, the original three functions in business (operations, marketing and finance) are becoming even more associated with a set of activities:

- Operations with infrastructure activities (including, but going beyond, capital projects, manufacturing, and logistics)
- Marketing with customer maintenance and market creation activities (including, but going beyond, data mining, customer requirements management, and dynamic pricing)
- Finance with knowledge and innovation generation (including, but going beyond, innovation portfolios and venture capital)

The preferred business form is the *horizontal corporation,* characterized by Castells as "organization around processes, not tasks; a flat hierarchy; team management; team performance; maximization of contacts with suppliers and customers; information, training, and retraining of employees at all levels." To obtain its goals it engages in flexible production (notably reengineering and computer-aided design and manufacturing), it becomes smaller than the traditional mass production organization, it engages in networking with other firms, and it establishes formal strategic alliances.

The combination of the horizontal corporation with information technology (which, in the case of web enabling, implies a complete mapping of the internalized flows of the corporation onto the Internet, with adequate connection points for external agents to participate) makes the *network enterprise* possible according to Castells: "the network enterprise . . . transform (market) signals into commodities by processing knowledge."

The structure of business organizations will be mirrored in the relationship between individuals and business. Humankind is entering the era of the network society: a society of interconnected individual participants (people, organizations, resources, processes, ideas). Because of the ease of exchange of information and communication within the Internet, there is no compelling need to physically group: virtual grouping is feasible and satisfying. The need to

differentiate against the others that surround us is fulfilled by communing with the ones far away. And the flexibility of networks allows participants (people, organizations, resources, processes, ideas) to quickly converge on opportunities, promoting innovation and dynamism. The result, again according to Castells, is "a capitalist economy based on innovation, globalization, and decentralized concentration; works, workers, and firms based on flexibility and adaptability, a culture of endless deconstruction and reconstruction; a polity geared towards the instant processing of new values and public moods; and a social organization aiming at the supersession of space and the annihilation of time."

How will individual lives change? Timothy Leary defined the *cyberpunks* as "the inventors, innovative writers, techno-frontier artists, risk-taking film directors, icon-shifting composers, expressionist artists, free-agent scientists, innovative show-biz entrepreneurs, techno-creatives, computer visionaries, elegant hackers, bit-blipping Prolog adepts, special-effectives, video wizards, neurological test pilots, media explorers—all who boldly package and steer ideas out there where no thoughts have ever gone before." In the popular imagination these characters have an adventurous, almost illegal aura, witness the hacker community. Hackers are expert programmers and networking wizards who are at the "bleeding edge" of technology. They are responsible for Linux and the Internet. However, in the public mind hackers are criminals that break into other people's "virtual" property.

Will we all become cyberpunks and hackers? Most likely no, although a degree of expertise with software will be required for professionals who want to become true innovators. However, the network society and the Internet/information technology substratum that supports it is already changing the relationship of the individual with the others and with society. Take economic relationships. Phenomena such as e-commerce, Internet-based auctions and procurement, day trading, Internet banking, and the emergence of job-search web sites suggest that each of us will become a fully enabled economic unit in our own right, participating (and competing) in financial, commodities, and human resources and capital markets:

- The scarcest resource in our economy is human capital, people with specific skills, knowledge, and abilities that are able to turn information and resources into value. Davis and Meyer believe that "The markets of the connected [networked] economy are too impatient, too well informed, and too greedy" not to turn such scarcity into markets. Thus, in an economy where all organizations and individuals have mirror images on the Web, it will be possible to accurately price individual abilities and to match them with opportunities. The consequences will be profound, as we will see our community (professional, scholarly, of interests, of skills) as our organization, and most work outside of infrastructure running and maintenance will be done in project teams that bring together the "optimal" group of talents available for a given reward.
- An individual involved in such tailored projects will invest her rewards (whether monetary or financial terms) in an array of financial options. These will surely include stocks and bonds (but the nature of those vehicles may be significantly different from today), as well as real estate and other material needs. They may also include reinvestment into one's own human capital, such as "paying" for taking jobs that we are not qualified for.
- Finally, the accurate picture of the individual that will develop as a result of one's connection with the Internet will allow marketers to target products very specifically, at a very satisfying premium. Or, we may contract for streams of goods, like fresh produce, and let the almost perfect markets enabled by information technology aggregate the contracts into an investment in a specific kind of farm.

VIII. THE PRIVATE SPHERE IN A WORLD OF VIRTUAL COMMUNITIES

The influence of the Internet will be felt most keenly, however, in the personal sphere. Envision a weekday evening, 40 years ago. After supper, the family would concentrate around the TV set, to watch the news and sitcoms offered to them by a handful of networks. (If you lived outside the United States, chances were you had only one choice.) This was classic one-way mass communication: someone prepared the contents and transmitted them for the masses to consume. Competition was minimal, and the scarcity of TV sets ensured that no deviant tastes (teenagers and oddballs) disturbed the uniformity of the medium.

Now move ahead 25 years. You will find multiple TV sets, and an assortment of consumer products (stereo equipment, early personal computers, and gaming consoles, VCRs), all struggling for our attention. Cable technology brings an assortment of TV

channels no one could have dreamt about in 1960. And, if none of the offerings is satisfying, there is always the option of listening to music, or slipping in a videocassette to watch a movie, or just pick up the phone (by now, there are multiple phone connections available to a household).

The Internet (and wireless telephony) has brought these developments to a head: no more mass media; everyone can be both a producer and recipient of content; and complete interactivity. Internet-based gaming, with its rapidly improving multimedia and communication capabilities, could well become the next wave of technology-supported entertainment, succeeding the movies, radio, and television. Hollywood and network offerings, with their fine-tuned production values and established product appeal, will still be popular. But they will not rule the roost. None of the content providers will rule the roost. Instead, loose associations of recipients of content *(virtual communities)* will, and the genius of the content providers will be to gather them behind their products.

Virtual (or electronic) communities are groups of people with similar interests who use the Internet to communicate and collaborate. We have already mentioned some, ie. open source coding communities and political activism. The above list towers like an iceberg, but much is under water. Virtual communities go from the unsurprising (expatriate e-mail lists and discussion groups) to the specialized (i.e., singing technique for Baroque music) to the unique (i.e., the three followers of Northern South American plains music in the United States). An accepted classification includes communities of transaction (something eBay excels at), communities of interest, communities of relations (gathered together by common experiences), and gaming communities.

Internet and wireless technology is evolving in such a way as to make virtual communities easier to organize. Personal web sites and blogs have been mentioned already. Another development is MUDs, or multi-user dungeons. MUDs allow participants to interact within the framework of a shared reality. Their success points to another attraction of virtual communities: in a world where the distinction between home and work is fast blurring (in part because of the same technologies), they provide a "third way," a place independent of home and work where one can manifest one's own personality, interests, and feelings.

All this points to a disturbing possibility raised by Castells. As more and more social and personal communication happens over the Net, with its inherent distribution and memory capabilities, the vanguard of social and cultural development may become tightly intertwined with the Internet. This will happen in a context of widespread social and cultural differentiation, which will add to the cognitive complexity required to participate in the constellation of virtual communities and be both a contributor as well as a receiver. If we add to this the sizable effort that is already being put into harnessing nascent virtual communities and creating new ones for power and profit, it is clear that many people may not have the preparation, or the inclination, to interact. Then, they will be interacted; in Castells' words: "the coexistence of a customized mass media culture and an interactive electronic communication network of self-selected communes."[3]

This does not mean that place has become irrelevant, but that its relevance has changed. The place of choice is becoming the node where flows of information and resources cross. And, in that privileged center, it is possible to bathe in information, to drown in it, to use it for the creation of an environment that is to the information age what the fountains of Rome and the swimming pools of Los Angeles were to the cities that grew around land and water. In those centers (New York, the California metropolises, but also Bangalore, Hyderabad, Shanghai, etc.) information, accessed globally, defines activity, expectations, and eventually will define lifestyle.

These points of confluence of information may create virtual realities. "Virtual realities" are electronically mediated environments, where "intelligent" devices react to our actions and commands to provide feedback, information, and experiences. While the extreme manifestation of virtual reality is complete "substitution" of the physical reality for the reality of cyberspace, there is no reason why the final relationship between physical space and cyberspace may not be more subtle and complementary than violent dichotomy. So, if today we see cyberspace as through a window, and if "extreme" cyberspace is our going through that window, the most likely possibility is a melding, progressively more seamless, of physical space and cyberspace. Still, a fuller immersion into cyberspace will be available to those who want to partake of the "third place," as today MUDs allow for.

Virtual reality will profoundly affect artistic expression. Computers have already been used extensively in the arts: Think of music synthesizers, or of *Toy Story,* or of the steady influx of visual material generated in computers. The emergence of virtual realities will give

[3] This is helped by the technological asymmetry of ease of downlink and difficulty of uplink. It is easier to download information from the Internet than it is to send information into the Internet.

these artistic manifestations a unique platform, as they will be seamlessly integrated into a constructed world, where electronic music *is* the music, digital images *are* the images, and whole worlds are up for creative grabs. This art may evolve toward a more community oriented, almost tribal manifestation, as virtual communities develop their own worlds, design avatars to populate them, and create artistic heritages that reflect their chosen identities.

What does this mean for social interaction? According to a study at Carnegie Mellon University by Kraut et al., "Greater use of the Internet [is] associated with small, but statistically significant declines in social involvement as measured by communication within the family and the size of people's local social networks, and with increases in loneliness, a psychological state associated with social involvement. Greater use of the Internet [is] also associated with increases in depression." These are effects of the Internet in its current manifestation. Will virtual reality further alienate users (some already talk of Internet addiction disorder)? Or is the act of entering the scene a liberating one? There are no clear answers to these questions at this moment.

IX. COMPUTER CRIME, SECURITY, AND PRIVACY

According to the Privacy Rights Clearinghouse, in 2000 there were about 700,000 cases of identity theft in the United States (against 35,000 in 1992). An identity thief (sometimes a computer hacker but, much more often, a dishonest employee) steals the Social Security number of an individual, along with personal information associated with it, and then proceeds to activate new credit cards, rent apartments, obtain legal documents, etc. When the fraud is exposed, it leaves the victim with a legal mess and a damaged reputation.

The above is an example of computer crime. The nature of databases on the Web and of electronic commerce makes it significantly easier to steal identities and profit from it. Computer crime goes from the cat-and-mouse games of hackers and network managers to the unleashing of computer viruses and targeted attacks on busy sites. It also includes violations of copyright, made easy by the information-rich nature of the Internet. And a rash of pornographic and hate sites has brought about heated discussions on the nature of free speech and decency.

Data theft of the kind suggested by identity theft is the foremost threat to e-commerce. Organizations need to protect their transaction data as well as their internal communication and documents. To do so, they use encryption, digital signatures and certificates, and firewalls. *Encryption* is a process of making messages indecipherable except by those who have an authorized decryption key. A *digital signature* or *certificate* is a phrase or document encrypted within a communication between two networks that confirms the identity of the sender. A *firewall* is a server that isolates a private network from a public network. The objective of these security measures is to protect the integrity and privacy of internal data and the authenticity, privacy, integrity, and identity of a transaction.

On the other hand, the very information that companies hoard may be put to uses that can be seen as infringing privacy and fairness. In October 2000, Amazon.com attempted to use dynamic pricing. The objective of dynamic pricing strategies is to charge each customer as much for a product as he is willing to pay. There are open techniques (such as auctions) and strategies (such as reductions in prices) that are accepted by the public. The amount of data collected by retailers through information technology also may allow for analysis of consumer behavior that determines how much he is willing to pay for a product *a priori* (the approach that Amazon.com seems to have followed). After public outcry, Amazon.com stopped its experiment.

This is an example of a general tendency toward less privacy. As computers become more and more pervasive and involved with our daily activities, and as the ability of computers to gather, store, transmit, retrieve and process information increases, it becomes easier to track anybody's daily activities. This provides huge potential to infringe "the right to be left alone."[4] Living our lives within the mesh of the Internet results in a life lived in public. This brings two questions: (1) Why should others have access to my life? (2) What are these others allowed to do with the information that they are bound to obtain from the Internet? The answers are not clear. For example, the electronic trail can be of tremendous value for criminal investigations. However, how much do you trust the government, anyhow? In the same vein, where does the voicing of opinions within an interest group become expression that damages society? The questions posed by this web enabling of our life will remain with us for the foreseeable future. Better encryption and security technology will surely help, as will as laws more tightly focused on the problem. More comprehensive

[4] ". . . the right to be let alone is the most comprehensive of rights and the right most valued by civilized men" (Justice Louis Brandeis, 1928).

solutions have been suggested. Hagel and Singer, in their book *Net Worth* suggest that people should get paid for the use of their data. And David Brin, in *The Transparent Society,* suggests that the protection of privacy is the wrong approach, and that, instead, societies should focus on the protection of access.

X. THE INTERNET: REQUIREMENTS FOR AND POSSIBLE OBSTACLES TO ITS DIFFUSION

All that has been presented so far is predicated on three events:

- A rapid expansion of the capability of the telecommunications structure to make real-time multiple-way communication and access to information quasi-instantaneous, easy, and transparent
- A rapid buildup and optimization of Internet-related supply capabilities, including Internet service providers, applications services providers, web-enabling tools, and the logistics infrastructure to support e-commerce
- A broad degree of acceptance of Internet-related activities by the public

With respect to telecommunications capability, George Gilder has issued a ringing proclamation in his book *Telecosm:* Bandwidth grows at least three times faster than computer power. Bandwidth is the amount of data that can pass through a transmission medium, eg. a simple twisted cable of the type commonly used in telephony can carry up to 33.2 Kbps (kilobits per second). On the other hand, optical fiber is expected to reach the Pbps (petabits per second, or 10 to the 15th bits, or approximately 10 to the 12th times the capacity of twisted cables) soon. The significance of Gilder's statement is that, if he is right, there will be no practical limitations to the amount of information that can be sent through the telecommunications networks: The constraints will be at the processing and memory stages and at the availability of content. However, the investment sums required to make this happen are staggering. Therefore, there may be many intermediate steps to reach the goal of total, fiber-to-the-home connectivity:

- Digital subscriber lines (DSL) and asymmetric DSL (ADSL), are options already put in place by telecommunications companies. With ADSL it is possible to reach bandwidths of 9 Mbps downstream and 800 Kbps upstream, which is an improvement of more than two orders of magnitude over current capabilities.
- Alternatively, cable television is already offering Internet connections over its networks of affiliated homes. Here the greatest possible capability is 40 Mbps, slightly better than ADSL. However, since the service is shared across many subscribers, it is possible to confront significant "clogging" of the line during peak times.
- In poor areas or areas with scattered population, cellular telephony will play a significant role. It is also relevant in providing mobile access to the Internet. At this moment, bandwidths of 2.4 Mbps are possible.
- Alternatively (and conjointly), LEO (low-earth orbit) satellites would provide dedicated bandwidth to reach focused clusters of customers and serve cellular telephony "to the curb."

Along with all this, such things as oceanic "fat" optical cables, bigger and better backbones, appropriate optical routing technology, and "curb" technology (solutions to the problem of making broadband available locally) will need to be invented and perfected, laid down and used.

Nonetheless, it seems that there are no overwhelming scientific and technological limitations to a world of "unlimited" bandwidth. The question, then, takes a different form. Can providers of goods, services and information make appropriate use of an unlimited broadband Internet? Will there be the demand for it that justifies building it up in the first place? If either of these questions is answered with even a qualified negative, it is probable that the development of the network will not progress rapidly toward fiber-to-the-home universal connectivity.

Let us assume that, if the demand is there, the supply will build up and providers of goods, services and information will come up to speed. In terms of interaction with customers, the Internet offers both great possibilities and risks. These are the opportunities:

- A sales and marketing channel that is available 24 hours a day, every day, and that has enormous amounts of information to display
- An environment that allows customers to gather easily, replicating such customer loyalty exercises as the physical gatherings of Harley-Davidson riders
- The possibility of extensive recording of every step of transactions between customers and companies

There are four major concerns when assessing how broadly acceptable the Internet is for people: the

fundamentally passive character of the Internet, the "last mile" problem, the asynchronous character of many Internet-based applications, and the issues of trust.

The Internet may act as a barrier between people and organizations. Web sites are basically passive. While they may be made to evolve to reflect the history of transactions of customers, they cannot provide the human interaction that personal contact, even at the telephone, does: the rapid adaptation, the serendipity, the surprise factor. Thus, it is possible that the ease of information handling of the Internet is best put to use with emphasis on personal interaction, the Internet as helping device.

The last mile problem can be posited thus: Is it possible to connect every individual on earth (or at least, a country or a region) through a full network? The answer depends on what kind of network. The advances in telecommunications summarized above suggest that the last mile in providing information will be practicable, although the actual implementation of a fully universal network of "fat" optics fibers may always be impractical. On the other hand, the implementation of logistics networks that replicate the capabilities of the Internet is still a major concern. For example, e-grocers like Peapod struggle with the design and operation of a logistics network that allows delivery of family-sized grocery orders (including perishables) in a fast, reliable, convenient, and cost-effective manner.

The last mile problem interacts with final customer behaviors in ways that present additional obstacles to the widespread adoption of the Internet as a transaction medium. The Internet's current advantage is its asynchronous character. In communication, it allows people to engage in conversation across longer periods of time corresponding to their patterns of activity. (The telephone, on the other hand, requires both people to be synchronized, which may or may not be convenient for either of them). In information exchange, it acts as the ultimate repository of documents, bringing whole libraries to the user (and saving us trips to tangible libraries and archives, some of them of difficult real access.) However, it remains to be seen whether the patterns of interaction associated with purchases and services enjoyment (the *raison d'être* of downtowns and shopping malls) can be advantageously countered by Web-enabled services. Thus, the *synchrony* of the social experience is dear to us, and we may be loathe to let go of it. Another possible obstacle to widespread, wholesale Internet adoption is the issue of trust. Can we trust a person with whom we have never interacted in real time with our activities in society, with our assets, with our education, with our health, with our government?

- Coordination and sharing of activities using the Internet and the attendant telecommunications networks has already shown tremendous success in the case of Linux. However, these are networks where there is a great emphasis on personal trust and ability, established through previous performance. In other words, the use of the Internet as a coordinating device may be predicated on preexisting trust, and may need face-to-face interaction to yield benefits in terms of activities accomplished. (On the other hand, there is pleasure in roaming chat rooms and interactive games in disguise.)
- E-banking is bound to become the predominant form of banking, because of the attendant cost benefits. However, as e-Trade has shown by opening a real store presence, it is probably true that the financial sector will settle into some mixture of "clicks and bricks." In that way, there will always be the option to "talk to someone," something that in matters as delicate as finances is bound to be of importance.
- The argument applies even more fully to health care. While transfer of archival and real-time data, teleconferencing, and even remote interventions will be welcome abilities, there will be emphasis on "personal" touch, knowing doctors and nurses even if they are thousands of miles away. In such a scenario, the physical infrastructure of health care as we know it will remain in place, while the transformation will be in the frequency and accuracy of communication with health professionals.
- Anyone who has downloaded tax documents or visa forms from official web sites is grateful that governments have a web presence. However, the full transition of government to the Internet will confront the problems associated with web enabling, compounded by the unique characteristics of authority in governments; and the need to build the citizenships capacity to respond through the Internet (what has been called e-democracy).
- E-learning is a tantalizing frontier, and one that the business world is actively trying to reach in the guise of "knowledge management." But it is far too early to know whether it is possible for a child or a teenager to obtain a "good" education centered around remote access to information and learning software. Again, there is a good

possibility that a "clicks-and-bricks" compromise will be achieved.

The previous discussion highlights another issue surrounding the diffusion and adoption of the Internet: will there be a *digital divide?* In the most straightforward sense, the digital divide is but the difference between analog and digital transmission, processing and storage. This translates into the relationship between people and society: Some have digital (informational) assets, acquired through education and experience; some do not. The former can insert themselves into the social networks that participate in the "virtual" life parallel to the physical world made possible by the Internet. The latter will "chop wood and carry water" (this is already happening in San Jose, California). In an extreme scenario, it is possible to see sizable percentages of the population not participating in e-learning, e-health, and e-government activities that would be taken for granted by the rest.

This is a good moment to close this survey. Information technology, in the guise of the Internet, does hold the promise of a world where information and communication will never be lacking. It is still too early to know the exact shape of such a world. But steps need to be taken to ensure that it will be a world available to everyone.

SEE ALSO THE FOLLOWING ARTICLES

Computer-Supported Cooperative Work • Crime, Use of Computers in • Digital Goods: An Economic Perspective • Economic Impacts of Information Technology • Electronic Commerce • Human Side of Information, Managing the Systems • Internet, Overview • Linux Operating System • Organizations, Information Systems Impact on • Privacy • Virtual Organizations • Virtual Reality

BIBLIOGRAPHY

Castell, M. (1996–1998). *The information age: Economy, society, and culture,* 3 vols. London: Blackwell.

Chandler, D. (1998). Personal home pages and the construction of identities on the Web. Available at http://www.aber.ac.uk/media/Documents/short/webident.html.

Davis, S., and Meyer, C. (1998). *Blur.* Reading, MA: Perseus Books.

Davis, S., and Meyer, C. (2000). *Future wealth.* Cambridge, MA: Harvard Business School Press.

Gilder, G. (2000). *Telecosm.* New York: The Free Press.

Ira, V. B. (June 2000). Katie's amazing video conferencing system. *The Exceptional Parent,* Vol. 30, No. 6, 40.

Kraut, R., Lundmark, V., Patterson, M., Kiesler, S., Mukopadhyay, T., and Scherlis, W. (September 1998). Internet paradox: A social technology that reduces social involvement and psychological well-being. *American Psychologist,* Vol. 53, No. 9, 1017.

McCaffery, L. (Ed.) (1991). *Storming the reality studio: A casebook of cyberpunk and postmodern science fiction.* Durham, NC: Duke University Press.

Mitchell, W. (1999). *e-topia.* Cambridge, MA: The MIT Press.

Raymond, E. (1999). *The cathedral and the bazaar.* Sebastopol, CA: O'Reilly and Associates.

Tapscott, D. (1995). *The digital economy.* New York: McGraw-Hill.

Tapscott, D. (1998). *Growing up digital.* New York: McGraw-Hill.

Turban, E., Lee, J., King, D., and Chung, H. M. (1999). *Electronic commerce.* Upper Saddle River, NJ: Prentice Hall.

Turban, E., McLean, E., and Wetherbe, J. (1999). *Information technology for management.* New York: Wiley.

Privacy

Jin H. Im
Sacred Heart University

GLOSSARY

anonymizer web site To allow a user to navigate through the Internet without worrying about various forms of information gathering since it disguises the user's identity by using its Web proxy address.
cookie A small text file by a web site on a user's hard drive, which can provide some useful features to a user by recoding a user's web surfing behavior on his hard disk. For future visits, for instance, a cookie file remembers a user's account number and a password.
encryption Scrambling the content of a file or an e-mail sent via the Internet so that a recipient can't read it without an appropriate key to decipher. Decryption is the process of converting encrypted data back into its original form so it can be understood.
remailer program To disguise the identity of a sender of an e-mail or a message to a bulletin site so that a user can protect his privacy from others.
seal program Is used as a way to promote self-regulations. Well-known seal programs are TRUSTe (www.truste.org) and BBBOnLine (www.bbbonline.org).

I. INTRODUCTION

A. Definition

The word "privacy" is defined by the *Webster's Ninth New Collegiate Dictionary* as "the quality of state of being apart from company or observation, freedom from unauthorized intrusion." Though the word privacy is not found in the United States Constitution, the Bill of Rights and subsequent amendments assure a person's privacy. In this information age, privacy should be a person's guaranteed right to remain anonymous at his choice while using various information systems. Thus, a person should reserve his right to determine when, how, and to what extent information on him is revealed to others.

B. Problem: Privacy versus Sharing

Concerns on privacy in various manners have existed for a long time. However, the unprecedented advancement in information technology in these days may put a person's privacy in great jeopardy, thus urging development of effective protection schemes. For instance, data warehousing and data mining make on-line analytic processing possible at reasonable costs by handling a vast amount of data in a short time period. The rapid expansion of the Internet raises a renewed concern on privacy invasion since the so-called "customer profiling" can be done easily on an unprecedented scale. Technologically, every move by a user in the Internet can be traced and recorded so that the data may be used for marketing purposes or something else.

According to the *Business Week's* survey about on-line shoppers in March 2000 (http://businessweek.com/2000/00_12/), on-line shopping grew rapidly. Forty-five percent of the surveyed people said that they did on-line shopping, two times larger than 22% in February 1998; 41% of respondents were concerned

"very," and 37% "somewhat" that the on-line site would send unwanted information based on their personal information. As a whole, 78% expressed their concerns about possible misuse of their personal information. The rate jumped up by 13% from the 65% in February 1999 showing that the alarming signals became louder among on-line shoppers. Further, the survey found that 94% of those who didn't do on-line shopping were concerned about their privacy. Unless their concerns are well addressed, emerging on-line businesses may face serious stumbling blocks.

A web site collects personal information with various reasons. First, it can save a user's time and efforts for repetitive works like rekeying, memorizing key information such as account numbers and passwords, user name and address, a credit card number, etc. Second, it can provide customized services based on a user's preferences. Third, it can provide individual marketing based on a user's consumption patterns that is simply not possible in the off-line world. However, users are concerned about unauthorized collection of personal information, unauthorized using of personal information for any other purpose, and unauthorized sharing of personal information with a third party. Thus, privacy issue are like a double-edged sword that needs a careful balance between protecting a user's privacy and sharing data for convenience and other benefits.

Thus, the key issue is who controls revealing (1) what information about a user, (2) when, (3) why, and (4) to whom. Privacy advocates insist that users should be allowed to make an informed decision on those matters.

II. DATA COLLECTION METHODS

A. User Registration

Users voluntarily give out personal information to web sites in exchange for free goods and services such as e-mail accounts (e.g., www.Hotmail.com), web-hosting services, Internet access, and phone/voicemail/fax services. Users may expect the site to use personal data only for the purposes for which they provide the information. Against their expectation, however, businesses then market this information to advertisers, or in some cases, to anyone else who may want it. As a matter of fact, that is how they can provide free services to users while making money by selling users' information. Users may wonder why they start receiving so much junkmail from all over after signing up on web sites with their personal information.

In one of the first cases of its kind pursued by the government, federal regulators in August 1998 accused GeoCities (http://geocities.yahoo.com) of lying to its Internet customers. The Federal Trade Commission (FTC) said GeoCities promised its customers that it wouldn't disclose their personal information, but still revealed to advertisers and others details that it collected about people on-line, such as their income and marital status. GeoCities, whose array of web sites makes it one of the most popular Internet locations, offers people free e-mail and free space to create personal web pages if they agree to answer questions about themselves. In settling the case with federal regulators, GeoCities agreed to change the way it collects and distributes information about its 2 million customers, especially children under 13.

B. Sharing

Web advertisers like DoubleClick want to send their ads to a focused group of customers. Thus, they are getting information about a user's interests and consumption behaviors by tracking a person's every move in the Web and building a personality profile. Based on that information, they may send more ads for credit cards and shopping to a user, and ads for golfing and traveling to another user. Computer codes called "pings" or "web bugs" count the number of visitors to a site and let companies know about the success of a particular advertising campaign.

A web site may write a cookie on a user's hard drive without the user's knowledge. A cookie is a small text file, which can provide some useful features to a user by recoding a user's web-surfing behavior. For future visits, for instance, a cookie file remembers a user's account number and password. Based on the personal preference in a cookie file, a web site can provide personalized web services such as My.Yahoo.com where a user can setup his own stock portfolio tracking, personalized news and TV program listing, local weather, etc. However, a web site also can keep track of the number of times a user sees a given ad, thus building a user's profiling on some sensitive issues such as medical history or sexual preference. Advertising companies like DoubleClick (www.doubleclick.net) and AdForce (www.adforce.com) use cookies to target ads and measure the ads' effectiveness on behalf of its advertisers and web publishers. Amazon.com explains what information will be captured automatically and saved in a cookie file, as follows:

> Examples of the information we collect and analyze include the Internet protocol (IP) address used to

> connect your computer to the Internet; login; e-mail address; password; computer and connection information such as browser type and version, operating system, and platform; purchase history; the full uniform resource locators (URL) clickstream to, through, and from our web site, including date and time; cookie number; products you viewed or searched for; zShops you visited; your auction history, and phone number used to call our 800 number.

A consumer sued DoubleClick, the largest Internet advertising company, on an allegation that the company planned to merge its on-line database with that of a direct-mail company it bought, thus creating personally identifiable information. The DoubleClick company denied such a plan and made it explicit in its privacy statement by stating, as follows:

> DoubleClick does not collect any personally identifiable information about you, such as your name, address, phone number or e-mail address. DoubleClick does, however, collect nonpersonally identifiable information about you, such as the server your computer is logged onto, your browser type (for example, Netscape or Internet Explorer), and whether you responded to the ad delivered. . . . The nonpersonally identifiable information collected by DoubleClick is used for the purpose of targeting ads and measuring ad effectiveness on behalf of DoubleClick's advertisers and web publishers who specifically request it. . . . nonpersonally identifiable information collected by DoubleClick in the course of ad delivery can be associated with a user's personally identifiable information if that user has agreed to receive personally-tailored ads.

It is not an illegal but flourishing business on the Internet to sell detailed information on a person such as social security number, address, employer, monthly bank or credit card statements, etc. On-line information brokers such as TR Information Services, A.S.A.P. Investigations, and, A1 Trace information are selling all kinds of information on a person including sensitive financial data to anybody who is willing to pay. For instance, an on-line information broker, Docusearch. com (www.docusearch.com), sells a wide array of personal information: (1) locating or identifying anybody's current address or social security number, (2) help to find a telephone number or trace the ownership, (3) reveal hidden criminal history's from county court records to nationwide wants and warrants, (4) discovering pending litigation, and civil recordings, (5) determine ownership, to obtain drivers records or locate a drivers license number, (6) uncover hidden assets or determine a person's financial condition, and (7) property to be searched by name or a specific address for details. Prices for most services listed in the company's site are under $50, thus making detailed, sensitive personal information available to anybody at a modest cost. The company claims:

> No matter where you live in this world; you can now access data about people residing in the United States. This is the information age, and information is power! Controversial? Maybe; but wouldn't you sleep easier knowing a little bit more about a prospective business partner, employee, baby-sitter, neighbor, or significant other?

In contrast to the company's claim, nobody can sleep easily after knowing that anybody at a modest fee can have access to their personal information. Amy Lynn Boyer on October 15, 2000, was stalked and killed by a former classmate who had purchased her social security number for $45 from the docusearch.com. With her social security number, the killer easily had access to her personal information including the address of her work place.

Where do on-line information brokers get or buy such information? Their main information sources are (1) credit bureaus like Equifax and Experian, (2) state governments, and (3) banks and financial service companies. For instance, Illinois state government now makes $10 million a year from the sale of public records from real estate filings to divorces and bankruptcies; and Rhode Island prices its motor vehicle records alone at $9.7 million. In May 2000, a lawsuit was filed in federal court in Los Angeles challenging Yahoo!'s controversial practice of routinely disclosing user information—without prior notice—in response to subpoenas. It is certain that web sites may be legally forced to share their customer information with law enforcement agencies.

C. Snooping

The Internet becomes a convenient tool for time-constrained employees to take care of their personal matters from their office PCs such as shopping, stock trading, gambling, sports and computer games, and viewing pornography. According to a recent survey done by Vault.com, 54% of about 1200 employers reported that they had caught employees browsing web sites that were not related to work. Increasing numbers of employers have started monitoring their employees' web use. According to the American Management Association, at least 45% of employers said that they monitored their employees' phone calls, computer files, or e-mail messages. Employers are in

favor of monitoring to reduce wasting company time, to avoid a hostile environment for women, to protect company secrets, or to avoid jamming their networks with bulk e-mail. Employers' fear of sexual-harassment lawsuits is one of the greatest forces driving the monitoring movement. In 1995, Chevron Corporation paid $2.2 million to female employees who asserted that they had been sexually harassed because of jokes sent through the company's networks. Of course, an employer can set up a filtering system to block employees from visiting certain web sites. But it seems impossible to specify all undesirable sites. For instance, America Online once blocked any discussion sites with a word "breast," but later released the word from its black list upon outpouring complaints from social groups for breast cancer.

As far as employers' right to read e-mail messages or track web sites is concerned, the court takes their side by ruling that employers can do what they want with computers they own in several cases. In the Buourke vs. Nissan Corp case in 1991, the California Court of Appeals dismissed a suit by ex-employees who had allegedly exchanged inappropriate e-mails on the ground that they had agreed to a company policy that e-mail be used for business purpose only. In the Smyth vs. Pillsbury case in 1996, the United States District Court in Philadelphia ruled: "The company's interest in preventing inappropriate and unprofessional comments or even illegal activity over its e-mail system outweighs any privacy interest the employee may have in those comments." Thus, the message is clear. As long as employees are warned that they may be watched, they should not expect any privacy in their on-line interactions. Message Inspector and Websense Enterprise are two widely used software packages for tracking e-mails and Internet use. Message Inspector developed by Elron Software of Burlington, MA, screens out inappropriate terms—as defined by whoever uses it—from incoming and outgoing e-mails. Websense Enterprise offered by Websense of San Diego, CA, blocks access to inappropriate web pages and logs every minute employees spend on each site. A police officer in San Diego was fired after his employer found a file on his computer which showed that he had visited www.whitehouse.com, a pornographic site. However, he claimed that his termination was wrong since he had intended to visit the American president's web site, www.whilehouse.gov, but mistyped by accident. He settled his case for $100,000.

Not only employers but also outside companies do on-line data collection. They argue that the information makes for a better customer experience, not abuse. In early 1999, Intel revealed that every Pentium III processor was assigned a unique ID number so that a web site could use the number in a PC as a form of identification. However, privacy advocates opposed Intel's practice by claiming that a user's web surfing could be tracked based on the number. In April 2000, Intel announced that it would give up the number stamping practice for the next generation of processors. RealNetworks Inc. designed its RealJukebox player to monitor users' listening activities and to send the data back to the company. However, it gave in to pressure from privacy advocates and got rid of that feature in the program. The notorious "ILOVEYOU" virus in May 2000 was designed not only to wipe out all image files and MP3 files on an infected hard disk but also to steal passwords stored in a user's computer and send them to a server to which the virus designers have access. Hackers may do so-called "identity theft" and do harm to some innocent people by falsifying documents with a victim's identity. For instance, Lt. Col. Stevens, 72, of Maryland learned thieves had set up 33 fraudulent financial accounts under his name totaling more than $113,000. In June 16, 2000, America Online acknowledged that hackers compromised some employee accounts and that the accounts were used to gain access to and view details of some personal user accounts such as the user's name and address to credit card numbers used for billing. It is known that the Federal Bureau of Investigation also uses its own e-mail snooping system called Carnivore.

III. PROTECTION SCHEMES

A. Self-Protection by Users

1. Common-Sense-Based Protection

The first line of privacy protection starts from users since it is a cheap, easy, yet effective protection scheme for users to adopt. Users should use their discretion based on their common sense. However, many users are basically clueless by not being aware of effective ways to protect their privacy by themselves. Several organizations for privacy advocacy provide their guidelines for users on how to protect their privacy while using the Internet. The Center for Democracy & Technology (http://opt-out.cdt.org/featured/topteninfo.html) suggests ten ways to protect a user's privacy on-line:

1. Look for privacy policies on the Web
2. Get a separate e-mail account for personal e-mail

3. Teach your kids that giving out personal information on-line is like giving it to strangers
4. Clear your memory cache after browsing
5. Make sure that on-line forms are secure
6. Reject unnecessary cookies
7. Use anonymous remailers
8. Encrypt your e-mail
9. Use anonymizers while browsing
10. Opt-out of third party information sharing

The Electronic Frontier Foundation (EFF) suggests its own 12 ways to protect a user's on-line privacy (http://www.eff.org/pub/Privacy/eff_privacy_top_12.html):

1. Do not reveal personal information inadvertently
2. Turn on cookie notices in your web browsers, and/or use cookie management software.
3. Keep a "clean" e-mail address.
4. Don't reveal personal details to strangers or just-met "friends"
5. Realize you may be monitored at work, avoid sending highly personal e-mail to mailing lists, and keep sensitive files on your home computer.
6. Beware sites that offer some sort of reward or prize in exchange for your contact or other information.
7. Do not reply to spammers, for any reason
8. Be conscious of web security
9. Be conscious of home computer security
10. Examine privacy policies and seals
11. Remember that YOU decide what information about yourself to reveal, when, why, and to whom
12. Use encryption

Version 5 of Microsoft Internet Explorer browser allows a user to have several options regarding cookie files in order to control the security level. A user selects (1) "Tools" from the menu, (2) selects the "Internet Options . . . ," (3) selects the "Security" tab, (4) selects the "Custom Level" button, and (5) scrolls down to the "Cookies" section. A user can choose first whether a cookie file will be allowed to be written on his computer; and next choose three options: (1) Disable, (2) Enable, and (3) Prompt for two different security levels. Certainly, users' caution in revelation of their personal information will be helpful. However, most users do not bother themselves even to read privacy policies in a web site since they do not want to spend their time in that way and/or they do not fully understand even when they do read those policies. Further, only a few, privacy conscious users may dare to learn how to use encryption of their e-mails or how to use cookie management software. Thus, the common-sense-based self-protection by users may not work effectively for many users.

2. Technology-Based Protection

Technically, there are various ways for users to protect themselves from customer profiling or snooping of their e-mails. Largely there are three protection tools available to users: encryption software, remailers, and anonymizers (or web proxies). Encryption software like PGP (Pretty Good Privacy) scrambles the content of a file or an e-mail sent via the Internet so that a recipient can't read it without an appropriate key to decipher. Consumers may use anonymous remailers that strip out a user's real name and e-mail address and replace them with dummy information that makes the sender impossible to track. A remailer program such as Cypherpunk and Mixmaster is used to disguise the identity of a sender of an e-mail or a message to a bulletin site so that a user can protect his privacy from others. A remailer program works in this way. When a user sends out an e-mail or post a message in a news group with a remailer program, the remailer program directs the mail to a remailer site. Then, the remailer site strips off the sender's name and address and assigns a new, disguised identity. It is then sent to an intended destination. When another user replies to the e-mail, it goes through the remailer's site to disguise the sender's identity in the same way. Thus, this remailing process protects both parties. For better protection, a remailer program should be supplemented with an appropriate encryption method.

Consumers can hide their identity by using programs that keep web surfing anonymous, such as Anonymizer (www.anonymizer.com), PrivaSeek Persona (www.privaseek.com), or enonymous advisor (www.enonymous.com). An anonymizer web site like Anonymizer.com allows a user to navigate through the Internet without worrying about various forms of information gathering such as JAVA, JavaScript, and cookies since it disguises the user's identity by using its web proxy address. Software like Freedom (www.freedom.com) from ZeroKnowledge Systems of Montreal can create secret digital identities not to reveal real identities. A Freedom user is allowed to have five separate identities that identify the user while surfing the Internet, thus, protecting the user's real identity. Consumers may use antitracking software like SurfSecret software to erase the various identifying marks made on a hard drive during an Internet

activity. It will clear all the trails left in history, cache, and cookie files.

A new promising technology called P3P (Platform for Privacy Preferences) was being created by the World Wide Web Consortium (W3C), the nonprofit organization in charge of technical standards for the Web. Technically, P3P is a set of standards for data sharing on privacy between a browser and a web site. In June 2000, Microsoft announced that the next versions of its web browser, Internet Explorer, and Windows operating system will support P3P. A P3P-enabled browser will allow a user to specify his preference on privacy such as (1) willingness to visit sites that do user profiling, (2) willingness to visit sites that save cookies on the user's hard drive, and (3) willingness to visit sites that sell user information to a third party. When a user visits a P3P-compliant web site with a P3P-enabled browser, the browser will compare the user's preference with the site's privacy guidelines and inform the user whether the site is safe to visit. The main advantage of the P3P is that it will serve as an early warning system for a user based on his predefined preference. But, one disadvantage is that it can't be used as a blocking or filtering system. Also the P3P can't check whether a web site actually abides by its privacy policy. Thus, it should be supplemented with Internet seal programs such as TRUSTe and BBBOnline for compliance assurance.

B. Self-Regulations by Vendors

1. Policies

A growing number of web sites have been posting their privacy policies in an effort to assure users of their privacy. NetCoalition (www.netcoalition.com), a coalition of six of the Internet's leading companies (America Online, DoubleClick, Inktomi, Lycos, Excite@Home, Yahoo!), promotes responsible industry self-regulation and limited government intervention. It supports providing users with (1) notice about their practices regarding the collection and use of personally identifiable information; (2) choice to control such collection and use; (3) contact information to permit consumers to consult them about their information practices and consumers' privacy concerns; (4) access to ensure the accuracy of information provided; and (5) security to protect the integrity of that information. However, policy-based self-regulation on privacy has three major issues to be answered. First, what should be covered in a privacy policy? Non-standardized policies at various web sites may confuse users further. Second, how can a user's acceptance of a Web site's privacy policy be verified? Many web sites claim that users are accepting the practices described in their privacy policies by visiting their sites, though most users do not read those policies. Third, who will check whether a web site complies with its own policy or not. For the first issue, privacy advocates widely support four fair information practices:

1. *Notice*—Web sites would be required to provide the consumer clear and conspicuous notice of their information practices, including what information they collect, how they collect it, how they use it, how they provide choice, access, and security to consumers, whether they disclose the information collected to other entities, and whether other entities are collecting information through the site.
2. *Choice*—Web sites would be required to offer consumers choices as to how their personal identifying information is used beyond the use for which the information is provided. Such choice would encompass both internal secondary uses and external secondary uses.
3. *Access*—Web sites would be required to offer consumers reasonable access to the information a web site has collected about them, including a reasonable opportunity to review information and to correct inaccuracies or delete information.
4. *Security*—Web sites would be required to take reasonable steps to protect the security of the information they collect from consumers.

A study done by Georgetown University in June 1999 surveyed 361 commercial sites and found that 9 out of 10 asked a user to provide personal information such as name and address. However, privacy statements were posted only at two-thirds of those sites. Furthermore, less than 10% only met the fair information practice principles. In early 2000, the FTC surveyed a random sample of 335 web sites and 91 of the 100 busiest sites; and found that 97% in the random sample and 99% in the most popular group collected personal identifying information such as an e-mail address. The FTC's 2000 survey showed that 88% in the random sample and 100% in the most popular group posted at least one privacy disclosure. However, it concluded that the four fair information practice principles were not fully covered in those privacy policies. For instance, only 41% in the random sample and 60% in the most popular group met the first two basic standards of notice and choice.

To soothe the rising demand for legislation on privacy protection by consumers and privacy advocates,

the Network Advertising Initiative (NAI), a consortium of major Internet advertising companies, submitted a self-regulatory plan to the FTC. Under the NAI standards, consumers will be:

- Allowed to opt out the collection of anonymous data on the Internet for the purpose of profiling.
- Given a chance to determine if they want to allow previously collected anonymous data to be merged with personally identifying information.
- Allowed to give permission for the collection of personally identifying information at the time and place it is gathered on the Internet.

In July 2000, the FTC endorsed the NAI's plan as a way to encourage industry's voluntary self-regulation. However, critics are suspicious whether the industry can write an adequate rule to regulate itself for consumers. It is obvious that those advertisers who are not members of the NAI are not required to abide by the rules. In August 2000, Amazon.com, Inc. posted a revised privacy policy on the Web announcing that customers' information is the company's asset, thus can be sold. The revised statement says "As we continue to develop our business, we might sell or buy stores or assets. In such transactions, customer information generally is one of the transferred business assets. Also, in the unlikely event that Amazon.com Inc., or substantially all of its assets are acquired, customer information will of course be one of the transferred assets." Certainly the revision raises a red flag among privacy advocates claiming that customers have no recourse if they don't want personal information such as credit card numbers and home addresses passed on to some other company. Toysmart.com planned to sell its customer list after filing for bankruptcy protection. However, it gave in to complaints by privacy groups and investigation by the FTC and agreed to sell to a party that abided by Toysmart's original privacy policy.

2. Seal Programs

As a way to promote self-regulation, a seal program is used such as TRUSTe (www.truste.org) and BBBOnLine (www.bbbonline.org). Both of them check a site's adoption of the wide-accepted fair information practices and allow the site if it fully complies with the practices. These programs also do compliance checkup regularly to make sure of the site's ongoing compliance. A seal mark certainly will enhance a user's confidence level of a site. An AT&T research report titled "Beyond Concern: Understanding Net Users' Attitudes About Online Privacy" in April 1999 showed that 58% respondents would be willing to provide personal information to a site if it has both a privacy policy and a seal mark. The FTC's 2000 survey found that more than 90% of the random sample does not display a privacy seal, and more than 50% of the most popular group does not display a seal. Thus, the wide acceptance of a seal program by web sites should be done first so that a seal program may serve as an effective self-regulatory tool. Critics argue that groups such as BBBOnline and TRUSTE, which support self-regulation, don't have the power to effectively enforce their guidelines. They argue that legislation is both needed and inevitable.

3. Opting Options

There are two different approaches to the degree of user empowerment on personal data sharing: opt out and opt in. Opt out means that a Web site posts its privacy policy and allows a user to opt out if he decides not to be profiled. Unless a user explicitly chooses to opt out, the web site assumes that the user agrees with the site's privacy practices specified in its policy. But, privacy advocates criticize this approach. First, a user may get confused since opt-out options at different web sites are not standardized in terms of icon, size, location, etc. Second, most users may not bother to opt out since they do not read or even pay attention to a web site's privacy policy. Thus, privacy advocates argue that an opt-out approach does not have a practical value but is an excuse by vendors for their convenience at the cost of users. They prefer an opt-in approach as an alternative that requires vendors to get approval explicitly from users each time when they do customer profiling or data sharing.

Of course, vendors prefer the opt-out approach to the opt-in approach. First, they fear that the opt-in approach may scare off their customers for e-commerce. Second, the opt-in approach would be more costly for a vendor to implement than the opt-out approach. Many web sites already let visitors opt out, but most of those opt-out boxes are not easily spotted. Thus, some of newly proposed privacy laws would require every web site to offer a clearly written, prominently displayed opt-out box. But even a prominently displayed box might not guarantee a user's informed choice, since many users don't bother to read or click on such a box. That's why some politicians and privacy advocates are pushing for even tougher protections. Rather than putting the burden on consumers to opt out, they want to put the burden on companies to get consumers to opt in. Before a site could start collecting

and selling most data, it would have to get people to check a box giving it permission to do so.

There's no doubt that to opt in would raise the cost of doing business on-line. The opt-in requirement may scare off some consumers. But the objections to the opt-in rule go beyond the issue of reduced traffic. There would be less free information available, making it harder for companies to put together the kinds of demographic profiles that allow them to target customers more precisely. Since opt-in methods are relatively extreme, they should be used only for the most sensitive information or details such as a person's financial holdings and his sexual preferences. For instance, the Children's On-line Privacy Protection Act (COPPA) of 1998 limits the collection of information on kids under 13. As a way to protect children, the act requires parents to opt in, by written letter or fax to the site, before their children can use on-line chat rooms and message boards. However, the strict opt-in requirements may put too much financial and administrative burdens on some vendors. For instance, Thomas the Tank Engine, a children's TV show, decided to stop providing e-mail newsletter service to children all over the world. The Britt Allcroft Group owning the show explained that it simply could not afford staff or budget to manage the parental opt-in requirement specified by COPPA. Ideally, the best way to protect privacy on the Internet is to combine the best elements of both opt out and opt in. Two hybrid approaches between the two can be used. First, a revised version of the opt-out approach may be implemented to alleviate burdens of opt outs from users. Namely, vendors may create a one-stop opt-out site where a user can select the opt-out option once and for all. Second, vendors and privacy advocates may work out a list of sensitive information like financial, health, and sexual preferences, which requires users to opt in. The opt-out approach will be used for the other personal information to alleviate burdens on vendors.

Under the 1996 telecommunication law, the Federal Communications Commission (FCC) mandated an opt-in requirement to telephone companies to allow consumers to knowingly waive, at least partially, the privacy of their phone records. However, telephone companies challenged the FCC's rule in the court *(U.S. West v. FCC)* by claiming that the rule violated their free-speech rights by taking away their ability to tailor their communications to their customer. In June 2000, the Supreme Court took side with telephone companies by allowing them to continue using customer information, such as their phone-using habits, to market other services to them without their permission. This precedent implies that an opt-in requirement in a comprehensive manner for web sites would be impossible.

C. Legal Protection

The FTC is the federal government body that is to protect consumers from unfair or deceptive acts or practices. With its broad investigative and law enforcement authority over business firms, the FTC often deals with privacy-related incidents on the Internet like the DoubleClick case. In 1998, the FTC issued its first annual report on on-line privacy titled "Privacy Online: A Report to Congress" that contained the results of its first on-line privacy survey of commercial web sites. According to the 1998 report, the FTC found that almost all web sites (92%) were collecting a vast amount of personal information from consumers. But, few web sites (14%) complied with all four widely accepted fair information practice principles. The FTC recommended that privacy issues could be better addressed through self-regulatory efforts by vendors rather than government regulations.

In 1999, the FTC issued its second annual report titled "Self-Regulations and Privacy On-line," which showed that there was no improvement in web sites' compliance with the four principles. The FTC recommended that self-regulation should have more time to work while urging the industry to comply with the principles. The FTC conducted a survey of commercial sites' information practice in early 2000 and presented the results in its 2000 report titled "Privacy Online: Fair Information Practices in the Electronic Marketplace." According to the survey, only 20% of web sites implement all four fair information practice principles, though 88% post at least one privacy disclosure. The 2000 survey found that there was not much improvement in complying with the four principles, and thus, concluded that self-regulation was not sufficient as an effective method for privacy protection mainly due to inconsistency and lack of enforceable means. In May 2000, the FTC recommended a new law to establish a basic level of privacy protection for consumers and to enforce web sites to comply. The FTC commission itself was not fully in consensus on its request for government intervention as shown in its vote of 3 to 2 on recommendation of new legislation. Of course, vendors are strongly objective to a new legislation arguing that government intervention would be rather detrimental to the growth of emerging Internet-based e-commerce.

Existing laws and rules on privacy are shown below:

- Privacy Act of 1974; it protects records held by United States governmental agencies and requires them to apply basic fair information practices
- Privacy Protection Act of 1980
- Electronic Communications Privacy Act of 1986

- Privacy Protection Act of 1994
- Children's Online Privacy Protection Act of 1998 limits the collection of information about kids under 13; the law requires parents to opt in, by written letter or fax to the site, before their children can use on-line chat rooms and message boards.
- Financial Privacy Protection; President Clinton introduced a financial privacy proposal in April 2000 to protect personal financial information; the proposal will set guidelines over the transfer of personal financial information from financial institutions to marketers.

IV. PRIVACY-RELATED ORGANIZATIONS

There are various organizations on privacy, as shown below. Some of them are consumer advocates while others are to protect vendors' interests.

- *ACLU on Privacy:* (http://www.aclu.org/issues/privacy/hmprivacy.html) American Civil Liberties Union provides news, papers, and resources on protecting privacy; view sections on medical records and patients rights.
- *CDT Privacy Watchdog:* (http://watchdog.cdt.org/) Center for Democracy and Technology invites users to participate in researching privacy on the Internet; includes detailed instructions.
- *Center for Democracy and Technology:* (http://www.cdt.org/) Aims to preserve democratic and civil liberties on the Internet; read news and policies on such issues as censorship and privacy.
- *Coalition for Patient Rights:* (http://www.nationalcpr.org/) Citizens and medical professionals lobby for full confidentiality of medical records; sign petitions, read features, and contact Congress.
- *CPSR—Potential Threats to Privacy:* (http://www.cpsr.org/cpsr/nii/cyber-rights/web/current-privacy.html) Computer Professionals for Social Responsibility's Cyber Rights Working Group provides an overview of potential threats to privacy on the Internet.
- *Electronic Frontier Foundation:* (http://www.eff.org) This site advocates Internet privacy and encryption technology, broad public access to Internet information, and the Electronic Freedom of Information Act. It includes updates on legislation and court cases, a newsletter, and an archive of past topics.
- *Electronic Privacy Information Center:* (http://epic.org/) EPIC focuses public attention on privacy issues; includes policy papers on computer security and cryptography.
- *Internet Privacy Coalition:* (http://www.privacy.org/ipc) Presents news updates from the world of cryptography and encryption policy, including links to pending legislation. Users can also join the Golden Key Campaign by displaying the provided logo on their web sites.
- *On-line Privacy Alliance:* (http://www.privacyalliance.org/) Alliance of corporations, small businesses, and other professional associations including America Online, Intel, and Yahoo!, offers articles, debates, and resources on online privacy issues.
- *Privacy Rights Clearinghouse:* (http://www.privacyrights.org) The Privacy Rights Clearinghouse offers information on privacy issues surrounding wireless communications, wiretapping, social security numbers, and also offers links to other privacy resources.

V. FUTURE PERSPECTIVE

The so-called "new economy" has been emerging as a new driving force for the current long-lasting economic boom while replacing the "old economy." It is quite conceivable that the Internet will become a virtual society where people meet, talk, share information, shop everything including groceries, rent movies, trade stocks, bank, etc., just like what we do in our real society. There seems a consensus that the growth of e-commerce will be hampered unless the rising consumers' concerns on their on-line privacy are well addressed. However, vendors and consumer advocates are sharply divided on how this issue should be handled. Vendors naturally favor self-regulation by pointing out problem areas in government regulation. First, government regulation can not respond timely and flexibly to the needs in this newly emerging, technology-intensive area in view of its bureaucratic nature. Furthermore, they argue that slow response would be detrimental to the growth of e-commerce. Second, government regulation may end up with overregulation resulting in heavy financial burdens on vendors. So, vendors may be forced to switch from the current free services in many cases to fee-based services to absorb their regulation-complying costs thus disserving their customers. On the other hand, consumer advocates favor government regulation thus urging congress to adopt new legislation. First, they point out inconsistent and confusing privacy disclosures posted by vendors. Second, vendors claim that the customer data

that they collect are their property, thus they can sell them in case of bankruptcy or merger/acquisition. They want to have a legal framework for basic privacy protection rather than entirely rely on vendors' whims.

There are three technological and social developments that may affect the online privacy in the future. First, a web-based application and data are getting momentum as a new computing environment. Microcomputer users typically load all software and files on their computer's storage space. In contrast, a web-based application allows users to save files on a server in the Internet. For instance, a user of the popular Hotmail web-based e-mail system saves all e-mail files on the company's server. Vendors so-called ASP (application service provider) such as ThinkFree (www.thinkfree.com) and MyFreeDesk (www.myfreedesk.com) allow a user to access applications software and data files resided on their servers far away from a user's home computer. Thus, a user can have access to necessary software and data files from anytime and anywhere through the Internet. In view of the competitive advantages over the traditional method, web-based application and data service are expected to grow substantially in the future. In July 2000, Microsoft also announced a major shift toward on-line software with a plan called Microsoft.Net. Once personal information is held in the hands of a third party, it would be more easily available to hackers, government investigators, etc. Thus, on-line service of applications and data will certainly raise concerns about protection of the data and privacy of users.

The second development is the emergence of all kinds of Internet-ready or compatible appliances and wide acceptance of the Internet-based transactions. Already, an Internet-ready mobile phone set allows a user to use e-mail services and have access to a web site on a full scale. An Internet-ready micro-oven can download a new recipe or warranty-related news from the maker's web site. The United States Post Office also released its intention to assign an e-mail address to every American in addition to their existing mailing address. When a user uses many of these Internet-ready appliances, privacy protection will be much trickier. In June 2000, lawmakers came out with a new legislative initiative to provide electronic contracts the same legal status as handwritten signatures. Electronic signatures and records will significantly enhance the digital economy by providing greater confidence to consumers in their on-line business transaction. The way consumers, industry, and government conduct business over the Internet will be revolutionized with this legislation. Third, the Internet has been connecting the whole world as one community by dismantling the traditional time and distance barriers. It will be much difficult either through self-regulation or government regulation simply because a nation's sovereignty cannot go over its border line unlike information flows. A world-wide regulation may be needed for effective privacy protection.

SEE ALSO THE FOLLOWING ARTICLES

Advertising and Marketing in Electronic Commerce • Crime, Use of Computers in • Electronic Commerce • Electronic Payment Systems • Encryption • Ethical Issues • Firewalls • Human Side of Information, Managing the Systems • Privacy • Security Issues and Measures

BIBLIOGRAPHY

ABC News. The Price of the Information Revolution: article by Richard Martin explores how government, private businesses and cybercriminals have extended their reach into our private lives. http://abcnews.go.com/sections/tech/DailyNews/privacy981002.html.

Cavoukian, A., and Tapscott, D. (1997). *Who knows: Safeguarding your privacy in a networked word.* New York: McGraw-Hill.

Online Privacy: A Report to Congress, Federal Trade Commission, June 1998, http://www.ftc.gov/reports/privacy3/ toc.htm.

Our Four-Point Plan: E-privacy and e-commerce can coexist. Here's how to safeguard both. *BusinessWeek,* March 20, 2000. http://businessweek.com/2000/00_12/b6373006.htm?scriptFramed.

Privacy in the Digital Age: http://www.cnet.com/Content/Features/Dlife/Privacy/index.html.

Privacy Online: Fair Information Practices in the Electronic Marketplace. Federal Trade Commission, May 25, 2000, www.ftc.gov/os/2000/05/testimonyprivacy.htm.

Self-Regulation and Privacy Online: A Federal Trade Commission Report to Congress, Federal Trade Commission, July 1999. http://www.ftc.gov/os/1999/9907/privacy99.pdf

Schneider, B., and Banisar, D. (1997). *The Electronic privacy paper.* New York: John Wiley & Sons.

Wang, H., Lee, H. K. O., and Wang, C. (1998). Consumer privacy concern about Internet marketing. *Communications of the ACM.*

Procurement

Michael Rosemann
Queensland University of Technology

GLOSSARY

e-marketplace Simulates the behavior of traditional marketplaces, where buyers and sellers come together to make presentations, negotiate, and trade. Likewise, every e-marketplace has well-defined and accepted rules. They can be public or only accessible for a limited group. E-marketplaces can be differentiated in vertical and horizontal marketplaces. Vertical e-marketplaces focus on one industry, whereas horizontal marketplaces focus on one business process (e.g., procurement) with participants from various, typically not competing, industries.

e-procurement Describes new forms of procurement using the capabilities of the Internet. It is especially, but not only applied to MRO (maintenance, repair, and operations) products. The main objectives of e-procurement are an accelerated process through the elimination of paper and employee self-services, an increased transparency, and reduced costs and a higher standardization through preferred vendor agreements and a reduced scope of products.

material requirements planning (MRP) The classical term for all activities related to calculating the required material. MRP solutions were among the first off-the-shelf software packages. Their core components are bill-of-materials processors and forecasting modules. MRP is nowadays embedded in manufacturing resource planning systems.

procurement Covers all strategic and operational processes that are required for purchasing materials, goods, and services.

purchasing A part of procurement that includes the operational activities of procurement. This covers issuing purchase requisitions, evaluation and selection of vendors, the actual ordering, monitoring of open purchase orders and goods receipt. The purchasing process ends with the receipt of the delivery note.

PROCUREMENT can be defined as all strategic and operational processes that are required for purchasing materials, goods, and services. It is interrelated with various concepts such as materials management, just-in-time, or supply chain management. Selected strategic procurement alternatives are discussed and the main functions along the procurement process are explained. Finally, the characteristics of e-procurement are outlined.

I. INTRODUCTION

Globalization and shorter product life cycles are the main drivers for increased competitiveness in many markets. As one reaction, companies focus on their core competencies and transfer various activities to business partners. This phenomenon is called *outsourcing* and describes a shift from internal production and services to an increased willingness to purchase required goods and services.

Consequently, the importance of procurement is growing as it increasingly contributes to the overall business result. Procurement can be defined as all strategic and operational processes that are required

for purchasing materials, goods, and services. This includes legal aspects (e.g., design of contracts or specification of ownership), logistical aspects (transport of goods and delivery of services), financial aspects (e.g., credit limits), and further administrative aspects (e.g., vendor selection).

Procurement has many objectives. From a *financial viewpoint,* it has to minimize the costs related to all purchasing activities. This does not mean a minimization of the costs for every purchasing transaction, but the minimization of the relevant costs over time. Initial investments in integrated procurement solutions should be evaluated regarding their long-term effect on the costs per purchase transaction. Thus, the focus is the net present value or the lifetime value of a procurement relationship.

Besides financial goals, procurement has *process-related* goals such as to

- Maximize the transparency over the procurement process (monitoring of the status of purchase orders).
- Minimize the processing time in order to allow purchasing transactions to occur close to the actual time of demand, which minimizes time-dependent safety stocks.
- Maximize the flexibility of the procurement process, so that, if required, vendors can be changed or purchase orders can be modified.
- Maximize the process reliability, so that dependent processes (e.g., production) can start at the planned time.

Procurement is a service function for internal customers demanding materials, goods, and services from the external market. From a *customer perspective,* procurement has to satisfy these customers regarding their individual expectations and priorities regarding time, cost, quality, and flexibility. Procurement may provide consulting services regarding the specification of appropriate materials, goods, services, or vendors for its customers. At any time, procurement should be able to inform about the current status of all open purchase orders and vendor's performance.

Procurement must be open for *innovation* and *change.* Procurement-related information such as the past performance of vendors or quality characteristics of goods have to be maintained. The market development has to be observed, potentials in an increasing global market have to be evaluated and new forms of Internet-based procurement (e-procurement) have to be taken into account.

The objectives of procurement were differentiated in terms of financial, process, customer, and innovation objectives. Thus, it is possible to use the *Balanced Scorecard* approach, which exactly measures performance along these four dimensions for an ongoing evaluation of procurement activities. In a Balanced Scorecard, each of the four objectives is captured in a separate perspective, which contains relevant key performance indicators. Cause and effect relationships between these objectives show existing interdependencies. It becomes clear that procurement is much more comprehensive than a cost minimization exercise.

II. POSITIONING PROCUREMENT

A. Related Terms

The increasing importance of procurement led to the development of a range of procurement strategies, which are often hard to differentiate. The terms closest to procurement are purchasing, materials management, just-in-time, and supply chain management.

1. Purchasing

Purchasing is a part of procurement and covers the operational activities of procurement. This includes issuing (manual) purchase requisitions, evaluation and selection of vendors, the actual ordering, monitoring of open purchase orders, and goods receipt. The purchasing process ends with the receipt of the delivery note. Verifying and paying the invoice is already a part of the more comprehensive materials management process or a part of accounts payable. Procurement is more comprehensive than purchasing because it also covers strategic decisions about the entire procurement infrastructure (e.g., degree of centralization) or the long-term selection of preferred vendors.

2. Materials Management

Materials management is the classical term for all activities related to calculating the required material. Solutions for materials management were among the first off-the-shelf software packages. The process was called material requirements planning (MRP). The core components are bill-of-materials processors and forecasting modules. Materials management is traditionally focused on physical material and products and does not include services.

Figure 1 shows the general materials management process. Based on the production program, which specifies the demand of final products, the primary

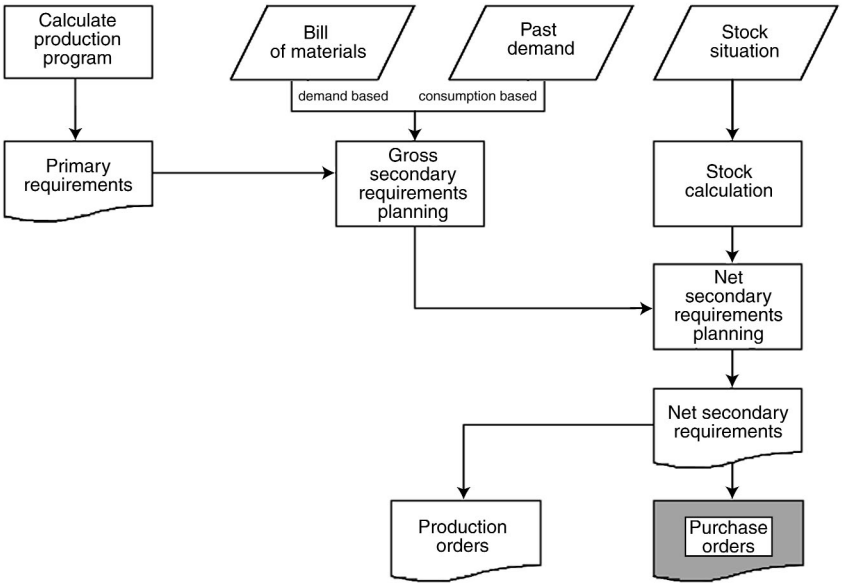


Figure 1 MRP reference model.

gross requirements are the initial input for the materials management process. This demand has to be converted into demand for secondary requirements, i.e., material that is a part of the final product. The calculation is based on the detailed bill-of-materials for A parts, which represent high-value parts. A parts can be identified through an ABC analysis, which classifies all parts regarding their value and the frequency with which they are ordered (Fig. 2). As a rule of thumb, it can be estimated that 70% of the procurement budget is spent on only 10% of all parts.

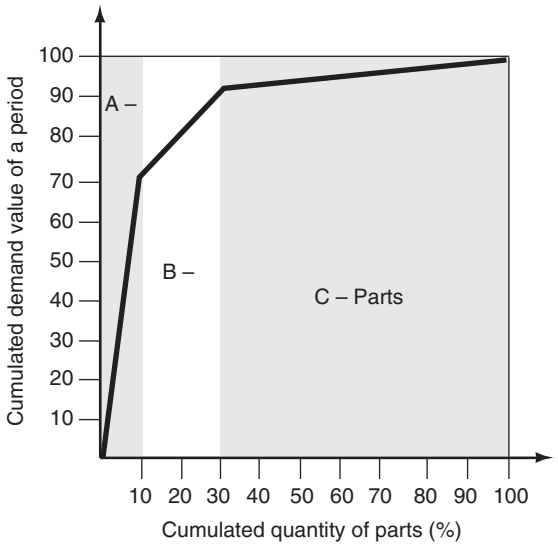


Figure 2 ABC analysis.

Consequently, it is obvious that A parts require a calculation based on the bill-of-materials as the most precise approach to determine their demand. However, this approach is too elaborated for most parts (C parts). Thus, more or less sophisticated forecasting algorithms based on past demand data lead to consumption-based demand. This gross demand does not reflect the current available stocks. To calculate the actual demand, the net secondary requirements have to be taken into account, which result out of the net secondary requirements planning process. These net secondary requirements lead either to production orders, if the demand is supposed to be satisfied through internal production, or to purchase orders. The latter one is input for the procurement process. Obviously, materials management includes the make-or-buy decision. This described process has to take place for every level of the bill-of-materials until the demand is completely calculated.

MRP solutions are nowadays embedded in more comprehensive manufacturing resource planning (MRP II) systems. MRP II extends MRP with further functionality related to master production scheduling, capacity management, order release, and the scheduling, monitoring, and controlling of the production process. As part of the material and warehouse management functionality, MRP II solutions support maintaining reordering points. These points specify a stock level at which a new production or purchase order is required in order to avoid negative stock or at least the need to use the safety stock.

Procurement and materials management overlap in the area of purchasing. Materials management goes

beyond procurement because it also includes activities related to warehousing. Moreover, materials management deals with the internal production, whereas procurement is only responsible for external materials, goods, and services. In contrast, procurement includes significantly more strategic decisions than materials management.

3. Just-in-Time

The terms *purchasing* and *materials management* are quite generic and can be found in most companies. Classical purchasing is still based on consolidating orders and building up stocks to satisfy internal demand. Just-in-time (JIT), however, describes a specific procurement strategy that aims to minimize stock. The main idea of JIT is to order and receive goods exactly when they are required. Ideally, the goods are transported directly from the goods receipt area to the production process without any further quality checks or required handling or sorting. JIT requires a close and long-term relationship with the vendor. Successful implementations of the just-in-time process can be found especially in the automotive and food industries.

Often, the vendor gets the information about the required variant via control points in the production. This would be, for example, in car production a point after which the production sequence of the cars is determined. In selected cases, vendors receive every 30 seconds a purchase order and deliver every 40 minutes goods in a sequence that corresponds with the production requirements. JIT has many prerequisites that form constraints to such an extent that it can only be applied in very selected situations. These prerequisites are as follows:

- The production process must be highly repetitive.
- Variants must be specified in advance.
- Just-in-time parts are typical AX parts with A characterizing parts of high value and X describing parts with a continuous and predictable demand.
- The vendor produces "100%" quality.
- The business partners are connected via a secure and stable connection for a continuous flow of data.
- The transport time from the vendor must be stable and predictable, but not necessarily short.

Obviously, JIT is a special form of procurement that represents an alternative for certain products if the prerequisites are met. Just-in-time requires a close integration between more technical operational production systems and business-oriented procurement solutions.

4. Supply Chain Management

The most recent term related to procurement is supply chain management. Unlike the previous concepts, supply chain management is focused on the *collaborative* design of procurement processes. Furthermore, supply chain management covers not only the transactions between two business partners, but the *entire business process* including the vendor's vendors and also the customer's side. Supply chain management aims for better coordination of the interrelated activities of all involved business partners. This includes the collaborative design of IT interfaces and logistical processes or the common management of warehouses. Of outstanding importance is the exchange of relevant information up and down the entire supply chain. This includes data about available resources or current expected processing times from the vendor and accumulated sales data from the customers. Thus, it will be possible for every business partner to base the planning process on current data.

Overall, expensive safety stock in the supply chain can be reduced, which leads to cost reductions. This transparent flow of information helps to avoid the *bullwhip effect* (Fig. 3). This effect characterizes a situation, in which from one business partner to another the uncertainty about the order situation increases along the supply chain. Supply chain management concepts minimize these stocks through a change of the classical push concept into a pull concept, in which the demand coming from the consumption side triggers all activities along the supply chain.

The establishment of a pull concept is also known in the retailing sector as *efficient consumer response* (ECR). In the ECR concept, information and goods flow in opposite directions (Fig. 4). Supply chain management can be characterized as a comprehensive inter-enterprise procurement strategy. It is of interest in long-term and intensive procurement relations, in which trust and collaboration are given.

B. Strategic Procurement Decisions

Strategic procurement includes all decisions with long-term and significant cost–benefit consequences. These decisions determine the procurement infrastructure. From a range of procurement strategies, the following alternative sourcing strategies are discussed:

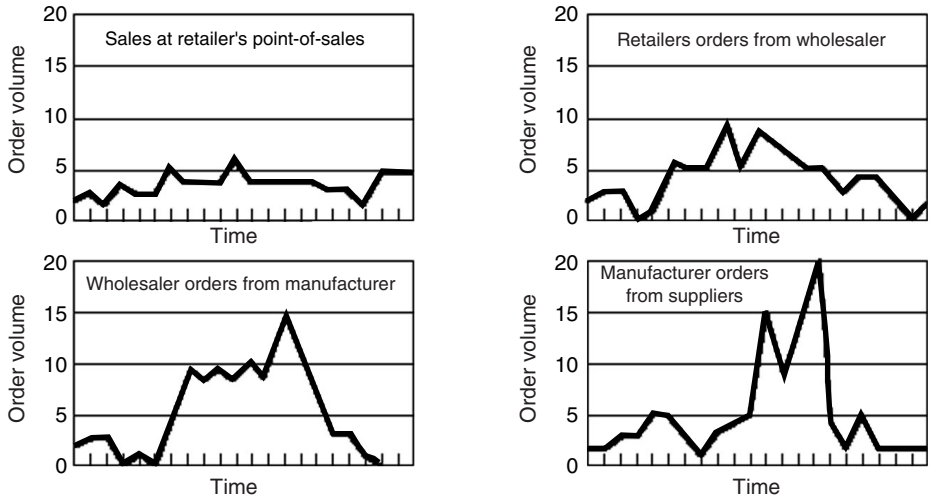


Figure 3 The bullwhip effect in the supply chain.

- Single, dual, or multiple sourcing
- Regional or global sourcing
- Modular sourcing.

1. Single Sourcing

Single (dual) sourcing describes a *vendor-related* procurement strategy, which concentrates the entire procurement for a material, a product, or a service on one (two) vendor(s). This contract is usually defined for a period of up to 3 years and guarantees the vendor a certain procurement volume, which supports significant economies of scale. Over time, though, the contract typically defines a continuous price decrease.

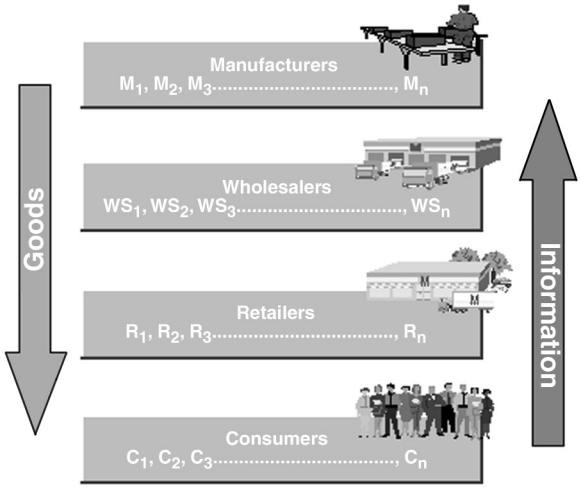


Figure 4 Information and material flow in the ECR concept.

This is based on the assumption that the vendor has a learning curve, which allows theoretically a price reduction of 20–30% with every doubling of the output. The main objective of single sourcing is to build up a long-term relationship with the vendor, which legitimizes comprehensive investments in the interbusiness procurement process. Information technology (IT) related investments include the design of interfaces for business administrative and technical product data and the logistical interfaces.

Single sourcing is quite common in supply chain management solutions and in the automotive industry. Many suppliers of car manufacturers are making 100% of their revenue within this industry. It is also not uncommon that 80% of their revenue depends on only one manufacturer. On the downside, single sourcing reduces competition between vendors. This means less pressure on price and product innovation. The concentration on one supplier also leads to an increased procurement risk. Moreover, single sourcing has a negative impact on the switching costs as investments into the design of IT interfaces are often specific for the relationship. As a consequence, it becomes difficult to change the vendor.

2. Global Sourcing

The differentiation between regional and global sourcing is based on *geographic criteria*. Global sourcing is becoming more attractive with economic agreements like NAFTA and those of the European Union. These associations standardize trade in the North American and European regions with a strong focus

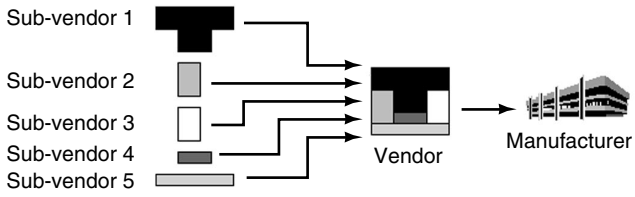


Figure 5 Modular sourcing.

on a reduction of the transaction costs. Furthermore, the significantly increased transparency provided by Internet transactions allows for a more comfortable investigation of foreign markets.

Global sourcing means participating in international production innovation and benefiting from advantages like lower personal costs, lower tax rates, or fewer limitations regarding production times. However, global sourcing also requires companies to handle differences in laws, language, culture, and quality norms, which add to the complexity of the procurement process.

3. Modular Sourcing

Companies with assembly processes have to make a decision regarding the modularization of the purchased products. A trend that can be observed in procurement is modular sourcing. Modular sourcing is a *goods-related* strategy, in which entire modules rather than single parts are purchased. It is based on outsourcing the assembly activities. Figure 5 describes the structure of the supply chain in the case of modular sourcing. The objective is to establish a pyramid of vendors with a significantly reduced number of logistical and IT interfaces. The coordination task moves to the tier one vendor.

III. THE PROCUREMENT PROCESS

During the 1990s, procurement solutions became integrated modules of enterprise-wide applications, so-called enterprise systems, also known as enterprise resource planning systems. As such, they are closely integrated with modules for production, sales, quality management, financial accounting, cost accounting, and asset management. The following paragraphs describe the main functions within the procurement process as they are supported by enterprise systems. The focus is on direct material and not on indirect material, services, or assets.

A. Master Data Management

Relevant master data in the procurement process are vendor and material master data as well as the interrelation between these two. The record, which relates vendors to materials, includes information such as the price or the average delivery time. Further relevant master data in procurement are the master records of employees working in the purchasing department, asset master data in the case of goods that can be depreciated, and also bank master data for the subsequent payment transactions.

Vendor master data can be differentiated into general data (e.g., number, name, address), logistical data, and accounting data. The key of a vendor master record is the vendor master number, which is either entered manually or derived from an automated numbering system. Perceived advantages of the manual system are that existing numbers from a legacy system can be used and that overall more semantics (e.g., regional information) can be integrated into the number. However, the general recommendation is to use an automated system, which minimizes manual mistakes, requires less maintenance, and is faster. If possible, the number should not include any semantic information. Instead, available vendor attributes should be used to the full extent. The detailed attributes of a vendor depend on its role. Besides the typical situation, in which a vendor receives the order, delivers the goods, sends an invoice, and finally gets paid, a vendor can also only be responsible for selected parts of this process. This role has to be specified for every vendor and determines the exact attributes and whether they are mandatory or optional. From an organizational point of view, the company must define who is in charge of maintaining the vendor master records. This is either the purchasing department or accounts payable or, usually, both. The required integration of both departments within the process of vendor data management is a critical challenge. A vendor master data process that is not optimized easily has a processing time of more than 2 weeks, which negatively influences the operational purchasing process.

A quantitative problem regarding vendor management is handling the number of vendors. Global manufacturers often reach a number of 20,000 or more vendors. This is not as much of a problem in terms of data storage space as it used to be because this resource has become quite affordable. The negative consequence of a fast growing number of vendors is instead seen in the loss of transparency in vendor-related reports and selection processes. Thus, the

company needs to define an archiving strategy, which regularly moves selected vendor master records into a data archive. The main criterion is typically a certain period in which no business occurred (e.g., 18 months). Furthermore, vendors that moved, merged, do not exist anymore, or represent incorrect data entries (spelling mistakes) should be identified and archived or directly deleted.

However, even with such an ongoing archiving strategy, the vendor master data must be further clustered in order to handle the remaining complexity. A simple, but popular approach is again the ABC analysis. An ABC analysis for vendors is based on the rule of thumb that 10% (70%) of all vendors get 70% (10%) of the procurement budget. Instead of such a value-based criteria, it is also possible to use the frequency of deliveries as a criterion.

After the selection of the criterion and the specification of the percentages starting with 10–20–70% as a default, procurement applications generate ranking lists showing the vendor with the biggest part of the purchase budget or the most deliveries at the top. The top 10% of vendors (A vendors) forms the business partners that require special attention. The performance of these vendors is, for example, observed more carefully and the evaluation criteria that must be maintained can also include information that has to be entered manually (e.g., degree of innovation, flexibility). Furthermore, A vendors are typical candidates for more sophisticated procurement solutions such as e-procurement (see Section IV). In contrast, efforts aimed at the huge number of C vendors should be minimized. Figure 6 shows an example from the enterprise system SAP R/3 demonstrating the graphical results of such an ABC analysis. In this example, the A vendors receive 40% of the purchase volume.

Material master data management contains the data maintenance for all raw materials, parts, product groups, end products, and further materials needed for transportation as well as services. Considering all forms of master data management, this one has the most interfaces with different business areas. In complex and comprehensive environments, the material master data has attributes that are maintained not only from procurement, but also from production, sales, financial accounting, cost management, asset management, quality management, and product development. From the perspective of procurement, the following attributes are of special relevance: order unit, shipping instructions, tolerances, average purchasing time, average price, reordering point, and JIT indicator. Material master data are organized in hierarchies and further structured by various classifications.

Similar to the vendor data management, it is important to regularly archive material master data and to further group material. Besides the ABC analysis based on the annual procurement volume, an XYZ analysis can help to differentiate between material with a regular demand (X parts) and material with unpredictable demand (Z parts).

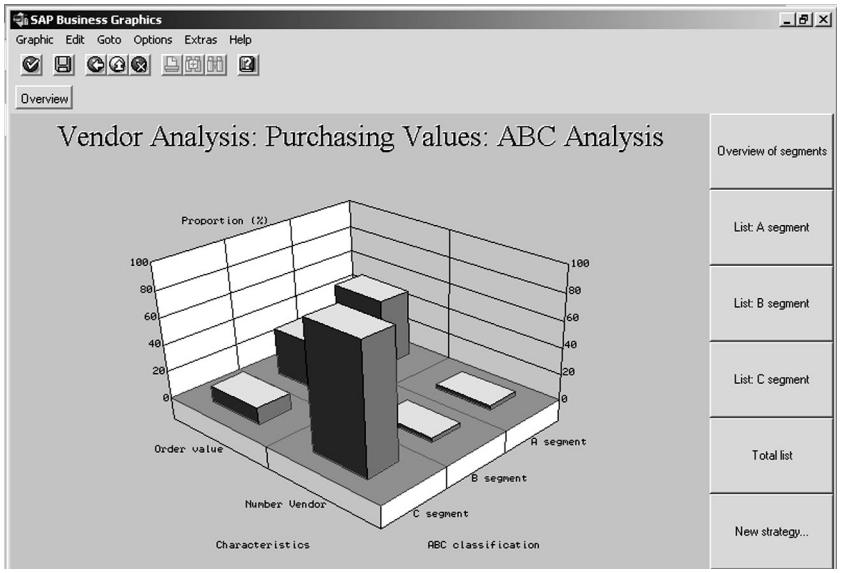


Figure 6 ABC-based vendor evaluation in an enterprise system.

B. Purchase Requisition

The operational purchasing process is initiated automatically or manually. An automatic trigger would be a reordering point system in which a purchase requisition is derived as soon as the stock falls below a predefined level. The reordering point describes the stock level at which new stock has to be ordered by a certain date to avoid negative stocks. Figure 7 indicates this in a simplified way because it does not depict safety stock. This approach requires basic materials management functionality and can often not be used for indirect material.

Besides the stock level, automated ordering systems are, especially in the retailing sector, often time based. In these cases weekly purchase orders are placed on predefined weekdays. In continuous replenishment systems, a vendor continuously (e.g., daily) refills the available stock back to a predefined level. Continuous replenishment is applied at retailers for consumer products such as milk or yogurt.

The purchase requisition can also be the manual trigger of the operational procurement activities. A purchase requisition includes a requisition header and items. The header contains, besides a unique number, attributes regarding the employee, who requires the goods or services, the cost center, and the date. The items of a purchase requisition include the material number, the desired quantity including tolerance, information about available vendors, and the desired delivery date.

A purchase requisition has typically to be approved through the head of the department or the cost center before it can be converted into a purchase order. This approval process is a classical shortcoming in procurement processes becaue it is time consuming and error prone. Workflow management systems can help to overcome this weakness as they automate the execution of the relevant functions and the identification of the employees in charge. This also includes mechanisms for replacement and escalation procedures. Workflow management might accelerate the approval process, but it does not reduce the number of processes to be performed. This can be achieved by defining budgets (per period and per single item) for employees. Within the budget, the employee can purchase without a formal approval process. Standards regarding preferred vendors and material further help to streamline this process. In the extreme form, an employee does not need an approval at all and can directly send a purchase order to a vendor. This principle is applied in many e-procurement solutions, when employees directly access the electronic product catalog of preferred vendors (Section IV).

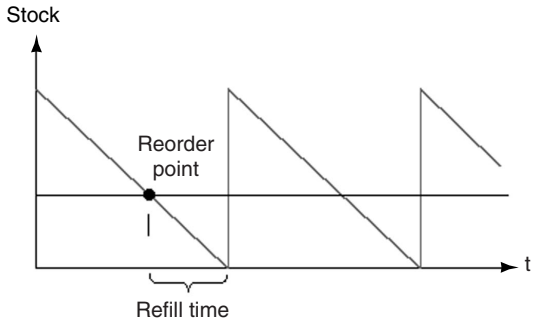


Figure 7 Reordering point.

All remaining valid purchase requisitions are forwarded to a central purchase department that formally evaluates and consolidates these requisitions. Once they are completed, including the final vendor and price information, they are converted into purchase orders. Purchase requisitions and purchase orders have a many-to-many relationship. In the cases for which a vendor is not known or not yet selected, a vendor evaluation and selection has to take place. Many public organizations require, for example, three quotes for purchase transactions that are not standardized.

C. Vendor Evaluation and Selection

Vendor evaluation is an ongoing activity that aims at adequately maintaining the vendor's performance in dimensions such as price, time, quality, and flexibility. Based on the importance of the vendor (see, e.g., results of the ABC-analysis) this evaluation is more (A vendors) or less (C vendors) comprehensive. Current procurement solutions support scoring models for elaborate evaluations. These models require a prioritization of the criteria. Each criterion can be maintained manually (e.g., evaluation of flexibility) or automatically (e.g., price). Relevant data not only come from procurement, but also from accounts payable (e.g., quality of invoices) and goods receipt (e.g., quality of goods and delivery notes). The vendors may also maintain to a certain extent their own evaluation criteria (e.g., service level agreements, planned delivery time, available services such as product development). This, however, requires typically Web-based vendor self-service solutions.

Vendors that perform well over a certain period of time might be selected as preferred vendors for certain materials (single sourcing). In this case, all material will be ordered without further selection processes

directly from this vendor. Besides the accelerated process, this also guarantees an ongoing revenue stream and protects related investments on both sides.

If a vendor has to be selected, the criteria gathered for the vendor evaluation can be used. This includes general data about the vendor (e.g., location) as well as individual data for the vendor's material (e.g., price, past quality, variants). Depending on the value of the items and the time of the last purchase, it might be useful to update these evaluation data. This can be done in the form of a comprehensive tendering process for special purchases. Faster solutions are possible within systems that allow reverse auctioning. In this process potential vendors can bid for an order. The process is usually transparent, so that the participating vendors can continuously modify their last bid in light of competing offers.

Once a vendor is selected, the purchase order is transmitted. Based on the number of previous similar transactions with the vendor, the information that is included in this purchase order is more or less comprehensive. In the extreme case of just-in-time purchasing the purchase order is reduced to the quantity of a specified material. All other information (details of the material, conditions, delivery time) is defined in long-term contracts.

D. Purchase Monitoring and Controlling

After the purchase order is issued, continuous monitoring and controlling of the current purchase transactions take place. Purchase monitoring includes the observation of open purchase orders regarding the fulfillment of the delivery date.

Purchase monitoring can also take place in the form of an early-warning mechanism. If a vendor does not perform well on other open purchase orders, this might be interpreted as an indication for potential overdue delivery. Purchase monitoring, however, also includes monitoring the internal demand and communicating eventual and possible changes of the purchase order to the vendor. Purchase controlling aggregates isolated monitoring data and updates attributes such as the average delivery time for a material with or without a reference to a vendor.

E. Goods Receipt

Within goods receipt the goods are received, a quality check is performed and the delivery note is compared with the actual goods received. Besides many logistical issues, which are not the focus of this article, it is the verification process within goods receipt that is of importance for the procurement process. Goods receipt is responsible for verifying that the received goods have actually been ordered, that the quantity is correct, and that the goods are not damaged. In an integrated environment, employees in the goods receipt enter the details of the delivery with reference to the relevant purchase order(s). This guarantees consistency in the process as it converts the relevant parts of the business document purchase order into a delivery note. The items of the purchase order that are actually received are selected. This activity consolidates purchase order(s) and delivery note(s), which have in most cases a many-to-many-relationship. The resulting document is called an *evaluated delivery note*.

Again, the goods receipt process can be streamlined through long-term contracts and standardization. Statistical analyses and a precise definition of the expected quality standards in the contract allow for a reduction in the intensity of quality checks for deliveries from selected vendors. In well-established relationships with vendors it might even be possible to offer vendor self-services in a form such that the vendor itself can refer to open purchase orders. From an IT point of view, the main task in goods receipt is the consolidation of purchase orders and delivery notes and the related quantity and quality check. The next activity, invoice verification, is responsible for the price check. It is the interface between procurement and accounts payable.

F. Invoice Verification

The invoice verification is triggered by the receipt of goods and invoices and ends with the posting of the invoices, which is input for the following payment process. The objective of invoice verification is to identify and overcome mistakes in the received invoices. The identification of mistakes reduces the amount of money that is spent on goods that were never received or not ordered. Thus, the quality of invoice verification has a direct monetary consequence.

Mistakes in received invoices can be specified as discrepancies between invoice and (evaluated) delivery note and/or between invoice and purchase order. Based on the document, which is used as a reference in the verification process (purchase order, evaluated delivery note, or invoice), different forms of invoice verification can be distinguished. Several concepts have been developed in order to streamline this

process, which in big companies easily occupies more than 100 employees. These approaches include the digitalization of the incoming data, the reduction of the number of invoice verification processes, and the replacement of invoices with credit vouchers.

1. Web-Based Data Entry

Web-based data entry utilizes vendor self-services to replace paper-based invoices with digital information. Instead of receiving a paper-based invoice, which has to be entered manually in the system with the risk of data entry mistakes, the vendor provides the invoice either in an electronic format (e.g., EDIFACT, ebXML) or enters the relevant data directly in a Web-based form. The invoice becomes available in the system and can also be verified to a certain extent automatically. Such a scenario, however, requires an unequal distribution of power, so that the vendors can actually be forced to perform this more time-consuming process.

2. Selected Invoice Verification

In ongoing business relationships, it will become obvious that some vendors are very reliable regarding the quality of the delivered goods as well as their invoice issuing process. In these cases, it does not seem economically efficient to check goods and invoices from these vendors with the same intensity as those from less reliable vendors. Selected invoice verification requires significant data about past performance. This approach reduces the number of invoice verification processes significantly by eliminating verification processes completely for selected vendors.

3. Evaluated Receipt Settlement

Evaluated receipt settlement (ERS) is the most radical approach in redesigning the invoice verification process. ERS is based on outsourcing the verification process completely. In this scenario, the payment process is not triggered by an incoming invoice, but by the goods receipt. These data lead to a payment liability record in the accounts payable system. Based on the contract with the supplier, the received goods are evaluated and a payment is issued. The time for any cash discount or payment commences on the date the goods are received. There is no need for the receiver of the goods to verify any invoice. Hammer and Champy quote the case of Ford, who uses Mazda as a benchmark. Apparently, Ford had 500 people involved in the invoice verification process. After introducing ERS, only 125 people were necessary. ERS is especially popular in the automotive industry, but can also be found in other industries. A key prerequisite for ERS is the relatively powerful position of the goods receiver because the main advantages are on his side. The time-consuming verification process moves to the vendors. Further prerequisites include these:

- The purchase order or the long-term contract has to be accurate because it specifies the data relevant for payment. It has to include price, payment terms, unit of measure, part number, quantities, and variants.
- Packing slips have to refer to a confirmed purchase order number or a long-term contract.
- Received purchase orders should be acknowledged by the vendor and any discrepancy should be noted immediately.

Even with ERS some invoices for special charge items (e.g., drum charges, customs) and purchase orders not referring to ERS are required. Invoices for ERS purchase orders typically will not be processed. Nevertheless, the goods receiver still requires a tax invoice for accounting purposes. This invoice has to be generated and is usually sent to the vendor for confirmation.

G. Payment

At the end of the invoice verification process, the correct invoices are posted and create open items in accounts payable. This is the main interface between materials management as a part of the logistical modules of enterprise systems and accounts payable as a part of the accounting modules. The following payment process is not part of procurement in a narrow sense.

The payment process is triggered by posted invoices. Its main objective is to pay invoices in time by making use of the best of alternative payment conditions. Payment processes are usually run in a predefined frequency (e.g., fortnightly). Open items in accounts payable are selected as long as they or the entire account is not blocked. All open items that are selected for payment because of the due date and a calculation of the interest rates in the case of alternative payments will lead to a payment in the form of checks or electronic funds transfer. At the same time the payment transaction is posted and equals the open item. This marks the end of the entire procurement process.

IV. E-PROCUREMENT

E-procurement is a term used to describe new forms of procurement using the capabilities of the Internet. It is especially, but not only applied to MRO (maintenance, repair, and operations) products. The purchased goods are high-volume, low-value items, ranging from stationery to office furniture. Some of the effects of the Internet have already been discussed above, for example, self-services in purchase monitoring and controlling. The main objective of e-procurement is to improve the process performance in terms of costs and processing time through automation and increased employee self-services. Further benefits are derived from corresponding standardization in terms of vendors and products.

One of the biggest organizational impacts of e-procurement is that it enables self-service in the actual purchase ordering. A prerequisite is that preferred vendors and products have been selected. These vendors offer their products ideally in electronic product catalogs, which can be based on their side, can be maintained by a third party (intermediary), or can be consolidated and stored on the side of the purchasing company. Thus, the interface with a central purchasing department can disappear and the procurement process can be streamlined. Furthermore, e-procurement aims for a paperless process. The consideration of economies of scale takes place during the negotiation and selection of vendors. As one consequence, e-procurement typically reduces the scope of the purchased products and supports further standardization.

Besides the described consequences of e-procurement (reduced number of vendors, reduced scope of purchased products, elimination of paper, employee self-services), it can also be characterized by new forms of interaction and collaboration called e-marketplaces.

Unlike classical business-to-business processes, e-marketplaces involve multiple business partners. They simulate the behavior of traditional (physical) marketplaces, where buyers and sellers come together to make presentations, negotiate, and trade. Likewise, every marketplace has well-defined and accepted rules. A marketplace can be public or accessible only to a limited group. They include the required functionality for the administrative part of the procurement process. In most cases they also support alternative forms of negotiation such as auctioning and reverse auctioning.

E-marketplaces can be differentiated in vertical and horizontal marketplaces. *Vertical marketplaces* are focused on one industry (e.g., automotive, chemical industry), whereas horizontal marketplaces are focused on one business process (e.g., procurement) with participants from various, typically not competing, industries. The objective in both cases is to consolidate market power in order to benefit from consolidated economies of scale.

Examples of vertical marketplaces are Covisient, a motor-industry site, where buyers including General Motors, Ford, DaimlerChrysler, and Renault/Nissan purchase their raw materials. Chem.connect and Chem.match are examples for the chemical industry. GlobalNetXchange is a marketplace for retailers, where Carrefour, Sears, Kroger, Sainsbury, Metro, and Coles deal with 70,000 suppliers, partners, and distributors worldwide. Trade-Ranger is an energy and petroleum exchange based in Houston with founders including Royal Dutch/Shell, BP Amoco, Conoco, Dow Chemical, Equilon Enterprises, Mitsubishi, Occidental Petroleum, Phillips Petroleum, Repsol YPF, Statoil, Tosco, TotalFinaElf, and Unocal. Multinational companies also set up their own portals (e.g., Wal-Mart's RetailLink). These portals are often operated by a separate company.

Horizontal e-marketplaces provide savings on indirect purchases such as office supplies or travel, which are not integral to the final products or services produced (MRO). One example of a horizontal marketplace is corProcure, the biggest Australian marketplace owned by Australia Post. Former shareholders and current participants are AMP, ANZ, Coca-Cola, Coles Myer, Foster's Brewing Group, Qantas, Carlton United, and Telstra. These companies have almost nothing in common, except for their huge size.

Marketplaces are of interest for companies that do not regard their procurement capabilities as a competitive advantage. Otherwise their participation would lead to a loss of this advantage because every participant gets the same lowest price. Moreover, the participating vendors of a marketplace are selected. Thus, marketplaces are usually based on high transparency and ongoing competition between the participating vendors. They do not conform to long-term contracts and single sourcing approaches.

From a more technical viewpoint, solutions for e-procurement often utilize *application service providers* (ASPs). Thus, a third party hosts the entire application. While the advantage is to outsource activities such as systems management that are not directly related to the business, systems integration can become a critical problem. Procurement is intensively interrelated with areas such as materials management, production planning, quality management, and financial accounting. Critical data such as vendor master records, vendor evaluation, products, inquiries, purchase orders, delivery notes and invoices, reordering points, etc. often have to be

maintained in the e-procurement solution as well as in the back-office application (enterprise system). These existing interrelations have to be maintained with managed redundancy in as ASP environment. If a third party is not only responsible for the provision of the IT solution, but also in charge of the actual procurement process, it acts as a *business service provider* (BSP). ASPs and BSPs can be different.

V. SUMMARY

This article stressed the perspective that procurement is much more than providing the right material and services at the right time, in the right quantity and quality, to the right places and people. It is also much more than minimizing the costs per procurement transaction. Procurement has been differentiated from related concepts and the main activities of a reference procurement process have been discussed.

Current procurement trends appreciate the value of long-term business-to-business relationships. Single sourcing and modular sourcing are two main strategies that provide vendors with a stronger role. At the same time, they help to reduce the complexity of procurement as they decrease the number of vendors, interfaces, and products.

The role of procurement will continue to increase regarding ongoing outsourcing activities and a process of internationalization. New opportunities under the umbrella term *e-procurement* are rapidly available and will change the classical procurement process. E-marketplaces will form new patterns of interaction within related business partners. However, at this stage it is difficult to predict what the new reference process for procurement will look like once these concepts and technologies are established.

SEE ALSO THE FOLLOWING ARTICLES

Accounting • Control and Auditing • Sales • Supply Chain Management • Transaction Processing Systems • Value Chain Analysis

BIBIOGRAPHY

Giunipero, L. C., and Sawchuk, C. (2000). *E-purchasing plus. Changing the way corporations buy.* Goshen, NY: JGC Enterprises.

Graever, M. F. (1999). *Strategic outsourcing: A structured approach to outsourcing decisions and Initiatives.* New York: Amacom.

Hammer, M., and Champy, J. (1993). *Reengineering the corporation. A manifesto for business revolution.* New York: Harper Business.

Kaplan, R. S., and Norton, D. P. (1996). *The Balanced Scorecard: Translating strategies into action.* Cambridge, MA: Harvard Business School Press.

Klaus, H., Rosemann, M., and Gable, G. G. (2000). What is ERP? *Information System Frontiers,* Vol. 2, No. 2, 141–162.

Simchi-Levi, D., Kaminisky, P., and Simiche-Levi, E. (2000). *Designing and managing the supply chain.* Boston: Irwin McGraw-Hill.

Wight, O. W. (1995). *Manufacturing resource planning: MRP II: Unlocking America's productivity potential,* rev. ed. New York: John Wiley & Sons.

Productivity

Jaak Jurison
Fordham University

GLOSSARY

business process reengineering The fundamental rethinking and radical redesign of business processes to achieve dramatic results in critical measures of performance.
conversion effectiveness The effectiveness with which IT investments are converted into useful outputs.
productivity The relation between products and services produced and the resources used to create them. It is the amount of output produced per unit of input.
productivity paradox The apparent lack of evidence of positive impact on productivity from IT investments.

PRODUCTIVITY has been a major concern since people began working in an organized fashion. It is equally important to manufacturing and service industries. Productivity in its broadest definition is the relation between the products and services produced, and the resources used to create them.

Productivity growth is one of the major challenges facing the developed countries of the world. Peter Drucker, widely regarded as the father of modern management, argues that

> the single greatest challenge facing managers in the developed countries of the world is to raise productivity of knowledge and service workers. This challenge, which will dominate the management agenda for the next several decades, will ultimately determine the competitive performance of companies.

To meet this challenge, all types of organizations have invested heavily in information technology (IT), some successfully, some less successfully. Many investments have been made without a clear understanding of how productivity and IT are related. The purpose of this chapter is to review the current state of knowledge of these relations and explore the role of information systems (IS) in productivity growth.

This chapter begins with a review of various productivity measures. Subsequent sections will examine the impact of IT on productivity and identify issues associated with understanding this relation. The chapter concludes with suggestions for managers and researchers.

I. PRODUCTIVITY—WHY IS IT IMPORTANT?

Productivity is an important indicator of economic performance. Which measures we choose and how we apply them determine how effectively we manage our resources. Productivity is defined as the relation between products and services produced and the resources used to create them. Productivity indicates how effectively resources are being used for the production of various goods and services. The resources may be labor, capital, materials, energy, or information—or any combination of these. Productivity can

be increased by producing more with the same amount of resources or producing the same amount with fewer resources.

Productivity can be measured at many different levels, ranging from the individual worker to an entire nation. At the national level, productivity is an important indicator of a country's economic strength and is the key determinant of a nation's standard of living. It also measures the country's competitiveness in the global market. Failure by a nation to keep improving productivity contributes to adverse balance of trade, unemployment, and other domestic problems.

At the industry level, a high degree of productivity is necessary to keep costs and prices competitive. Industries with a record of continuous productivity gains are able to survive and flourish (as long as they are not displaced technologically), while low-productivity industries experience business decline. With globalization, many industries in advanced industrialized nations are facing increased competition from low-wage developing countries. These industries can remain competitive only by increasing productivity at home to offset the low wage advantage of their competitors in the less industrialized world.

At the firm level, productivity is the main source for competitive advantage. High productivity is essential to profitability and survival. Firms with above average productivity tend to enjoy higher profit margins than their competitors. Failure to maintain adequate productivity growth is often the cause of business failures.

Finally, at the individual level, high productivity is necessary for maintaining one's standard of living and is a source of personal satisfaction and self-fulfillment.

Productivity is equally important at every level, because in the final analysis, productivity and the living standard of a nation and its citizens depend on the productivity gains made at all levels within the nation.

II. PRODUCTIVITY—HOW IS IT MEASURED?

A. Productivity Defined

The concept of productivity is relatively simple and straightforward. However, efforts to operationalize it, and to find and apply appropriate measures, are quite difficult.

The traditional approach to productivity is to compare the quantity of output (desired results) with the quantity of one or more inputs (resources) used to produce the output. This relationship is expressed as a ratio:

$$\text{Productivity} = \frac{\text{Units of output}}{\text{Units of input}} = \frac{0}{\text{I}} \qquad (1)$$

The *output* usually represents some unit of production or desired results. Examples of output are the quantity of products produced, number of reports prepared, number of contracts negotiated, number of customers visited, etc. The main requirement is that the output is expressed as a measurable quantity. In case of multiple outputs, a common set of units is used, usually dollars. At the firm level, output can be defined either as gross output or value-added output. Gross output is the total production of the organization, including intermediate goods and services that are purchased from the outside. Value-added output is defined as the output resulting from the efforts of the organization's own resources—it excludes the cost of intermediate goods.

Inputs are the resources consumed during production of the outputs defined in the numerator. Typical inputs are labor hours, capital equipment, energy, raw materials, supplies, and electricity. Like outputs, inputs must be quantifiable and measurable.

Productivity measurements can be based on a *single input factor* or a combination of *multiple factors*. Single-factor measures, also known as partial productivity measures, indicate how well a firm is using that single factor in the production of its output. Multifactor measures take into account many or all input factors, expressed in a common set of units, usually dollars. Converting all data to the same set of units makes it possible to compare firms with different inputs and outputs.

B. Single Factor Measures

The most common single-factor productivity measure is labor productivity. It can be expressed in terms of output per worker or output per hour worked. Economists use this indicator to determine a nation's standard of living relative to other countries. At the firm level, particularly in labor-intensive industries, labor productivity is used to determine the efficiency of the organization. Other single-factor productivity measures can be output per machine, output per ton of material, or output per some other type of input.

While single-factor productivity measures are useful indicators for managers, they do have some serious shortcomings. The primary problem with defining productivity too narrowly is that it can lead to unsound management decisions and counterproductive behavior. It is easy to increase the productivity of one factor by replacing it with another. For example, a firm may invest in new technology that reduces the number of labor hours needed to produce a particular product. Clearly, labor productivity has increased,

but at the same time, capital input increased. Economists call this effect capital-labor substitution. The substitution effect accounts for most of the long-term labor productivity gains made in the United States manufacturing industry. As a result of capital investments in new technologies, the direct labor content in the United States manufacturing industry has declined to a point where it is only a small percentage of total production costs.

C. Multifactor Measures

In order to account for the capital-labor substitution effect, economists developed the capital-labor multifactor or total factor productivity factor. This measure takes into account the contribution of both labor and capital in the production of goods and services. It is expressed as:

$$P_1 \frac{O_v}{L + C} \tag{2}$$

Where P_1 = total factor productivity
L = labor
C = capital
O_v = value added output

The capital input C is the cost of all capital resources devoted to the production of goods and services, including working capital (cash, accounts receivable, and inventories) and fixed capital (structures and equipment). Note that this measure focuses only on labor and capital. It neglects the effects of intermediate goods and services such as purchased items and outsourced labor. Therefore it is necessary to measure the value-added output, O_v, in the numerator.

A broader view of productivity expands the notion of capital-labor substitution, expressed by Eq. (2), to take into account all input factors (e.g., materials, energy, outside services). Total productivity is calculated from the formula:

$$P_t = \frac{O_T}{L + C + R + Q} \tag{3}$$

Where P_t = total productivity
O_t = total output
L = labor
C = capital
R = raw material and purchased parts
Q = other miscellaneous goods and services

In this case intermediate goods are treated as input and therefore must also be included in the output.

This type of a total productivity index, represented by a single equation, is useful because it gives managers a handy scorecard to answer the question: How are we doing?

It has an advantage over partial- or single-factor measures by making information available on tradeoffs among factors and allowing senior managers to make more informed business decisions.

The concept of total productivity is particularly useful at the firm or business unit level of the organization. Partial productivity measures; such as given by Eqs. (1) and (2) are useful for day-to-day operational control at lower levels of the organization.

D. Productivity Over Time

The relations discussed thus far are static. They indicate where an organization or industry is at a point in time. They are useful when comparing current performance against other organizations, industries, or countries. To understand whether productivity has improved or declined, measurements must be made over time.

Logically, productivity can be increased in several ways over time:

- Producing more output with the same level of input.
- Producing the same level of output, but with less input.
- Increasing both input and output, but increasing the amount of output more than the amount of input.
- Decreasing both input and output, but decreasing input more than output.
- Producing more output with less input.

Productivity measurements must be made in a single set of units, typically money. The value of inputs and outputs depends on price. However, prices fluctuate over time due to inflation and other factors. In order to account for the effect of price changes on input factors and outputs over time, current prices must be deflated by their respective inflation factors. In order to assess productivity changes over time, prices are kept constant at some base-period value. Various price indices, such as the consumer price index, producer price index, or an in-house product price or cost indices are used for inflation-free productivity measurement.

III. PRODUCTIVITY PARADOX

Productivity is an important economic indicator that serves many purposes. Various productivity measures can be used to evaluate the efficiency of an individual,

an organization, an industry, or a country. Productivity can also be used as the fundamental economic measure of information technology contribution. Managers are primarily interested in measures of individual and organizational productivity to enhance organizational performance. Productivity measures, for them, are useful performance indicators for assessing the results of their IT investments. They provide a simple method to keep score and to facilitate performance comparisons across individuals, organizations, and time. They are commonly used in benchmarking; the process of comparing an organization's performance against best performing firms. In productivity improvement programs, the measures are used to track changes in productivity over time.

Management researchers and economists are usually concerned with productivity at the industry sector and national levels. They prefer sophisticated mathematical models for analyzing productivity behavior and the linkage between inputs and outputs. Although these models enjoy theoretical advantage over simple ratios, their focus on technical elegance and statistical rigor makes them difficult to apply in ordinary business settings. Furthermore, because they are based on aggregated data from a large number of firms, these models may not accurately reflect the full impact of IT on productivity. For many years economists concluded that computers did not pay back on their investment because no correlation was found between IT spending and labor productivity. This apparent lack of evidence that IT investments have an impact on productivity has been labeled the " IT productivity paradox" or simply the "productivity paradox" after the Nobel Laureate economist Robert Solow who noted that "we see computers everywhere except in productivity statistics." The paradox has been a major concern for business managers because IT investments have long been regarded as having enormous potential for improving the productivity and competitiveness of all types of firms. Almost all large firms in the industrialized world have made massive investments in computers and communications systems based on the assumption that these investments will improve organizational productivity.

While macroeconomic studies have failed to show the correlation between IT investments and productivity, firm-level studies have been able to show that the use of IT has indeed increased productivity. A comprehensive firm-level study by researchers at the MIT Sloan School of Management shows that IT spending has indeed been productive and concludes that the productivity paradox has disappeared.

More recently, IT-related productivity gains are also starting to show up in United States macroeconomic data. Economists now agree that IT is the key factor in American productivity growth in terms of both labor productivity and total factor productivity. A study by economists at the United States Federal Reserve Board shows that about half of the large increase in the average labor productivity growth rate from the first half of the 1990s (1.6%) and the second half of the decade (2.9%) can be attributed to the use of IT. The remaining half of the increase comes from the production of computers and semiconductors affecting total factor productivity. Therefore, we can conclude that IT does have a productivity payoff, but that it has taken more time than we realized to emerge.

Why did it take so long to see productivity gains at the overall national level? There are several explanations. First, IT is like any new technology—it takes time to learn to use it effectively. Economic historians have observed that the productivity benefits from major technological innovations like the steam engine and electricity took many years to develop. In the case of electricity, productivity did not start to accelerate until 40 years after its introduction. It took that long for firms to reorganize their factories to take full advantage of electric power. Information technology investments appear to have similar delayed payoffs. The time lag between IT investments and the benefits stream varies depending on the type of investment and organizational absorption rates. This time lag can range from less than one year for the simplest applications, to more than ten years for the most difficult. It is particularly lengthy in the case of IT infrastructure investments which make future applications possible. The benefits of IT infrastructure investments usually accrue from the applications, often years later than the infrastructure development. For example, the benefits of a sophisticated communications network that links a firm's worldwide locations aimed to position the firm for anticipated future business opportunities are not known at the time of the investment and will not materialize until the applications for exploiting the business opportunities are in place.

In the case of United States service industries, where enormous IT investments were made in the early 1990s, many productivity improvements and cost-cutting measures were not taken until the mid-1990s when competitive pressures started to mount. Many investments may have initially created organizational slack (defined as resources an organization possesses in excess of what is required to maintain the organization), which was eventually eliminated by downsizing and business process reengineering in response to increased global competition.

The second explanation of the productivity paradox is that IT productivity studies based on data at the

industry sector or the whole economy suffer from aggregation effects. Studies that aggregate data from a large number of different businesses do not capture the critical mediating effect of management and organizational changes. Information technology benefits depend to a large degree not on the size of the investment, but on management's effectiveness in converting the investment into results. Because organizations differ vastly in their conversion effectiveness, productivity gains are not achieved uniformly across all firms; In fact, they vary enormously from one company to the next. While some firms enjoy phenomenal success from IT spending, others do not. Frequently, IT projects have been mismanaged or the technology has been used inappropriately. A large percentage of IT projects never get implemented because they get canceled before completion. Therefore, in the aggregate analysis, the negative experiences of some firms cancel out the benefits obtained by effective firms.

Another reason why United States macroeconomic data failed to show IT impact on productivity is that, for many years, overall IT spending represented only a small share of total capital spending. Therefore, the economic impact on overall labor productivity was, until recently, relatively insignificant.

Another explanation for the productivity paradox is that traditional productivity measures ignore many key benefits of IT. Because firms invest in IT for many different reasons (e.g., improving quality, increasing the variety of products and services, or providing better customer service), conventional productivity measures used in macrolevel studies do not adequately capture the full range of IT-generated benefits.

It should be noted that the various explanations for the productivity paradox are not mutually exclusive. Most researchers agree that a combination of factors and not a single factor have contributed to it. But the lesson learned from the productivity paradox is that generalizations from macroeconomic data about the relation between IT and productivity can be misleading and more insight can be gained from individual firm-level studies.

IV. IT AND PRODUCTIVITY

A. IT Impact on Productivity

The key to improving productivity is not working harder, but working smarter by using innovation in the technology and organization of work. Information technology has a potential for improving productivity in two major ways: (1) reducing labor by input by automating many manual operations and (2) reorganizing or enabling improvements in business processes. The term "information technology" refers to a wide range of capabilities offered by computers, software applications, and telecommunications networks. Each has a potential for improving productivity. Table I shows some commonly used productivity-enhancing technologies and their uses at various organizational levels.

Information technology impacts on productivity can be viewed at various organizational levels, ranging from individual to interorganizational. At the individual level productivity applications provide support to the end user in two distinct ways: (1) by automating routine tasks and (2) by providing information for better decision making. At the organizational level IT enhances productivity by providing the means for more effective communication, collaboration, and sharing of information and knowledge by workgroups and teams that are either collocated or dispersed through distant geographical locations. It also allows the integration of activities across an entire enterprise, eliminating time-consuming bottlenecks and streamlining complete business operations. Interorganizational systems make it possible to integrate the operations of a firm with its suppliers, customers, and business partners, thus eliminating inefficiencies at organizational boundaries and improving productivity for all participants.

Table I Productivity-Enhancing Information Technologies

Individual level
Word processing
Desktop publishing
Spreadsheets
Decision support systems
Expert systems
Organizational level
E-mail
Groupware
Workflow systems
Local and wide area networks
Teleconferencing and videoconferencing
Group decision support systems
Enterprise resource planning systems (ERP)
Executive information systems
Mobile computing
Interorganizational level
E-commerce applications
Supply chain management systems
Electronic data interchange (EDI)

B. Productivity at the Individual Level

Early productivity gains from IT were achieved mostly by automating highly repetitive, routinized, manual processes, primarily at the individual worker level, substituting capital for labor. Examples of these early IT investments are

- Robots that replace manual production workers and operate 24 hours a day.
- Computer-aided design (CAD) systems that automate the repetitive steps in the creation and revision of new product designs.
- Desktop publishing systems that simplify and automate the production of professional quality documents.
- Spreadsheets that help organize data and perform tedious calculations in a matter of seconds.
- Expert systems that allow automation of routine decision making by providing access to special expertise within the whole organization.
- Automated call answering and customer service systems that make it possible for employees to handle significantly more customer inquiries.
- Automatic teller machines (ATMs) that automate the work of bank tellers. They allow banks to handle a large increase in the number of deposits and withdrawals without hiring additional tellers.

Automation efforts have been particularly successful in improving productivity in manufacturing industries. As a result, direct labor costs in most manufacturing firms in the United States and other advanced economies has been reduced to a fraction of total production costs. Further manufacturing productivity gains are no longer achieved from the automation of individual processing steps, but from using IT to augment the work of designers and managers. For example, simulation systems allow product designers to simulate the performance of new designs under complex, real-world conditions before committing designs to production, thus reducing or eliminating costly design changes later in the product life cycle. Managers can make more informed decisions about quality control and inventory management, allowing parts and materials to move faster and more efficiently. Using inventory control and logistics systems, they can keep parts and work-in-process inventories at minimum levels, thus minimizing warehousing costs and avoiding costly stockouts.

Compared to manufacturing, the productivity impact of automation in service industries is less clear. In services, work is mostly performed by white collar and knowledge workers whose inputs and outputs are difficult to define. Knowledge work is done by office workers who work with information and professionals who create information and knowledge. Scientists, lawyers, architects, journalists, and managers are all knowledge workers. Their output is mostly intangible and therefore difficult to measure.

For knowledge workers, the computer and communications systems typically:

- Allow automation of routine and repetitive activities, leaving exception handling as the primary knowledge worker responsibility.
- Provide the ability to undertake tasks that were too large to perform previously.
- Allow them to make better decisions in less time.
- Improve the ability to collaborate.

In addition, by performing many routine tasks, IT can provide permanent memory of what was done (so a firm is not dependent on the memories of individuals). By organizing the work, the computer frees people to work at the next level of knowledge where better ideas pay off in competitive advantage.

It is important to understand that the role of IT in knowledge work is mostly informational—instead of replacing human labor, IT augments and enhances it. Informational systems provide the necessary information for managing and controlling various operations in the firm. Their impacts depend not on the amount of information, but on how effectively that information is used to make and implement management decisions.

Examples of impacts from improved decision making are:

- Decision support systems (DSS) improve decision making by allowing users to rapidly evaluate the effects of alternative decisions.
- Data visualization techniques help employees analyze large amounts of data rapidly, spot problems and unusual conditions, and make faster and better decisions.
- Executive information systems (EIS) provide easy access to timely key performance information for senior executives in order to make more informed decisions.
- Data warehouses, special databases containing vast amount of data, used with on-line analytical processing and data mining software allow users to analyze a variety of business patterns and trends.

An important productivity issue for knowledge workers is how the time saved is used. If IT allows a given amount of output to be produced by less labor input,

then there are more labor hours available. The key question is how these hours are used. If they remain unused, they become what is known as organizational slack, with no contribution to productivity. Productivity increases only if these hours are reallocated as an input to an another productive activity. For example, managers can use the increased time to think about new ideas that improve the business or they can spend it playing games on their computers. Sometimes this added thinking time results in major benefits. Measuring the output value of this added time is difficult because not all new ideas improve business performance.

One of the ironies of improving productivity through technology is that many investments are made to support individual workers, yet the payoff comes at the organizational level.

C. Productivity at Organizational Level

The most dramatic productivity gains from IT accrue at the organizational level using networked computers to support the communication and collaboration necessary to make working teams, business units, and complete enterprises more productive.

Information technology applications can improve organizational productivity in various ways:

- Electronic mail is now universally accepted as an effective medium for communication and coordination of work.
- Various types of groupware can enhance productivity of teams and working groups by facilitating human interaction. Groupware provides support for communications and collaboration among workers and for coordinating group activities. It makes it possible to collect and distribute information, allowing team-based organizations to be more effective.
- Decision support systems, initially developed for individual decision makers, is increasingly used by workgroups under the name of group decision support systems (GDSS). They facilitate communication and provide modeling capabilities for group decision-making processes.
- Workflow systems eliminate bottlenecks in document processing by automatically routing electronic documents via local and wide area networks to appropriate workers for their contribution. Insurance firms have achieved enormous productivity increases by combining workflow systems with imaging systems that replace paper documents.
- Wide area networks can be used to bring together the best experts in a company. For example, Boeing Company was able to increase productivity, avoid building a physical prototype, and cut the design time of the 777 aircraft in half by using sophisticated software and a wide area network to connect 12,000 engineers and support staff to 2200 workstations and four mainframes.
- Videoconferencing systems link remotely located team members for strategic planning, project reviews, and job interviews.
- The Enterprise resource planning (ERP) system allows integration of a large number of individual applications across the whole enterprise, speeding up the processes, eliminating delays, reducing error rates, and raising overall productivity.
- Mobile computing allows workers with notebook computers, pagers, and mobile phones to work on group projects from anywhere in the world.

All of these systems have the classic features of substituting capital for labor as a way of improving productivity. But the impact of computers and communication networks is much larger than that. Information technology can fundamentally change the ways businesses operate. Consider the following:

- As the use of IT eliminates barriers of time and distance, firms can locate factories, offices, and research and development facilities where productivity is highest.
- Firms undertake “relationship marketing” where, for the same effort, products are customized to individuals rather than being produced in lot sizes and where design and production quickly respond to changing demand.
- The number of middle managers needed to control the organization is reduced because the span of control can be increased.
- By storing and disseminating information, IT facilitates building and sharing of knowledge throughout the company.
- In addition to eliminating tasks and streamlining processes, IT can create the basis for new organizational forms ranging from cross-functional teams to fully integrated organizational networks.

These examples demonstrate IT’s role as an enabler of business process redesign. In this role, IT offers the potential for the largest gains in organizational productivity. By redesigning or totally replacing antiquated business processes with new processes, firms

can take full advantage of the new advanced capabilities of IT. The fundamental rethinking and radical process redesign, commonly known as business process reengineering, is the key to getting organizational productivity improvements from IT.

In business process reengineering, IT can be used to eliminate tasks and reduce layers of hierarchy. Information technology can also be used to change the sequence of activities in such a way as to eliminate time-consuming bottlenecks and make the whole process more efficient. For example, concurrent engineering allows many design tasks that were previously done sequentially to be done in parallel. Properly designed databases coupled with workflow software make such process redesigns possible.

Effective business process reengineering requires making major changes in organizational structures and human resources policies. A reengineered business process inevitably affects individual jobs, skill requirements, and reporting relationships. Training programs have to be changed in order to develop new skills. Additionally, new incentive and reward systems are required. In effect, IT is a catalyst for organizational change. The implication of this change is that the focus on productivity can shift from cutting cost to increasing both organizational and individual effectiveness.

D. Productivity at Interorganizational Level

Many strategic IT systems extend beyond the boundaries of an individual firm. In these interorganizational systems, IT helps a firm to integrate its internal operations with those of its customer, suppliers, and business partners. By sharing information, these interorganizational systems enhance the ability to communicate and collaborate more effectively than their competitors. Early interorganizational systems used expensive dedicated high-speed telecommunications channels. Today an increasing number of these interorganizational systems are becoming web-enabled, using the Internet to establish quick and inexpensive linkages.

It is widely predicted that the full productivity impact of IT will come from future Internet-based e-commerce applications. The most significant impact is most likely to come from business-to-business (B2B) e-commerce. It can make businesses more efficient in two major ways:

1. Lowering procurement costs by making it easier to find the lowest cost suppliers and place orders.
2. Making the supply chain more efficient and lowering the cost of delivering goods and services.

Firms that have switched to on-line ordering report substantial savings in transaction costs and the cost of goods and services purchased. In some industries these savings can amount to as high as 40%. On-line supply chain management can reduce costly inventories and eliminate layers of middlemen, leading to complete transformation of business practices. Many companies are copying Cisco's and Dell Computer's build-to-order business model in their efforts to achieve the productivity gains necessary to remain competitive. Other companies are trying to figure out how to either complement or integrate their existing services with the new channels made possible by the Internet.

V. PRODUCTIVITY AND PROFITABILITY

While the IT impact on productivity is generally positive, its impact on financial performance has not always been as successful. There is growing evidence that, in general, IT investments may not lead to higher profitability in the long run. Productivity enhancement technologies are usually available to all competitors. Information technology innovations can be copied relatively easily and it is hard for technical innovators to create barriers of entry and enjoy sustainable competitive advantage. Due to competitive pressures, firms are forced to pass the productivity gains on to the consumer in the form of lower prices, better quality, or better service. For example, IT investments by commercial banks created benefits which they did not capture, but passed on to their customers in the long run. Similar spillover benefits occur in other service industries like finance and insurance. Benefits that are passed through to the downstream sector of the value chain are referred to as consumer surplus by economists.

Another example of such a pass-through is drug wholesaler McKesson's electronic order entry and distribution system Economost, which provided benefits for both McKesson and its customers, the independent pharmacies. The customers benefited from reduced transaction costs, reduced wholesale prices, reduced inventory holding costs, and various value-added management services. McKesson was able to increase productivity in order entry, sales, and warehouse operations. The introduction of Economost led to a major improvement in the entire drug distribution industry. Mostly, it benefited the large national and regional wholesalers, who were able to exploit the technology. The most noteworthy result, however,

was that many of the benefits from Economost were passed on to the independent pharmacies in the form of lower prices, allowing them to compete against the large chains.

VI. NEW PRODUCTIVITY MEASURES

A. Limitations of Traditional Productivity Measures

While IT has a positive impact on productivity, its true impact is much larger. Information technology can create business value in many different ways. The variety of ways business is deploying information technology makes a traditional single global metric, such as productivity, no longer adequate for assessing IT's full value. Today, IT investments are made for a variety of reasons, including improved quality, increased variety of products or services, and better responsiveness to customer needs. It is generally believed that traditional measures of the relationship between inputs and outputs fail to account for nontraditional sources of value.

Traditional productivity measures described in the beginning of the chapter are focused on efficiency, the ability to convert input into outputs at lowest cost in a given time period. Appropriately applied, these productivity measures can be powerful tools. However, they were originally developed for manufacturing operations, where outputs and inputs are easily quantifiable and the link between resource usage and outputs can be described and understood. As businesses increasingly become more service-oriented and knowledge-based, the weaknesses of traditional measures become further apparent. The classical productivity measures are especially limited in dealing with time, quality, knowledge work, and IT infrastructures.

B. Time

Many leading companies achieve competitive advantage by focusing on management of time, whether it is in production, product development, distribution, or service. Time differs from other resources. It cannot be purchased like labor or capital equipment. However, just because time cannot be bought, it is not free. If two companies are producing identical products with the same amount of resources, then by traditional measures, they are at the same productivity level. Then again, if one firm ships products faster than its competitor, most managers would consider that firm to be more productive. The less time it takes to get results, the more productive the organization.

Companies that recognize the key role of time in their operations usually develop customized measures for tracking performance based on time. These time-based measures are based on specific business objectives and typically include turnaround time, product development cycle, customer response time, and order fulfillment time.

C. Quality

In addition to reducing costs, IT can increase the quality of products and services for consumers. The benefits of quality are difficult to measure and are not included in traditional productivity measures. Quality and productivity are closely interrelated, but their relationship is often misunderstood. Many managers still think that quality and productivity are mutually exclusive—that high quality can only be achieved at the expense of productivity. We have compelling evidence now that improvements in quality actually lead to increased productivity. Defects are not free—it takes almost the same amount of resources to correct a defect as to produce another unit.

How should quality be factored into productivity considerations? The traditional way defines output in terms of products or services that meet quality specifications. This approach focuses on producing only acceptable outputs, rather than on sheer numbers. It cannot adequately deal with improvements in the quality of products and services demanded by customers.

The need for new measures that take quality improvement into account has been recognized for some time, especially in industries that use IT to gain competitive advantage through superior quality. Many experts believe that measuring the quality dimension can be just as important as measuring the quantity of goods and services. As an example, innovative service firms use customer-based quality measures to make periodic assessments of how the customer perceives quality and what relative value the customer places on each component of quality. Based on this information, these organizations devise special quality metrics and use them regularly to monitor the quality of their intermediate and final outputs.

D. Service and Knowledge Work

Defining meaningful units of output is particularly difficult for white-collar and knowledge work. Traditional

productivity measures, which count the number of outputs and inputs, are difficult to apply and are often inappropriate to organizations engaged in knowledge work. The output is mostly intangible and the process of measuring knowledge work is complex. New measures are needed that can define and determine outputs more accurately and provide improved insight into the linkages between inputs and outputs. Better methods are also needed to measure how IT creates value, especially in the service industry, where quality, timeliness, and outcomes are instrumental for success.

While some progress has been made in measuring the effectiveness of knowledge work, most organizations still lack appropriate measures. A commonly used approach in industries such as banking and financial services uses transactions as the basic unit of output. However, productivity measures based on transactions can be misleading unless they are adjusted for quality variations. An alternative method of measuring output in service industries is to measure the outcome of the service. In this way, the full value of the service is captured without the need to adjust for quality.

Because knowledge workers represent the fastest growing service sector in all advanced industrialized economies, it is critically important to measure and enhance the productivity of their work. This is perhaps the most important and difficult challenge ahead. It appears that the focus needs to change from output to outcome, from efficiency to effectiveness. The challenge lies in how to define and measure effectiveness. It is generally agreed that the measures should be tied to the goals of a particular organization. Therefore they must be multidimensional—a single productivity measure will not suffice for all organizations. They must be defined in terms of customer needs and values, and take variations in quality and variety into account.

One potential approach for dealing with variations in quality and variety of products and services is based on the hedonic framework. This increasingly popular framework among economists measures the value of goods and services in terms of the value of the salient characteristics of the good or service.

Because the use of information technology in knowledge-based industries often changes the way businesses operate, productivity is not a sufficient measure of IT benefits. Therefore it is sometimes more appropriate to focus on broader organizational performance measures. For example, one powerful method for measuring organizational performance is economic value added (EVA) analysis. This allows us to measure the productivity of the firm by taking into account all factors of production, including the cost of capital. It is a single powerful measure of organizational performance that is rapidly gaining widespread acceptance among managers.

It is unlikely that a single universal productivity measure that captures all components of business performance and at the same time provides insight and guidance for managers for corrective action will be found. Organizational variance in terms of products, services, and strategies means that what works for one firm may be inappropriate for another. The complexities of IT and its potential impacts on organizational performance necessitate multiple perspectives, of which productivity is just one.

VII. GETTING THE PAYOFF FROM IT INVESTMENTS

The previous discussion illustrates the complexity and difficulty of realizing the full potential of IT investments. Investing in more IT and having more information is not sufficient to improve productivity. Technology by itself does not automatically create productivity gains or economic benefits, but it is an essential part of the management process that produces results. The role of IT is to make new and innovative processes possible.

The full realization of productivity benefits from IT generally requires not only significant investment in hardware and software, but also a major change in the firm's organizational structure, business practices and processes, and culture. Many business processes were created long before business computers were developed. If these archaic processes are not reengineered, the introduction of IT simply may speed up some activities, but not lead to significant changes in overall productivity. In some cases a firm may be even worse off after investing in IT. Similarly, small incremental changes are not likely to yield significant productivity gains. Substantial productivity gains can be achieved only through radical business process redesign, known as reengineering.

Most reengineering projects, particularly those of strategic nature, are expensive and risky. Technology cost is usually the smallest part of the overall reengineering cost, the largest cost is in changing the organization. For every dollar invested in IT, there are several dollars of organizational investments needed to achieve the full productivity potential. Without in-

vesting in reengineering, training, incentive systems, and other organizational changes it is not possible to get the full payoff from IT investments. One way to think about business processes is to consider them as assets that provide benefits over a long time period. From this perspective, the cost of reengineering can be treated as the cost of acquiring a new asset. Information technology offers the greatest benefits when IT investments are complemented with appropriate organizational assets.

In general, the organizational changes that lead to higher productivity include self-directed work groups, decentralized decision making, increased training and education, and incentive and reward systems that encourage teamwork. One study found that firms that combined IT investments with decentralized work practices were about 5% more productive than firms that did not.

From a broader perspective, the productivity impact of IT depends on how well the purpose and objectives of the system are aligned with the firm's business strategy. There are two broad categories of IT investments:

1. Investments focused on internal efficiencies and control.
2. Investments with customer orientation focused on quality, service, flexibility, and speed.

Information technology in the latter category are often characterized as strategic information systems, seeking to improve customer and supplier relationships for competitive advantage and increased market share. While the two strategies are not mutually exclusive, they have different implication on productivity.

The purpose of the internally focused systems is to improve productivity and achieve operational effectiveness through better use of various resources. Focused investments targeted to improve the performance of specific processes can have considerable productivity impact, often with modest expenditure. However, the biggest payoff at the organizational level comes from investments in core business functions rather than in peripheral processes.

Strategic systems, on the other hand, are often targeted to provide other benefits such as increasing sales and retaining the loyalty of existing customers. Their impact on productivity is often indirect. Other measures of business value, described in the previous section, are needed to evaluate the benefits of these systems.

As new technologies become available, there will be new opportunities for IT-based productivity improvements. Managers need to know what gains they can obtain in their own organization. Simply throwing money at the problem is not enough. Managers need not only analyze the options available to make sure that they make business sense, but also to look at how the options they select will be implemented. Implementation is a process of matching business processes, people, and technology. Implementation is an area that separates winners from losers.

VIII. MANAGEMENT GUIDELINES

We conclude with a set of guidelines to help managers make better investment decisions and guide implementation efforts.

The key to productivity payoff lies in a systematic management process that starts with the alignment of business and technology strategies. Priorities about IT investments must be developed through a dialog between business and IT leaders. Clear answers must be found to the questions:

- What is the business purpose of each investment?
- How will this investment create value?
- How does it support our business strategy?

These questions are, of course, true for all investments. If the purpose of the investment is to create new strategic or tactical opportunities, productivity is of secondary importance. In other investments, typically involving ongoing operations, the major objective is to increase productivity.

After answering these fundamental questions, managers should follow up the investment decision with a systematic management process that includes the following steps:

- Reengineer business processes to take full advantage of the potential of IT.
- Make sure that there is a champion who promotes use of the technology and helps overcome the resistance to implementation.
- Develop appropriate productivity measures for assessing the benefits of the investment. Measures tailored to the specific objective of the investment are often more useful than broad productivity measurements.
- Use pilots and small-scale experiments to obtain early feedback on progress and validate the measures.

- Develop a tracking system for measuring these benefits as well as the costs.
- Use the productivity measures to develop incentive and reward systems that help the organization to achieve better performance.
- Invest in training and education.

IX. CONCLUSIONS

The question is no longer whether IT investments are productive, but how they should be made more productive. In successful companies managers start with realistic expectations, understand why they are investing in IT, understand the processes that need reengineering, and are willing and able to make appropriate organizational changes.

We can conclude that IT is a major contributor to productivity improvements at individual, firm, and national levels. In the last few decades IT has changed the way of conducting business for practically all firms in the industrialized world. The continued growth of investments in Internet and e-commerce is expected to produce further productivity gains and economic growth in the years ahead.

SEE ALSO THE FOLLOWING ARTICLES

Benchmarking • Control and Auditing • Cost/Benefit Analysis • Knowledge Management • Management Information Systems • Operations Management • Service Industry • Total Quality Management and Quality Control

BIBLIOGRAPHY

Brynjolfsson, E., and Hitt, L. M. (1998). Beyond the productivity paradox. *Communications of the ACM,* 41(8), 49–55.

Gray, P., and Jurison, J., eds. (1995). *Productivity in the office and the factory.* Danvers, MA: Boyd & Fraser.

Hitt, L. M., and Brynjofsson, E. (1996). Productivity, business profitability, and consumer surplus: Three different measures of value. *MIS Quarterly,* 20(2), 121–142.

Quinn, J. B. (1992). *Intelligent enterprise: A knowledge and service based paradigm for industry.* New York: Free Press.

Willcocks, L. P., and Lester, S. (1999). *Beyond the IT productivity paradox,* Chicester: John Wiley & Sons.

Program Design, Coding, and Testing

Ruth Guthrie
California Polytechnic University, Pomona

GLOSSARY

baseline software Software that has been "frozen." No more changes are possible without a formal change control process.
cohesion The measure of relatedness within modules or classes. The designer's goal is to maximize cohesion.
computer aided software engineering (CASE) tool An automated tool that assists designers in developing data flow diagrams, data dictionaries, and entity-relationship diagrams. CASE tools can also generate source code.
coupling The measure of relatedness between modules or classes. The designer's goal is to minimize coupling.
design rule or **pattern** Rules given to describe solutions to software problems. The rules/patterns are used as a guide to produce high-quality software.
formal test High level system testing to ensure that requirements are satisfied.
informal test Lower level unit and integration testing done by the developer.
integration test Combining elements that make up the system (modules or classes) and then testing to see that they operate correctly.
metrics Measures of software quality throughout the life cycle.
software development life cycle (SDLC) The process of creating a software application. Phases of the SDLC include requirments, analysis, design, code, test, and maintenance.
unified modeling language (UML) The industry standard for designing and generating object oriented programs.

I. DESIGN, CODE, AND TEST IN THE SOFTWARE DEVELOPMENT LIFE CYCLE

Development methodologies for software projects have evolved from a highly structured, end-to-end process into more flexible, iterative processes. In the 1960s when data processing was in its infancy, programs had features termed "spaghetti code" and programming languages had statements including GOTO that enabled programmers to jump anywhere they liked in the programming logic. As a result, software was hard to understand, develop, and maintain. A need arose for a more rigorous methodology in development of software. To compensate for the lack of logic and rigor, modular, hierarchical techniques were created and the software development life cycle (SDLC) was introduced.

The life cycle ensured that a defined process was followed in the development of software applications. Software development was divided into several independent phases known as the SDLC. Specifically, the phases are Requirements, Analysis, Design, Code, Test, and Maintenance as shown in Fig. 1.

With the traditional methodology, each phase is an independent, isolated activity. One phase has to be completed before the next phase begins. This methodology is termed waterfall, because phases "trickle down" into one another until the application is developed. The structured, inflexible nature of this methodology created problems with software development. Waterfall methodology fails to provide feedback between phases of development. At the onset, the requirements are set in stone. After analysis is performed, no newly discovered knowledge can be used

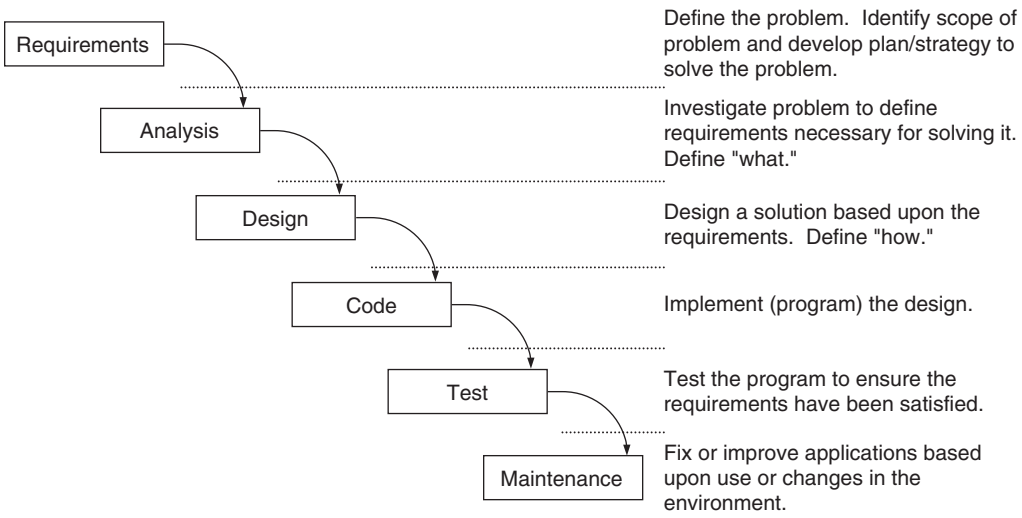


Figure 1 Traditional SDLC.

to improve the requirements. Similarly, after design is done, analysis cannot be changed without great exception. Often, by the time a project was complete, technology had advanced, requirements had changed, and the people who once approved the requirements had moved on to better jobs. The result was the development of systems that failed to meet users needs. Often these projects were plagued with cost and schedule overruns.

Continual problems with cost overruns and late deliveries caused the software industry to examine the life cycle and come up with ways to improve software quality by inventing more flexible methodologies. The need for involving the user throughout the life cycle and the need to mitigate risk by breaking systems development into smaller subsystems became apparent. To make the waterfall methodology less risky, systems were broken into smaller functional units and then integrated as a later phase of the life cycle. However, phased development and implementation still had many of the same problems with inflexibility. A new methodology was needed to build software in such a way that changes in design could be easily achieved without burdening other components of the system. Rapid prototyping methodology and spiral development arose to replace traditional development. The development of object oriented languages and graphical user interfaces directly fit the new methodologies.

The iterative or spiral development life cycle introduced by Boehm in 1988, shown in Fig. 2, was adopted to overcome the shortcomings of the traditional waterfall method. Instead of each phase being independent, phases provide feedback and are reiterated until the design is complete. During each iteration of the prototype, functionality is added and increasingly the users needs are implemented. Requirements are flexible at the beginning and become more defined as development continues. This more flexible approach allows new and better knowledge to be integrated during the development rather than waiting until the end of the life cycle and doing it as part of maintenance.

Having a flexible methodology allows developers to apply new knowledge to the software design as it becomes known. This creates a better solution to the problem, prior to implementation. This is a preferable way to operate for two reasons. First, it ensures that the user's needs are met. Second, it is far less expensive to improve a solution during the early stages of a project rather than after the program is deployed.

However, poor quality can exist with iterative and traditional methodologies. It is necessary to control the software development process in a way that ensures the quality of more reliable, maintainable, expandable systems. Rigorous design, code, and test are all essential to the development of quality software. Understanding and managing these processes helps to better understand and meet customer's requirements. It is also imperative that the customer be involved during the development process so their feedback can directly improve the design.

II. STRUCTURED DESIGN, CODE, AND TEST

Design, Code, and Test phases of development are often lumped together in the SDLC and referred to as implementation phases. These phases occur after the analysis phase of the life cycle. During the analysis phase, developers work towards defining what the system

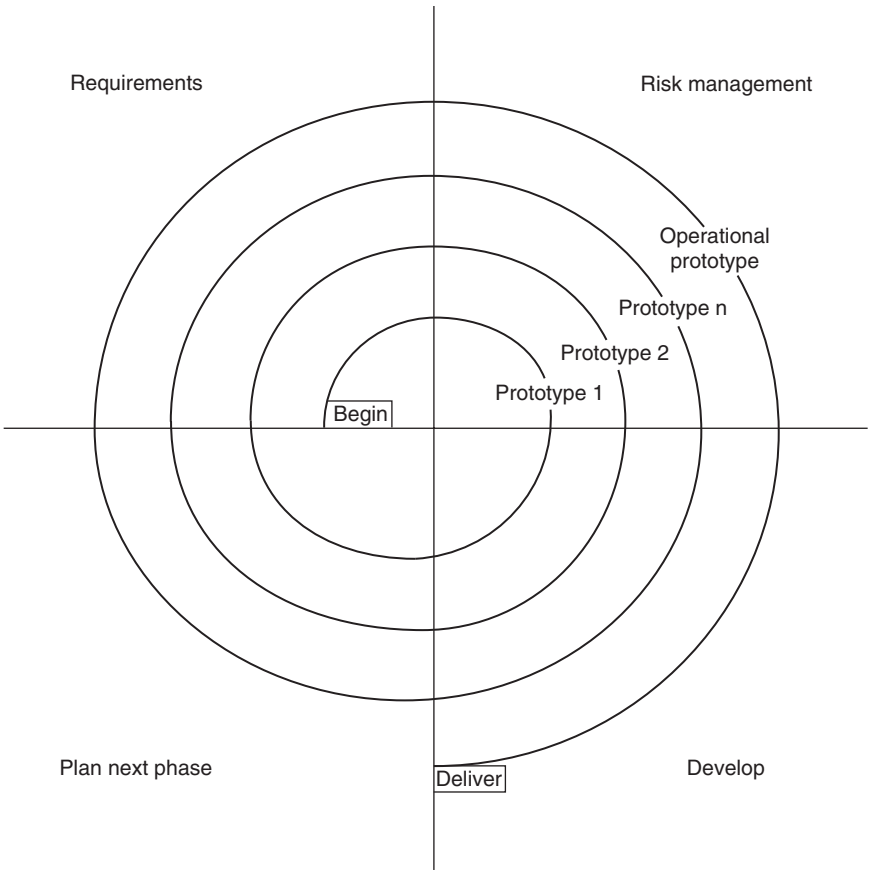


Figure 2 Spiral development methodology.

should be. During the design phase, the focus shifts to defining how to implement the findings of analysis. After the design model is built, programmers will build (code) the system to the design specifications. After construction, the system is rigorously tested to ensure that all the requirements have been met. Figure 3 identifies artifacts produced during each phase of structured development to document and communicate the system. The artifacts from each phase provide a starting place for the next phase of development.

A. Structured Design

Design is where the requirements defined during the analysis process are transformed into logically structured artifacts that model the system. The analysis phase of the SDLC generates a set of specifications identifying detailed system requirements. Outcomes of the analysis process are used during design to produce a logical solution of the system. Development of a software application has many types of design work associated with it. Generally, four types of design activities are recognized:

- Data design—defines the data format, type, and relationship to modules in the program
- Architecture design—brings data and procedures together and defines how modules are related
- Interface design—how the human will interact with the computer, how modules will interact with each other, and how modules will interact with devices external to the system
- Procedural design—design of the programming implementation

The design process is iterative. The first iteration may keep the design at a high level. As the design process progresses, increasingly detailed aspects of the design are defined. Finally, the design is complete and the artifacts of the design process are used as a blueprint for building or programming the system.

1. Data Design

Data requirements are formed during the analysis phase with the development of entity-relationship diagrams (ERDs). This modeling technique is used to

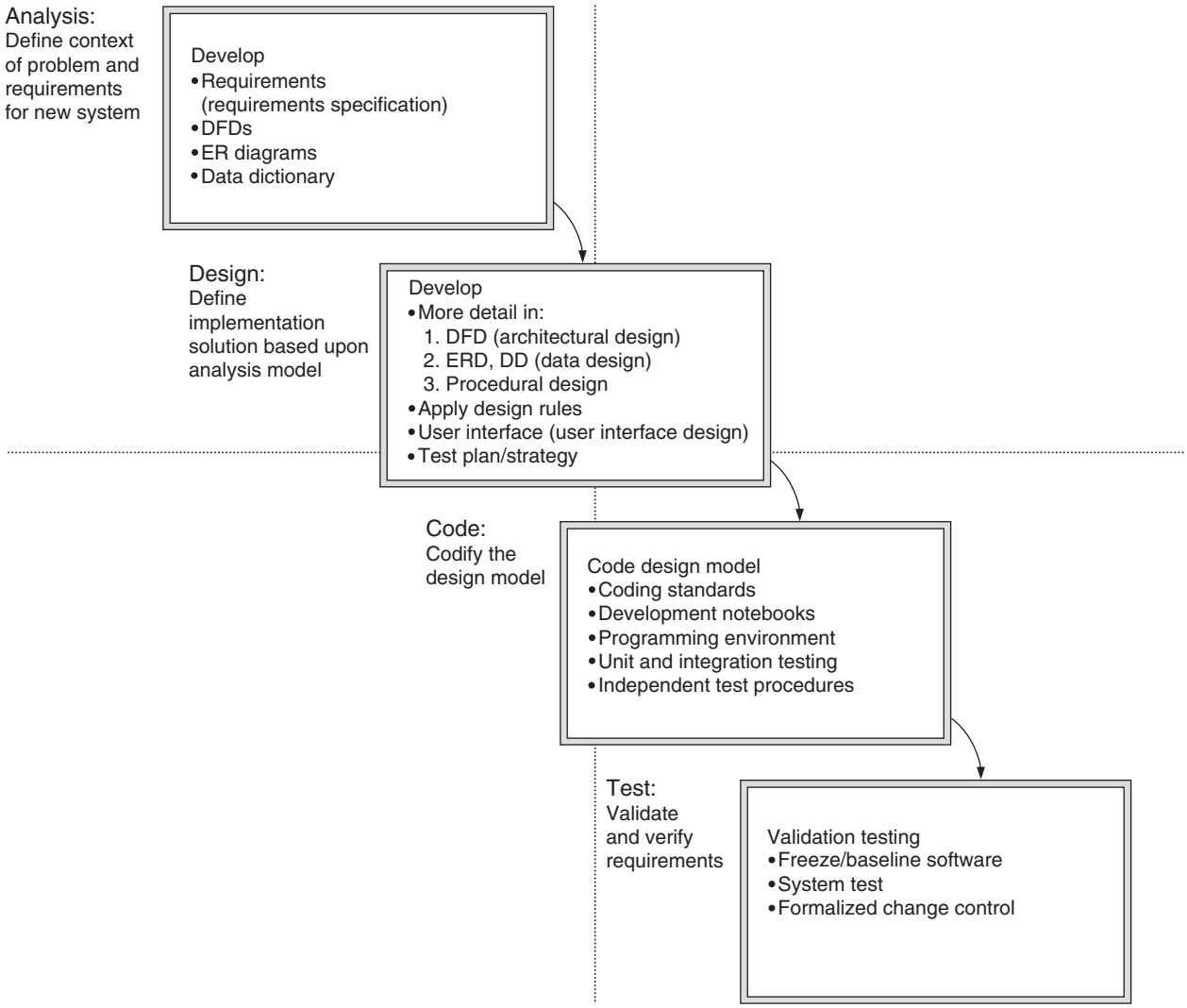


Figure 3 Activities and artifacts of structured analysis, design, code, and test.

define the data elements for the system, data attributes, and the relationship between data elements. Figure 4 shows a simple ER diagram for a bank deposit. On the diagram, entities are represented by rectangles, attributes of the entities are represented by ovals, and relationships are represented by the lines between the entities. You can further tell which attributes will be primary keys in the data design because they are underlined. From the odd notation, representing the relationship between the account and the deposit, the diagram shows that for a single account there can be many deposits. ER diagramming allows for modeling of the data and its relationships, one-to-one, one-to-many, or many-to-many, providing the model for developing the database.

As ERDs are developed to identify and describe the data, a data dictionary should also be developed to specify the data. The data dictionary contains specifications for all data used in the system. Typically, a dictionary entry will include what the data are called, what type of data it is (e.g., Character, Floating Point), and what modules use it. Defining the data, how it is used by the system, and what modules use it gives feedback to the designer that can reduce program complexity and improve modularity.

2. Architecture Design

At a high level, an application is made up of several subsystems with specific functionality. Subsystems are broken down hierarchically into lower level processes or modules. Designing the software architecture is a matter of grouping modules into programs and subsystems in a way that optimizes program structure.

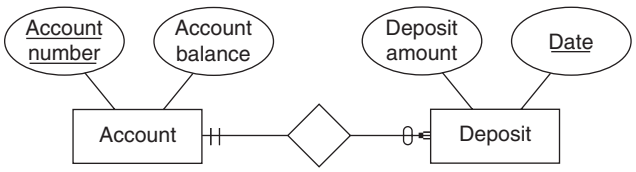


Figure 4 ER diagram example.

Data flow diagrams (DFDs) developed during the analysis phase of the project are now developed more fully during design. Figure 5 shows the DFD for a simple ATM. The higher level DFD identifies two processes, Login and Perform Transaction. The Login process is broken down into three processes that identify the flow of data for logging into the ATM and the data that are needed for these processes to work together. In the data flow diagram, the circles represent processes and the arrows show how data flow throughout the system. Further information is added by including data stores, depicted by parallel lines. By modeling the major processes of the system and working toward lower level details, a complete logical model of the solution for the system can be produced and then partitioned into a logical implementation.

Design rules, defined in Table I, are a guide to developing quality software design. A developer cannot always follow each rule; often trade-offs must be made to accomplish the goals of the system. However, they do provide a heuristic for building logically structured software. Two essential guides to software design are to minimize coupling and maximize cohesion. Cohesion is the strength of relatedness within a module.

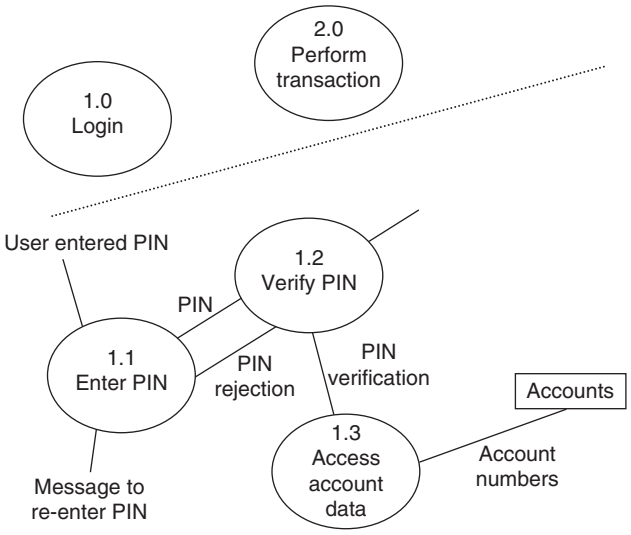


Figure 5 Data flow diagram for ATM.

Coupling is the strength of the relationship between modules. There are several advantages to designing systems this way. Effective modularity makes it easier to develop, test, and maintain the system. Keeping the codependence of modules to a minimum ensures that when one module is changed, the other modules in the system maintain their integrity. When each module operates as a "black box," many efficiencies are achieved. Further, the design is robust in that errors in one module do not degrade the entire design. Being able to isolate modules and correct them has the advantage of minimizing the amount of rework necessary when an error is being corrected or a requirement has been modified.

3. User Interface Design

Three types of interfaces exist when a software program is being designed. First, program modules must interact with each other. Second, the program interacts with external databases, outside sources of information, and computer peripherals. These interfaces are best depicted by the DFDs. The final type of interface is the human-computer interface. Defining how the user will operate the application is often key to acceptance of the application. A high quality user interface can improve productivity and reduce errors made by the user. Designing a user interface is complex. A wide range of people, with different levels of expertise and different views of the world, will use the software application. The user interface needs to accommodate novice and advanced users, it needs to provide for correction of user errors, and it needs to do this in an aesthetically pleasing way that meets the user's functional needs.

One approach to defining how users work is task modeling. Often, when a designer asks a user to describe their work, the user gives a general description of what they do. They have a great deal of difficulty articulating small details or nonfrequently occurring tasks. Often, users may fail to identify tasks that they perform daily because, from their perspective, it is so routine, they overlook it. Using task analysis is a way to overcome the inadequacies of a simple interview. Task analysis requires that the developer observe the user in the work place and model their tasks. This not only gives the developer first hand knowledge of what the user does, it shows the work process, it shows what data sources the user accesses and manipulates, and it shows how and what data sources users produce. Tools of task analysis can be simple step-by-step descriptions, flow charts of user activities, or hierarchical charts that organize user tasks.

The value of the task analysis is that it gives the developer tremendous insight into how the data are

Table I Design Rules for Structured Program Development

Rule	Description	Advantage
Functional independence	Develop modules with a single purpose and a minimal need to work with other modules.	Functional independence is supported by the rules of coupling and cohesion. Designing software with this in mind creates code that is easier to understand and maintain. It is robust in the sense that program modifications remain isolated to a module, minimizing effects to the entire system. This greatly simplifies rework.
Span of control	Higher level modules should control no more than seven subordinate modules.	Understanding the flow of control of the system is easier if it is broken into comprehensible units. When the number of submodules exceeds seven, it becomes cognitively difficult to understand.
Coupling —data —stamp —control —common —content	Minimize dependency and communication between modules.	Minimal coupling creates a robust design in the sense that an error in one module cannot effect the logic and processing of another module. If an error is located in a single module, it becomes easier to identify and fix without impacting other modules.
Reasonable size	Modules should be a reasonable size (50 to 100 lines of code).	This makes it easier for programmers and testers to understand the module. If the module is greater in size, the system may need to be broken down into lower level modules.
Cohesion —functional —sequential —communication —procedural —temporal —logical —coincidental	All instructions within one module should relate to the same function.	When modules contain several instructions, relating to different functions it is difficult to determine the purpose of the module or the flow of control when the outputs of the module are produced. Further, if one function needs to be changed, editing the module may impact both functions, requiring more rework.

processed by the end user, defining how the person will interact with the computer. However, user interface design also has an aesthetic quality to it. Many people will tell developers, "I don't know how to describe what I want, but I'll know it when I see it." This seems ridiculous, but when you consider the wide range of human preferences and tastes, it is easy to understand why this task is difficult. Regardless of user variation in aesthetic needs, there are several basic design rules that improve the quality of the user interface. Ben Shneiderman in 1987 defined 8 golden rules of user interface design, shown in Table II.

An additional technique for defining the user interface is that of prototyping. Building mock screens and menus that mimic what the system will do is a good way to gain user feedback early in the development life cycle. When the user can actually test a mock-system, they can give extensive verbal feedback as to whether or not the system actually satisfies their work needs. Further, their aesthetic needs will also become apparent. Letting the users see the interface early in the development ensures that when the final application is done, they are already familiar and satisfied with it.

4. Procedural Design

Procedural design defines how the logical structure of each module will operate to achieve the lower-level, functionality of the system. Three tools are commonly used to define the algorithms, flow charts, tables, and pseudocode.

Flow charts use graphic symbols to represent the programming logic for each algorithm. Figure 6 is a flow chart for calculating an average. At the beginning, a condition is checked to see if the user has entered a number. If the condition is determined to be true, state-

Table II Summary of Shneiderman's Eight Golden Rules of User Interface Design

Rule 1: Strive for consistency.	The application should be consistent in many ways. Using similar screens, naming conventions, icons, sequences of commands, and location of objects builds consistency into applications. High consistency improves productivity and shortens learning curve.
Rule 2: Enable frequent users to use shortcuts.	Systems need to be flexible so that novice users can get appropriate help and so that experienced ('power') users can use shortcuts to perform tasks more efficiently.
Rule 3: Offer informative feedback.	When a user performs an action, they should receive visual or audio feedback that the operation actually worked. An example of this is double clicking to open a file. Once the user has double clicked a file icon, they see an hour-glass indicating for them to wait until the page opens. Feedback gives the user confidence that their actions are controlling the system.
Rule 4: Design dialogs to yield closure.	At the completion of a series of operations, the user should get some confirmation that their task was accomplished.
Rule 5: Offer error prevention and simple error handling.	When possible, constrain the user so that errors are avoided. Some error prevention can be handled technically, giving the user informative feedback when they perform erroneous operations or data entry. Other errors can be prevented through design, by only allowing the user to select a constrained set of values with the mouse, instead of entering them.
Rule 6: Permit easy reversal of action.	Design systems so that operations can be reversed easily. This enables users to recover from mistakes and encourages them to explore other functionality of the system because they have confidence that they can recover their original data files.
Rule 7: Support an internal locus of control.	Make sure the language and sequence of messages used to help the user affirms the user's control over the computer. For example, "The computer saved your file" places the computer in control. A better message would be neutral, "Your file has been saved."
Rule 8: Reduce short-term memory load.	Design the interface so that users do not need to remember commands to operate the system. Seven plus or minus two is the number of items of information a user can effectively process in short-term memory. Chunking can help the user comprehend large amounts of information because they are categorizing into smaller groups. Pull down menus and visible controls can reduce the user's short term memory load.

ments are executed to calculate the sum and to count how many numbers have gone through the control structure. When the condition is determined to be false, the average is output to the user, as indicated by an output or print statement. Statements are depicted by rectangles; conditions are shown by diamonds. Flow of control is depicted by arrows and sometimes words. Flow charting is a tool that assists programmers in structuring the logic of the module before actually writing any code.

If a complex series of conditions exist, they are sometimes represented in a table. Decision tables that map out a variety of cases that the software could operate around may be a quicker way to model the logic than a complex, nested flowchart. For example, if logic to assign children's height to a size and weight were needed, a simple table might depict the information more easily than a nested flow chart.

Pseudocode, or structured English, is also useful in modeling something that may not be readily attainable in a flow chart. Pseudocode is a high level description of the programming logic. Perhaps a series of statements need to be executed to achieve a system goal. Modeling this in a flow chart would be tedious. Writing the logic in pseudocode is much more helpful to the person who will eventually code the program.

B. Structured Code

The coding phase of the SDLC is where the system is built or generated based upon the design documents and models. Structured programming languages include COBOL, Fortran, and Pascal. These languages are categorized as procedural programming languages because the programs they are written with follow a strict procedural hierarchy, giving instructions for how manipulation of data will occur. Procedural languages separate data from the program procedures. Modules are independent of the data, and data are typically shared globally by the entire program. Each module or function in the program reads data from the global file and then manipulates it and returns the altered values. This can make test and maintenance difficult.

To develop a large application, a driver program will control several subprocedures or functions to

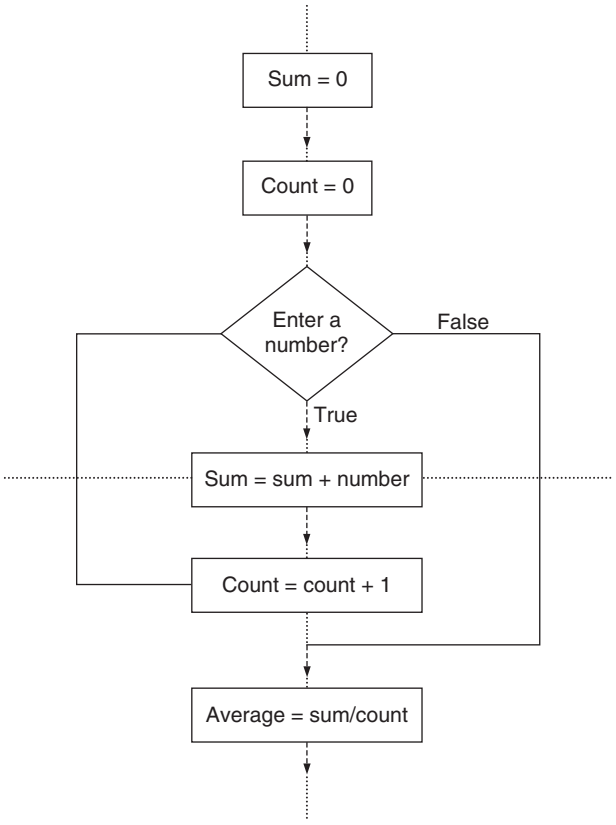


Figure 6 Flow chart for finding an average.

accomplish the solution found during structured design. However, the purpose of the coding phase is not just to code the design, but to code the design in such a way that the software has high quality. Several process controls can help to improve software quality. Among these are control of the programming environment, code and configuration management techniques, and software metrics.

A programming environment can consist of many things including hardware and software that help design and diagnose the software. Essential to this effort is a debugger. Programming languages come with debugging tools that either identify errors and point the programmer to the place where the error occurs or allow programmers to step through their logic until the error is revealed. Using a software lab or workstation as the production environment ensures that the operating system, hardware, databases, debuggers, versions of programming languages, and versions of software modules being developed are controlled in a single environment. This is important because while the software is under development, if an error occurs, the programmers need to be able to regenerate the error and figure out what caused it. If development was done in multiple environments with different configurations, isolating and duplicating an error would be extremely difficult.

The use of coding standards can improve software quality by ensuring that structure, size, and naming conventions are adhered to throughout the application. Coding standards ensure that everyone follows the same set of assumptions and presents their work in the same way. This is an efficient way for a software development organization to work because as people change responsibilities, all program elements will be familiar. All the programmers will know what each module should look like and where to find information about the logic and strategy that went before the coding of each module.

Configuration management is the control and management of software and software artifacts throughout the software life cycle. The configuration management process ensures version control and change control on all code, documentation, and data. There are many automated configuration management tools that keep code modules in a library and automatically track changes. The configuration management activity helps in monitoring the progress of the program development. For development of a software product, version control is very important. Imagine on a large project if several programmers kept altering code without knowing what other programmers were doing. The result would be disastrous. At some point, no one would understand all of the changes that were made and how they effect the entire system.

At some point, subsystems of an application are joined to form a baseline product. The baseline is the first mock-up of what the operational system will be. Naturally, there are still many errors with this program. As testing and rework correct these errors, new versions are released, forming a new baseline. Knowing what version has been tested and what errors have been remedied is essential to delivering a quality program. Control allows programmers the visibility to see if an implemented change has affected other portions of the system. If the change is not successful, it is easier to return to an earlier version.

Software metrics measure characteristics of the software throughout the software development life cycle. During the analysis phase metrics may focus on software complexity. During design, metrics can be used to determine how well software adheres to design rules such as modular independence. During the test phase, metrics can be taken to give a likelihood that testing is complete and the errors in the program have been found. Numerous software metrics have

been defined, but what is important is that the metrics add value and insight into producing high quality software. To this end, Ejiogu in 1991 defined characteristics of successful metrics to be simple and computable, intuitively persuasive, objective, consistent in use of units, programming-language independent, and providing quality feedback. Knowing what to measure and how to use the measures to improve software processes and procedures can build quality into the development process.

Numerous tools are available to help build and test the design model. Computer aided software engineering (CASE) tools provide graphic assistance to build DFDs, ERDs, and data dictionaries for structured programs. Additionally, the tools can provide for configuration management and error checking in the design and code generation. Powerful tools, when used properly to create a complete design, can generate thousands of lines of code, often creating gains in productivity. However, it is also common for the generated code to require alteration before it can operate properly. Some programmers feel it is easier to code a system by hand than to debug and recode software generated by a case tool.

C. Structured Test

In his seminal book in 1979 on software testing, Meyers states, "A successful test case is one that finds an error. An unsuccessful test case is one that causes a program to produce the correct result." The purpose of a software test is to identify any errors left in the system before it is implemented. A test that finds no errors on the first trial is probably not very rigorously designed.

Software testing takes place throughout the software life cycle at many different levels. Debugging a program can be viewed as the most basic kind of testing that occurs. As the application is developed and the modules are joined to form programs, and programs are joined to form systems, the type of testing changes from informal to formal tests. At the beginning of coding or construction, programmers perform debugging and unit level testing to ensure that the software is working correctly. As they build modules, they continue testing until they are satisfied that each program is working properly. This is considered as informal or unit-level testing. The tester is the developer, so one could argue that the test is not unbiased. The developer may keep a programming notebook in which they document testing that was done, or where to find data files that were used to test the program.

Programmers continue to test and integrate modules until they are satisfied with the resulting product. The objective of this lower-level integration testing is to systematically combine modules into subsystems and ensure that the modules work correctly together. Often, integrating subsystems uncovers problems not only with interfaces but with functionality of the modules as well. There are several ways to perform software integration. Most commonly, a top-down approach is taken. This is advantageous because a driver program usually controls the subprograms that make up the entire application. Prior to integration testing, a test suite of data is built, where all the outcomes are known and understood. Using a top down approach, a few modules are tested with the known data. As modules are added, the data are checked to ensure that the system is operating as expected. Should an anomaly occur in the generated data, the module where the miscalculation occurred is easily identified. A bottom-up approach to integration is also common. The idea is that the system is slowly constructed and tested to identify where errors exist.

When the programmers are satisfied with their integration testing, the software is baselined and formal change control begins. At this point, the errors found and the fixes to those errors are tightly controlled so that changes do not degrade the entire system. An independent test group becomes responsible for exercising the software and formally demonstrating that it meets the requirements defined in Analysis. This is known as validation and verification testing. Verification means that the software does what it is designed to. Validation means that it operates correctly. At this point in the test process it is important that the test group be independent from the developers. Developers are uniquely suited to test their programs. However, they are also so close to the development effort that they can easily miss something that an independent reviewer may find obvious. Having an independent group test software to the requirements is essential to software validation.

The independent test group begins work long before the software is coded. When the requirements are defined, the test group develops a test plan based upon the requirements. Care is taken in defining the requirements to ensure that they are testable. Each requirement needs to be validated as part of a formalized test. The independent test group writes a test plan during the analysis phase of the life cycle. During design, a test procedure is written, and at the test phase, the procedure is executed. Test procedures include precise tests to validate requirements in a controlled setting. In developing the test procedure, test

engineers use testing that exercises the software in nominal and extreme conditions. This is done by creating test data, test scenarios, and step-by-step procedures that thoroughly exercise the software. Since it is virtually impossible to test every condition, care is taken in the creation of the test data and procedures to ensure they thoroughly validate the software.

When errors are found during validation testing, they are formally identified and investigated so that a correction can be made. After the code is modified, the validation test (or a subset of the test) will be repeated to ensure that the software is now functioning properly.

1. Other Types of Testing

When one thinks about testing software, formalized testing over the life cycle is probably not what comes to mind. Beta testing and usability testing are more popularized types of testing that people are familiar with. Several other types of testing can occur during the software development life cycle:

- System Testing—Application software runs on hardware and needs to work with peripheral devices. The software and hardware together is the system. Often system testing will be performed formally.
- Usability Testing—This testing involves observing a user exercise the system. Often it is done in a usability test lab and the developers can observe how the user interacts with the user interface.
- Recovery Testing—Some system must be fault tolerant to ensure that lives are not lost if service is interrupted (software on an airplane). Recovery testing will purposely degrade the system to ensure that the system recovers in a prespecified amount of time.
- Security Testing—Systems that house private or proprietary information may require special tests to ensure that hackers or anyone else cannot access the sensitive information.
- Stress/Load Testing—This type of testing ensures that when the system experiences a greater than expected volume of traffic, it still operates successfully. Systems are typically designed for expandability. A stress test often proves that the system can handle the additional load.
- Burn-In Testing—Sometimes when a new system is developed, it will be run in parallel with an existing system. If it operates correctly within a specified time, the old system is eliminated.
- Alpha Testing—Tests performed by the end-users at the developers site. The end-user has expert knowledge about how the system will operate for them, and they can uncover problems the developers may not be aware of.
- Beta Testing—Tests performed by the end-users at their own workplace. Developers are not present for this type of testing. Feedback that the users give them help to correct the application before it is formally released.

III. STRUCTURED VS OBJECT ORIENTED METHODOLOGY

Structured design breaks a system down into functional modules. Each module can be described as having inputs, processing, and outputs. Structured design is a top-down decomposition of system functionality, while object oriented design focuses on system behavior. One can draw analogies between the two methods: a module is now an object and a program is now a class. However, there is a fundamental difference in the way that these programming methodologies operate.

Many principles of structured design, code, and test are similar to object oriented design, code, and test. During design, emphasis is placed on developing a program structure with independent components in a way that minimizes the need for connections between the components. Programming the system still requires the use of statements and control structures and testing still requires that an independent group verify that requirements have been met. Where structured and object oriented methodologies differ is in how they handle data. With structured programming, data are separated from the operations that manipulate it. Data are typically a global resource to the structured program and many modules can access and change the data, creating difficulty with data control and data integrity. In object oriented programming, the operations (methods) are encapsulated with the data. Each object can hide data from other objects. The nature of objects can ensure that only objects using specific data have access to it. This has some advantages in testing and maintaining the program.

The tools for building object oriented systems have also changed. While ER diagrams and data flow diagrams may still be used, the system model is built using unified modeling language (UML). An integrated UML tool assists developers in building the logical model of the system, defining and integrating class diagrams, collaboration diagrams, and sequence diagrams. These tools are discussed in the following section.

IV. OBJECT ORIENTED DESIGN, CODE, AND TEST

In the early 1990s, several object oriented analysis and design methodologies emerged, giving programmers a new paradigm for designing systems. Instead of focusing efforts on functional decomposition, the focus has shifted to describing systems by how they behave as real world objects. As stated earlier, the goal of the design phase is to take the findings of analysis and create a logical solution to the problem. In a sense, the designer is creating a blueprint for building the system. The object oriented development process is shown in Fig. 7. This is different from the structured process depicted in Fig. 3. The figure depicts that the process is truly iterative and that analysis, design, and test occur iteratively until the software is complete. During the analysis process, use case diagrams and descriptions and a preliminary class diagram are developed. As more modeling occurs, the definition of classes, attributes, and methods occurs and a prototype of the user interface is designed. When the system is constructed, developer testing occurs. During the analysis, design, and implementation phases, as more information becomes known, the system model is changed and enhanced to reflect the new knowledge. As shown in Fig. 2, the development process is cyclic, incorporating new functionality and newly discovered requirements as the software evolves into a fully operational system.

Before, analysis, design, and test were three distinct activities that did not interact or overlap. With object oriented development analysis, design, code and test develop together, so that solution subsets can be examined fully before adding further detail to the system. In writing structured programs, terms like program, module, subroutine, and function were used to describe software components. With object oriented programming, the idea of using components to build software is the same, but the terminology has changed to reflect the relatedness of the data to the components. An object oriented program is made up of:

- Classes—A template for an object
- Objects—Software representation of real world entities with characteristics
- Methods—Algorithms used to change objects
- Properties—Characteristics of an object

An example of this might be a Person class. A "Person" is a real world object that we can relate to and understand. The Person has attributes, first name, last name, gender, age, and height. We can create an object by defining its properties. For example, Ricardo

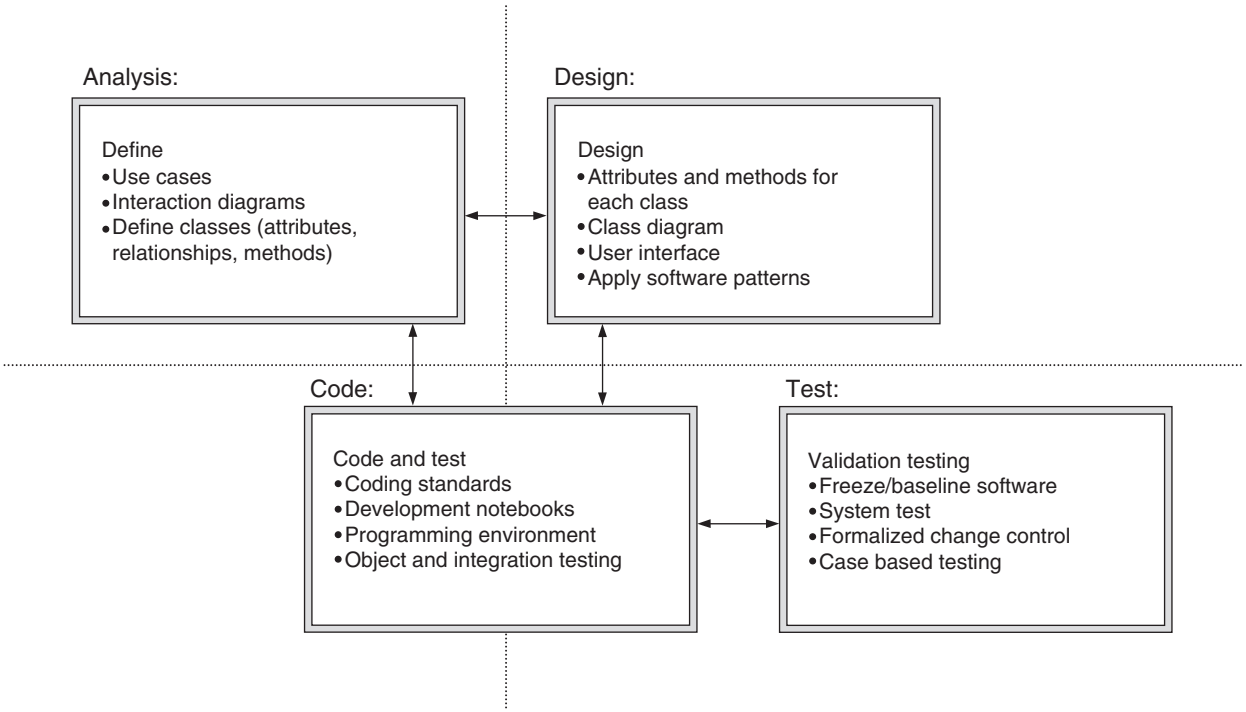


Figure 7 Object oriented analysis and design artifacts.

Sanchez, male, 25, and 5′10″. A method can change the object. So, if Ricardo has a birthday, a method would be used to change the Person object for Ricardo to "Ricardo Sanchez, male, 26, and 5′10″.

Object oriented programs also use the concept of inheritance to simplify programs. Inheritance means that a subclass inherits all the characteristics of a class. So if Male and Female were subclasses of Person, they would automatically inherit the attributes of the Person class. Object oriented programs also use encapsulation so that object details are hidden. An object's methods can be accessed by a user, but the details of the data and how the object operates are hidden from the user.

A. Object Oriented Design

The analysis phase of an object oriented development program begins with defining use cases. Use cases are high level descriptions of how users interact with the system. Use cases are behavioral descriptions of the system that are graphically and textually represented. From the use cases, classes, attributes, and methods become apparent. Several strategies are used to identify these elements from the use cases. Once these program elements are identified, they are modeled in class and interaction diagrams that show what the system is. After class diagrams have been constructed in analysis, the details of the classes need to be designed.

Several different processes have been identified for object oriented design. In structured design, data, architectural, user interface and procedural designs formed the design model describing how to implement the system. With object oriented design, the shift from structure to behavior is evident. Most object oriented design processes recognize four types of design work:

- Subsystem design—Partitioning the analysis model into subsystems. Divides the system into functional subsystems that support subsets of customer requirements.
- Class and object design—The detailed design of each class (this could be seen as a module in structured programming) with hierarchies from a high level to increasingly specialized classes.
- Message design—The details of how each software object will communicate with other objects.
- Responsibilities design—Data and algorithm design for each object, and the attributes and methods of that object.

The tool used to build object oriented software applications is the unified modeling language (UML). Prior to 1997, several founders of object oriented analysis had their own visual notations for modeling systems. In 1997, James Rumbaugh, Grady Booch, and Ivar Jacobson combined their notations to form what is now the standardized UML used in industry to model software, sometimes called the unified process of software development. The Rational Software Corporation was the first company to provide an automated tool to model systems using UML. The analysis model is represented by several documents and diagrams. These include

- *Use Cases and Use Case Diagrams*—Use cases describe in simple language how the system will operate. For example, if we are building an ATM machine, and the person wishes to withdraw cash, a narrative describing the process will be written and a use case diagram depicting the possible options will be developed. Figure 8 shows a simple use case diagram for an ATM machine.
- *Class diagrams*—The class diagram shows what software classes will exist in the system and what associations and attributes they will have. Using the UML notation, classes are indicated by rectangles divided into three sections. The top of the rectangle indicates the class, the center portion of the rectangle indicates the attributes, and the bottom of the rectangle is reserved for methods defined in the design phase. Associations between classes are depicted by lines with arrows and numbers defining the relationship between classes. The class diagram is a static view of the system. Figure 9 is a portion of a simple class diagram for an ATM machine. Classes are represented by rounded rectangles. Classes are divided into three sections. The top section identifies the class name, the middle section

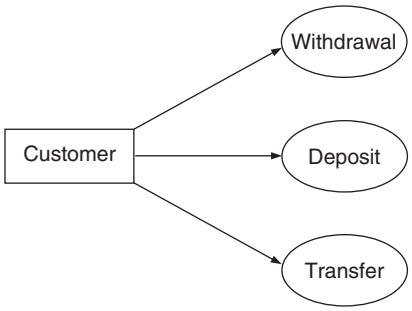


Figure 8 Use case diagram for ATM machine.

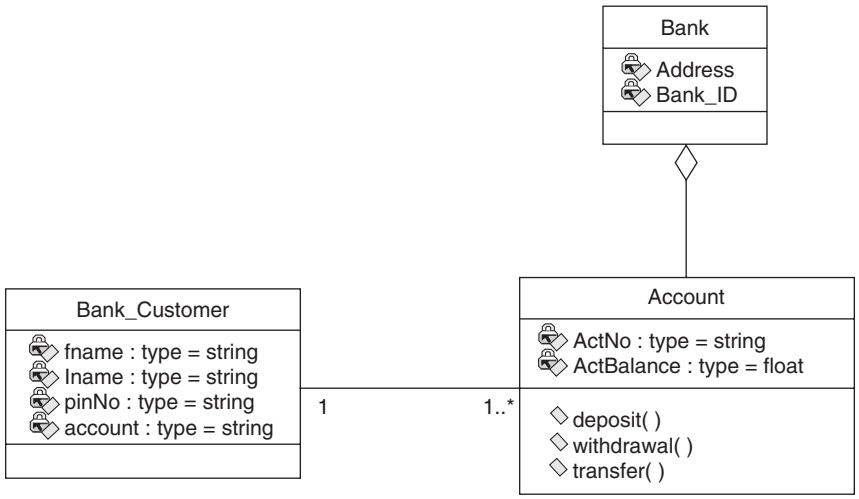


Figure 9 Class diagram for ATM machine.

identifies the attributes, and the lower section identifies methods within the class.

- *Sequence Diagrams*—This is an interaction diagram showing how messages travel through the system classes in a time-based sequence as shown in Fig. 10.
- *Statechart Diagrams*—A diagram that indicates the dynamic states of the machine. Events are modeled showing how the state of the system changes.

From the analysis model, the design (or how the system will be built) will begin. The design model is a direct extension of the analysis model, generating more detail to the model and creating new artifacts including:

- Collaboration/Interaction Diagrams—An interaction diagram that shows the structural organization of the system objects. How messages are sent and received and the creation of instances is depicted. Figure 10 shows the collaboration diagram for the ATM machine.
- Design Class Diagrams—The design class diagram is a more detailed depiction of the class diagram from Figure 9.

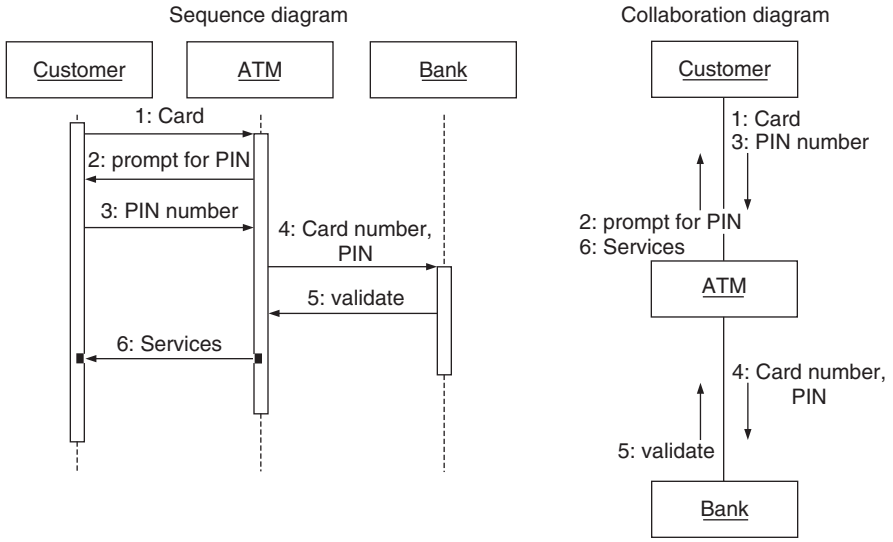


Figure 10 Collaboration diagram for ATM machine.

1. Object Oriented Design Rules

As with structured methodology, the goal is not simply to design a system, it is to design a system of high quality. Design rules for object oriented development are similar to those of structured development in that the developer wants to minimize coupling and maximize cohesion. Systems designed with functional independence, modularity, high cohesiveness, and low coupling are more reliable, maintainable, and testable. However, object oriented programming has an advantage in this area because the very nature of objects is to be modular and cohesive. The concept of encapsulation, where all the methods and details of an object are hidden to the other objects, creates a highly modular design. After the design model is developed, the task of system development turns to assigning responsibilities and applying design patterns.

A pattern is a recurring solution to a software problem. An experienced software engineer probably applies these patterns without actually thinking about them. They have developed enough software to know the errors that occur if specific patterns, or design rules, are not followed. The patterns are applied to interaction diagrams in such a way that object responsibilities and collaborations become apparent. The design rules help the developer to assign responsibilities in ways that produce quality software. Larman lists 9 general responsibility assignment software patterns (GRASP) that are commonly used in development. Five common software development patterns are summarized in Table III.

Table III Common Software Patterns

Pattern	Description
Expert	Assign the responsibility to the class that has the information to perform the responsibility.
Creator	Class B should create an instance of class A if they are tightly coupled with B controlling A in some way.
Controller	System events should be handled by class that: —represents system —represents organization —represents real world object involved in the task.
Low coupling	Assign responsibilities in a way that minimizes coupling. Measure of relatedness of elements among classes. Classes should be functionally independent.
High cohesion	Assign responsibilities in a way that maximizes cohesion. Measure of relatedness of elements in the class.

B. Object Oriented Coding

Object oriented coding is different from coding with procedural languages. Similarities exist in the way languages use control structures and declare variables. However, with object oriented programming, data and the methods used to manipulate the data are grouped together. This makes object oriented programming advantageous in that debugging and testing a program are less confounded. Since data are locally defined and manipulated, errors are likely to be isolated instead of impacting the entire software application.

As with structured programming, quality in object oriented coding can be greatly enhanced by a controlled programming environment, coding standards, and configuration management. Instead of using a CASE tool, object oriented software can be generated from a UML tool. If the design is defined well enough, this can be a great productivity tool. Since the development is iterative, making changes to the model and being able to regenerate the code can be highly efficient. Another great benefit believed to come out of object oriented programming is that of software reuse.

When object oriented methodologies were invented, reuse was viewed as a major advantage of using object oriented programming. Since objects were designed as encapsulated, black-boxes, they should be reusable by other applications. However, many opportunities for reuse have not been realized. Though developers agree with the concept, sometimes designing objects in a generic enough manner to be reusable is too time consuming at the time of design to be accomplished.

As with structured programming, metrics exist for object oriented programming too. The purpose and measures are similar, but the metrics in an object oriented program are taken over classes and subclasses. Still metrics for complexity and adherence to design rules are taken. However, the measures have been altered to fit the nuances of object oriented programming.

C. Object Oriented Test

Object oriented programs combine operations (methods), data, and properties of objects into classes. Methods may also be shared across classes. The complexity is exaggerated with the concepts of inheritance, when subclasses to a class inherit the characteristics and methods of that class. With procedural programming, module level testing focused on correctness of calculations. Class testing, the equivalent of unit level testing in structured programming, focuses on the correctness of

methods contained within each class. After developers are satisfied with the correctness of class level testing, integration and validation testing are performed.

Structured integration testing was performed incrementally, either as a top-down or bottom-up compilation of modules was done. Object oriented programs do not have the hierarchical structure that makes this type of testing sensible. Rather, programs are constructed by adding classes that interrelate based upon designed system behavior, not hierarchy. Use cases can be used to write scenarios that exercise the functionality of groups of classes. If an error occurs, it is the sequence of operations that drives discovery of the error. The principle of adding functionality and checking to see that the outcomes are as expected is still present. Another approach, use-based testing, first tests the independent classes and then progressively adds dependent classes, ensuring that as classes are added, the system works as expected.

Once the application is baselined, validation testing can occur. As with structured programming, a test plan and procedure were written prior to the coding of the software. The plan and procedure were developed to prove that the requirements for the application had been satisfied. In determining the requirements and scope of the system, use cases were developed describing how the system would operate. These use cases can provide a starting point for validation test planning. The validation testing proves that the application has met the requirements for which it was designed.

V. SUMMARY

Design, code, and test are three phases of software development where processes and techniques can greatly improve the quality of the software. Using structured or object oriented techniques, design is accomplished by modeling the system in such a way that modules or classes act as black boxes, or wholly contained entities that process data to achieve the goals of the system. As modules and classes are developed, a detailed design emerges, defining exactly how to implement the solution for the system. Both methods seek to maximize cohesion and minimize coupling. Accomplishing this creates software that can be more easily coded, tested, and maintained. While both methods share some commonality, they differ in many ways too. With structured design data are independent of software modules. Programs operate by calling subroutines to process data. Object oriented methods combine classes, methods, and data into single objects. Structured techniques use CASE tools to build system models. Object oriented design uses UML tools. Both types of tools model systems and can be used to test design and generate code. However, artifacts created by CASE tools are hierarchical and based upon process, whereas UML tools model the system based upon behaviors and messaging between system components. Testing techniques vary between the two methodologies as well. A structured test is based upon systematically building the software and testing as it grows. Object testing is more integrated in that nonhierarchical classes are exercised via use cases or use case scenarios. Validation testing for both methods is similar in that it attempts to prove that the system meets the requirements for which it was designed. New and better techniques for design, coding, and testing are still emerging. Regardless of methodology, the goal of software design, code, and test is to improve software quality through the use of well defined processes and techniques.

SEE ALSO THE FOLLOWING ARTICLES

Computer Assisted Systems Engineering (CASE) • Control and Auditing • Database Development Process • Documentation for Software and IS Development • Error Detecting and Correcting Codes • Quality Information Systems • System Development Life Cycle • User/System Interface Design

BIBLIOGRAPHY

Boehm, B. (1988). A spiral model for software development and enhancement. *Computer,* Vol. 21, No. 5, 61–72.

Booch, G. (1994). *Object-oriented analysis and design,* 2nd ed. Redwood City, CA: Benjamin–Cummings.

Coad, P., and Yourdon, E. (1991). *Object-oriented analysis,* 2nd ed., Englewood Cliffs, NJ: Prentice Hall.

Jacobson, I. (1992). *Object-oriented software engineering.* Reading, MA: Addison–Wesley.

Meyers, G. (1979). *The art of software testing.* New York: Wiley.

Miller, G.A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Science,* Vol. 63, 81–97.

Larman, C. (2000). *Applying UML and patterns, an introduction to object-oriented analysis and design.* Englewood Cliffs, NJ: Prentice Hall.

Rambaugh, J., *et al.* (1991). *Object-oriented modeling and design.* Englewood Cliffs, NJ: Prentice Hall.

Shneiderman, B. (1987). *Designing the user interface, strategies for effective human-computer Interaction.* Reading, MA: Addison–Wesley.

Programming Languages Classification

Charles Shipley and Stephen Jodis
Armstrong Atlantic State University

GLOSSARY

binding Creating an association between an entity and some property of that entity.
high-level language A language that is relatively removed from machine language and better able to express the abstract steps of a solution to an application problem.
iterative statement A statement form that specifies the looping through of a series of statements until some condition is satisfied.
low-level language A computer language that is relatively close to machine language.
machine language A language that is "understood" directly by a computer processor; statements in almost all machine languages are nonsymbolic and consist of strings of zeros and ones.
recursive function A function whose definition includes the possibility of invoking itself.
semantics The meaning of the constructs in a programming language.
source program Program that is considered as input to a translation process that will convert the source into a different (usually lower) language.
syntax The correct form of the constructs in a programming language.
translation The process of converting a statement in one language into a statement or set of statements in another.
von Neumann architecture Fundamental architecture for digital computers featuring a central processor, primary memory with cells that are sequentially addressed and that can contain both data and instructions and a default flow of control that is sequential.

THERE ARE MANY KINDS OF LANGUAGES that have been used on computers. This article discusses three different schemes that can be used to classify the wealth of languages that have been used for programming computers.

I. CLASSIFICATION BY GENERATION

Programming languages are sometimes classified by generation. In the very earliest days of computer programming, programmers were required to couch their instructions directly in the machine code "understood" by the hardware central processing unit (CPU). Such instructions consisted of strings of zeros and ones representing the operations to be performed and the addresses of the operands. Coding in binary machine code was extremely tedious and prone to error. Not only was the programmer required to learn the bit patterns representing each of the operation codes, the programmer also had to decide (and remember) where in memory to put each of the quantities that were relevant to the particular program. Thus, for example, if the program required that the tax be added to the purchase price in order to arrive at the total due, the programmer would have to decide where to put the "tax," the "purchase price," and the "total due." If the operation code for "add" was 33, written in binary as 00100001, and the programmer decided to put the tax in memory location 44 (binary 00101100), the purchase price in memory location 56 (binary 00111000), and the total due in location 72 (binary 01001000), then the machine code

instruction to calculate the total due might look something like

00100001 00101100 00111000 01001000

where the instruction would mean "add the contents of the first address to the contents of the second address, and put the results in the third address." Imagine making a mistake and having to trace through hundreds or thousands of such instructions in order to find an error. So these first generation languages were soon supplanted by a symbolic counterpart called assembly language. In this second generation, a programmer could give memory locations names, and the locations could be referred to by name instead of by address. The operation codes also have symbolic names, so the machine instruction of our example, above, might in assembly language look like

ADD TAX, PRICE, TOTAL

It is important to understand, however, that such an assembly language instruction cannot be directly executed by a CPU. Instead, another program must translate the symbolic assembly into binary machine code.

Machine languages and assembly languages share the property that any given language is tailored to a particular CPU (or family of related CPUs). Thus, for example, machine code for an IBM mainframe will not be understood by a microcomputer with an Intel CPU. Third-generation languages address this limitation by using constructs that are removed from the details of any particular CPU, and that reflect more closely the thought processes that are typical of problem-solving algorithms used by humans. Hence, the third-generation languages (and later) are often called high-level languages (HLLs) or problem-oriented languages (POLs). Languages such as FORTRAN, COBOL, and Pascal are typical third-generation languages.

Fourth-generation languages (4GLs) were developed in an effort to make languages even more problem-oriented than the third-generation languages. This was a result of frustration on the part of some business users with large conventional languages such as COBOL. Such users were often not professional programmers, but they wished to obtain quick results from data stored in a computer. One of the key features of 4GLs is that they contain nonprocedural aspects. First, they permit the specification of what is to be done without specifying how to do it. Second, they often permit the specification of a condition of when something is to be done, without requiring that the check for the condition be explicitly invoked at each point in the code where in fact the condition might be recognized and acted upon. To clarify these points, consider the following fragment of a typical 4GL used for report writing.

```
MAIN
BEGIN
    AT TOP OF PAGE
         DO headings;

    SELECT FROM subscribers
         SORTED BY subscriber;
    FOR EACH subscribers
         DO details;
END
```

Here, the program asserts that the records in the file "subscribers" should be provided in sorted order by using the field "subscriber," but the algorithm for sorting is not asserted, thus providing an example in which *what* is specified, but not *how*. The fragment also illustrates the second feature, since it specifies that the "headings" routine is to be executed at the top of a page of output, but it is not necessary to provide code that checks—as each line of output is created—whether that line is indeed at the top of the page.

No 4GL is fully nonprocedural. Indeed, in the final analysis all steps of all processes must be fully specified—by a process of translation from the language of the program to the native (first-generation) machine language of the computer. The decision as to which things to make nonprocedural is dependent upon the problem domain for the language—different languages intended for different specialized purposes choose different nonprocedural aspects—and different 4GL languages specify the "what" in different ways. No 4GL has gained more than limited ad hoc status. Most have been developed for a particular computer or family of computers, and for a particular application or type of application. Hence, the entire category of 4GL languages is often ignored in comparative programming language textbooks. Beyond the common feature of being nonprocedural, it is difficult to discuss 4GLs, given their number and variety. We can say, however, that most 4GLs involve one or more of the following:

- Application generators—Provide a language for *describing* routine applications; a description would then be used to generate an application in some third-generation language. Note that the description is typically nonprocedural. Typical operations would include data entry and updating of files or databases.

- Query languages—Used with databases, allowing the user to ask questions relating to one or more of the fields in the data records.
- Decision-support languages—Usually business oriented and used in the planning stage for analyzing projected financial investment.

As the above list might indicate, when discussing 4GLs the distinction between a *language* and *package* is not always clear.

The distinction is even less clear when we discuss fifth-generation languages. The category stems from an ambitious and wide-ranging software project launched by Japan in the early 1980s, called the Fifth-Generation Computer project. It proposed to develop the technology for automatic language translation, the construction of knowledge-based systems, speech recognition, and automatic software production. Many of these areas must exploit parallelism in order to have a chance of success. The language commonly used for writing software in these areas is Lisp, which, like other artificial intelligence languages, is sequential. The Japanese efforts centered upon using Prolog for artificial intelligence applications, because Prolog can be modified so as to make it highly parallel. Note that Lisp and Prolog are firmly rooted in the third generation of computer languages. It is the ambitious *application* areas that are new and fifth generation.

II. CLASSIFICATION BY APPLICATION AREA

A second way of classifying programming languages is by area of application. Such an approach would yield categories such as

- *Commercial languages,* which are particularly concerned with manipulation of files of alphanumeric data and with the production of reports
- *Scientific languages,* which are used mainly for the manipulation of numeric data
- *Systems programming languages,* which are particularly suitable for writing operating systems
- *Command languages,* which are used as an interface between the computer user and the operating system
- *Special purpose languages,* which are designed with a limited objective for some specific application area, such as controlling an industrial robot
- *Data-flow languages,* which permit the flow of data between the statements of a program to be examined so that inherent parallelism is revealed, permitting the program to be executed on a specialized machine that can take advantage of the parallelism to achieve increased speed

III. CATEGORIES OF HIGH-LEVEL LANGUAGES

As a glance at the sources in the bibliography will reveal, modern texts on comparative programming languages do not emphasize generations or application areas. Instead, the focus is on the categories of the high-level languages based on differences that are intrinsic to the languages themselves.

Before we discuss the different categories of programming languages, we will first mention what they have in common. All of the high-level languages assume that their programs will be translated into a machine language prior to being executed. The program to be translated, known as a source program, will be presented to the translator as input, and will be in the form of text lines. The text will be represented in some standard character set, such as ASCII. The individual characters are the atomic entities of source programs. The translators recognize compositions of the characters into lexical units called tokens, and compositions of tokens strung together to form source programs that follow the rules of the language syntax. Finally, each syntactically correct program has some meaning or performs some action, as defined by the semantics of the language.

A. Imperative Languages

1. Von Neumann Architecture

The von Neumann architecture—the fundamental architecture upon which nearly all digital computers have been based—has a number of characteristics that have had an immense impact on the most popular programming languages. These characteristics include a single, centralized control, housed in the central processing unit, and a separate storage area, primary memory, which can contain both instructions and data. The instructions are executed by the CPU, and so they must be brought into the CPU from the primary memory. The CPU also houses the unit that performs operations on operands, the arithmetic and logic unit (ALU), and so data must be fetched from primary memory and brought into the CPU in order to be acted upon. The primary memory has a built-in addressing mechanism,

so that the CPU can refer to the addresses of instructions and operands. Finally, the CPU contains a register bank that constitutes a kind of "scratch pad" where intermediate results can be stored and consulted with greater speed than could primary memory.

2. Characteristics of Imperative Languages

The imperative languages have three characteristics which directly reflect aspects of the von Neumann architecture. First, there are variables , which are named memory cells. Second, there are assignment statements, which put values into those named cells. Finally, there is a reliance on a particular method of achieving repetition called iteration.

a. Variables

Most, but not all, imperative languages require that the variables be declared. A declaration establishes the existence of the variable, and, more importantly, establishes the type of the variable, so that the program can correctly interpret the bit patterns that will be placed into the variable. For example, in the language Pascal, a variable to hold the count of how many employees had been processed in a payroll program, and a variable to hold the various pay rates, might be declared as

```
VAR
Count: integer;
PayRate: real;
```

b. Assignment Statements

Assignment statements place values into variables. For example, to place an initial value of 0 into the variable "Count," Pascal would use the assignment statement

```
Count := 0;
```

c. Iteration

Computers owe their utility not only to their ability to execute instructions rapidly, but also to their ability to be told to repeat a set of instructions. Thus, for example, processing a payroll for 100 employees does not require that the computer execute 100 sets of payroll instructions, but rather that the computer execute the payroll instructions 100 times. On a von Neumann machine the payroll instructions would be stored in consecutive locations in primary memory. Iterative repetition would entail repeating the same sequence over and over and checking between sequences to see if the sequence should be repeated again. On a von Neumann machine, it is easy (quick) to get back to the start of such a sequence, and the checking, to see if the repetition itself should cease, is often made very fast by the use of registers. An example of an iterative structure to accomplish repetition in Pascal, for example, is provided by

```
for i := 1 to 5 do
      Sum := Sum + 2*i;
```

Here, the variable "i" takes on each of the integral values from 1 to 5, and the value of the expression "Sum + 2*i" is assigned to Sum for each of those values. Such a statement is called definite iteration, since the action is repeated for a specifically determined number of times. Definite iteration is a common form of iteration in imperative languages. Such languages also provide one or more nondefinite forms, called conditional iteration.

A simple example from Pascal is

```
Sum := 0;
read( Current);
while Current <> -999 do
    begin
          Sum := Sum + Current;
          read (Current)
    end;
```

Here, integral values are repeatedly obtained from the keyboard, with the value "-999" serving as a signal that the looping should cease. So long as that value has not been read, the number is accumulated in the running total maintained in "Sum."

Thus, the most popular programming language paradigm—imperative—reflecting the most popular computer architecture—von Neumann—is characterized by three features: an emphasis on variables, an emphasis on assignment statements, and a dependence on iteration to accomplish repetition.

3. Categories of Imperative Languages

a. Procedural Languages

Until recently, the most popular of the imperative languages have been procedural. FORTRAN, Algol-60, C, and Pascal are but a few of the many languages in this category. The name of this category can lead to confusion. Programs in these languages have the ability to abstract processes, that is, the ability to associate a specific set of steps with a name, and an ability to invoke those steps by invoking that name. The discrete syntactic entity that defines the set of steps is called a procedure. But it is not the existence of syntactic procedures that gives rise to the category designation. It is not the ability to abstract a set of steps and give it a name. Rather, it is that the exact se-

quence of steps must be specified in the definition of the program (and its procedures.) That is, how a process is to be done must be fully specified. As an example, if we wanted to execute a sequence of 10 steps, but only so long as a certain condition were true and remained true after each step, but we wanted to exit the sequence as soon as the condition became false, then in a procedural language we would need to explicitly check the condition after each of the 10 steps.

Programs in procedural languages have an emphasis on the steps required to accomplish the desired purpose. An illuminating analogy often employed by instructors of procedural languages is that a program is like a recipe. First, the ingredients are listed, and then the steps which manipulate the ingredients are detailed in the appropriate order. The steps manipulate the ingredients, just as the instructions manipulate the data. Although there are two aspects to a program—its data and its manipulative steps—the relative emphasis is revealed by the category name: procedural. That is, the data are conceptualized as passive agents that are acted upon by the algorithms expressed by a sequence of instructions.

The necessity to specify the details of the manipulations results in procedural languages generally having a rich set of instruction types for controlling the sequencing of instructions. For example, procedural languages will need to be able to test conditions and then branch to different sequences of statements depending upon the result of the tests. Such statements are called "selection" statements. As we have seen, procedural languages will also have one or more types of "repetition" statements, and these statements will generally be iterative in nature.

b. Object-Oriented Languages

The second subtype of the imperative languages is the object-oriented languages. This category is often classified as a distinct type, rather than a subtype of imperative. We do not do so here, because even though there are significant and important differences between the procedural languages and the object-oriented ones, most object-oriented languages share the three ear-marks of imperative languages: variables, assignment statements, and iterative control of repetition. How do object-oriented languages differ from their imperative kin? First, in general terms, by emphasizing data rather than algorithms; and second, and more specifically, in three essential ways expressed by three technical phrases: data abstraction, inheritance, and dynamic dispatch.

i. Emphasis on Data In procedural languages the pattern is to declare the variables that are to be manipulated, and to then describe the steps involved in the manipulations. The sequence of steps is called an algorithm. In complex applications, the data are often structured. For example, the application may involve data in such a manner that some of it is conceptualized as being in a list, some of it is conceptualized as being the leaves on a branching tree, and some of it is conceptualized as being arranged as a stack, with only the top element immediately accessible. In such a situation, the algorithm for the application is likely to involve steps that involve more than one data structure. This would make it difficult to reuse some of that same algorithm for a different application, because the data structures would be likely to be different as would the interactions between the data structures. Object-oriented programming attempts to overcome this difficulty by conceptualizing the data of an application as objects that have behaviors. These behaviors called "methods" are elicited by the reception of communications, called "messages," from other objects. Algorithms now consist of defining appropriate objects and then specifying the sequence of messages they send to one another. The advantage of this approach is that if an object is defined appropriately, then that object or one similar to it is likely to be present in some other application, and hence the code that describes that object's behavior can be reused.

ii. Data Abstraction One of the most important components of object-oriented programming is that of data abstraction. Data abstraction involves the defining of a particular data type by a grouping, called encapsulation, of manipulative functions and procedures with the data to be manipulated, in such a fashion as to hide unnecessary details from units outside of the encapsulation. In addition, since the underlying representation of the type is hidden from the program units that use the type, the only manipulation of the data is via the operations that are explicitly provided by the definition of the data type.

In object-oriented languages, the mechanism by which the data type is grouped with the manipulative functions and procedures is by defining classes. The entities in a program are objects, which are instances of classes. The class is the repository for the behavior associated with an object, in the sense that all objects that are instances of the same class can perform the same actions.

iii. Inheritance Abstract data types, realized as classes, with their encapsulation and enforcement of access privileges, provide an obvious choice for entities to be reused. But the problem is that in new situations the old definitions are seldom exactly right.

Inheritance provides a solution to this problem. Inheritance permits the creation of a new abstract data type by "inheriting" the data and operations of the old, but allowing the new type to add additional data elements and to redefine or add methods. When a new class is defined in such a manner, it is said to be "derived" from the old class. Thus, classes can be organized into a hierarchical inheritance tree, and we can speak of "parent" and "child" classes.

iv. Dynamic Dispatch In computer language theory, "dynamic" refers to events that occur during the execution of a program, as opposed to, for example, during the translation of the program into machine code or during the process of loading the machine code into memory for execution. "Binding" is a general term in computer science and, when used as a noun, refers to an association, such as between an entity and some property of that entity. When used as a verb, binding refers to the act of creating such an association. So "dynamic type binding," for example, would mean that the type of a variable is not determined by a declaration statement that can be analyzed at compile time, but is rather determined at run time (such as at the time of an assignment to the variable.) "Dynamic dispatch" refers to the dynamic binding of messages to method definitions. That is, the specific method that is called by sending a message to an object is determined at run time rather than compile time. The purpose of this is to achieve the flexibility provided by a kind of polymorphism. The idea is to facilitate the construction of code that can perform operations on generic class objects. Less technically, polymorphism is the ability for the same program to work for more than one type of object. If an operation can be applied to objects of any class that is derived from the some base class, then this generic code will not have to be changed if new derived classes are created.

To demonstrate the utility of dynamic dispatch, we consider a brief example. Suppose we have defined a "shape" class and have subsequently defined several different classes that are derived from the shape class, such as rectangle, circle, triangle, and so forth. For each of these derived classes, we could have a "draw" method by which an object could know how to draw itself. Now suppose that we are interested in creating "lists" of shapes, and that we wish to be able to traverse such a list and have each element in the list draw itself. If each of the elements of any list are all of the same shape, then the language translator can determine statically which "draw" method to use. But what if we need a list of shapes and each shape could be any of the derived shapes? There are solutions to this problem that do not use dynamic dispatch, but they are all awkward and lack an important property: the ability to add additional types of shapes and to not be required to change any of the existing code for traversing a list and having it print out the constituent derived shapes. For an elegant solution that adds the property, we need a generic description of traversing a list and dynamic determination of the correct draw method, and that is exactly what dynamic dispatch enables. The important point here is that the flexibility exhibited in this example is not artificial and is not limited; it is just as valuable in many applications having nothing whatever to do with shapes or lists.

B. Functional Languages

Functional programming languages are based on mathematical functions and function composition. They are sometimes called applicative languages because their programs are based on the definition and then application of functions. The functional programming paradigm is one of the most widely used alternatives to the imperative paradigm. In its pure form, this style of programming uses function application as the only control structure, as opposed to the control structures for selection and repetition that are found in imperative languages.

Functional language programming is done by constructing functions based on previously developed functions in order to build more complex functions that transform an initial dataset into an answer. Walking through the code is a matter of evaluating the successive functional transformations that are made on the data to arrive at an answer. This is in contrast to imperative languages in which one considers the successive state changes caused by individual statements.

We should note that as some of the functional languages have evolved, they have taken on concepts from the imperative languages. Thus, although they are not really necessary, many (but not all) functional languages have added assignment, selection, and iterative looping constructs, although they take on different forms. But in so far as a functional language is a pure functional language it will not use variables or assignment statements. Nor will it use iteration. Instead, it will accomplish repetition by using recursion.

Briefly put, a recursive function is a function that invokes itself in the course of its execution. A simple example might serve to clarify the distinction between iteration and recursion. Consider the "problem" of describing a procedure for how to walk up a flight of 20 stairs. An iterative solution might be

```
Procedure Walk-up-steps;
Begin
```

```
      While (not at the top)
           Take a step;
End
```

Another iterative solution might be

```
Procedure Walk-up-steps;
Begin
    Assign 0 to count;
    While (count < 20 ) do
         Take a step;
         Add 1 to count;
End
```

A recursive solution might be

```
Procedure Walk-up-steps;
Begin
    If (not at top) then
         Take a step;
         Invoke Walk-up-steps;
End
```

The fundamental point to be understood from the above examples is that recursion provides an alternative method for achieving repetition. What the simple examples do not reveal is that there are many problems whose solutions are easier to describe and/or invent by using recursion than by using iteration. The emphasis on iteration in the imperative languages springs primarily from the fact that iterative solutions (if they can be found!) are invariably executed more efficiently.

We will next consider in brief some of the more popular functional languages.

1. Lisp

The Lisp language is the oldest and most widely used functional language. Lisp is essentially typeless, but originally had two types of data objects: atoms and lists. Indeed, Lisp stands for "LISt Processing." Lisp has long been a popular language for applications in artificial intelligence. Lisp dialects today contain imperative style variables, assignment statements, and iteration yet still maintain many of the fundamental concepts of functional programming. Scheme is a well known dialect of Lisp that was created at MIT. It is small in size and is available on many platforms. Common Lisp is another dialect of Lisp. It was created in the 1980s with an objective to unite many of the Lisp dialects into a single language. Hence, Common Lisp is quite large and complex, yet it retains the basic nature of Lisp. Common Lisp data structures include records, arrays, complex numbers, strings, and others.

2. Other Functional Languages

ML (meta language) is a functional language that was developed in the 1980s at the University of Edinburgh. It differs from Lisp in many ways. It is strongly typed, uses a syntax similar to Pascal (an imperative language), includes exception handling, and has the capability to implement abstract data types. ML contains lists—as does Lisp and its cousins—but their appearance is different than that found in Lisp. ML also has arrays, enumerated types, and tuples (records). It has type declarations, is strongly typed, and uses type inferencing (variables do not need to be explicitly declared).

Haskell is a pure functional language. Haskell is a strongly typed language which uses a syntax similar to ML. It too includes type inferencing. It distinguishes itself from ML in that it is purely functional. The language has no variables or assignment statement, allows no side effects, and does not contain imperative features of any kind. Haskell uses an evaluation type known as lazy evaluation. Lazy evaluation means that a subexpression is not evaluated until it is known that its value is required. Haskell also provides a method for constructing infinite lists called list comprehensions.

C. Logic Programming Languages

As the designation suggests, "logic programming" languages are based on symbolic logic. Programs consist of statements, or propositions, that are stated or declared. Execution of such a program consists of using logical inferences to obtain results. There is another important sense in which logical programming languages are "declarative," and that is in the sense that they are nonprocedural. That is, only the desired results are stated rather than the detailed procedures for obtaining them.

Perhaps these defining characteristics can best be made clear by discussing the most popular of the logic programming languages: Prolog. Prolog programs, like programs in other languages, consist of a set of statements. In Prolog, the statements are constructed from terms. A term is a constant, a variable, or a structure. The constants are either atoms or integers. If we define an "identifier" to be a string of letters, digits, and underscores that begins with a lowercase letter, then we we can say that an atom is a symbolic value which syntactically is either an identifier or is a string of printable characters that is "quoted" by apostrophes. A variable is an identifier that can be bound to a type and a value during the process of drawing inferences. A structure is of the form

functor(parameter list)

and represents an atomic proposition of predicate calculus. Here, the functor would serve to identify the structure and is any atom. The parameter list can be any list of atoms, variables, and other structures.

The statements of a Prolog program can be divided into three categories: statements of fact, statements of rules, and statements of goals. The statements of fact, in the simplest case, would be simple propositions in the form of a single structure whose argument list consists solely of atoms. The interpretation is that the atoms have the property named by the functor. For example,

```
state(georgia).
town(savannah).
county(chatham).
is_in(savannah,chatham).
is_in(chatham,georgia).
```

are five propositions asserting three factual properties and two factual relationships.

The statements of rules correspond to propositions of logic of the form "if p then q," written in Prolog as

```
q :- p.
```

In Prolog, the conjunction of two propositions is indicated by separating those two propositions with a comma, so

```
state(georgia),    town(savannah).
```

asserts that Georgia is a state *and* Savannah is a town.

Variables can be used in Prolog statements in order to generalize their meaning. Thus,

```
is_in(X, Z) :- is_in(X, Y), is_in(Y, Z).
```

could be used to express the general property that the relationship "is_in" is transitive.

The third category of Prolog statements corresponds to propositions that we want the system to either prove or disprove. These propositions are called "goals" or "queries" in Prolog. The syntax for a goal is the same as that for a fact. Consistent with our example, above, we could assert the goal

```
is_in(savannah,georgia).
```

The system would respond with either yes or no. Since conjunctions and propositions with variables (called propositional forms) are also legal goals, we could also assert the goal

```
is_in(savannah, X).
```

In this case, the system would not only respond with a yes or no, but also—since the response is a yes—with an instance of X that makes the propositional form true.

Table I Categories of Languages

Procedural	Object-oriented	Functional	Logic
Ada 83	Ada 95[a]	APL	Prolog
ALGOL 60	C++[a]	Common Lisp	
ALGOL 68	Delphi[a]	FP	
BASIC	Eiffel[a]	Haskell	
C	Java	Lisp	
COBOL	Oberon	ML	
FORTRAN	Modula-3	Scheme	
Pascal	Smalltalk		
Modula-2	Visual Basic		
PL/I			
SIMULA 67			

[a]Hybrid languages created by adding object-oriented support (as well as other things) to existing procedural languages.

Since the syntax of facts and queries is the same, how can the system differentiate? Different dialects of Prolog do so differently. One method, common for interactive forms of Prolog, is to enable the user to toggle modes—one for entering facts and rules and the other for entering goals.

IV. CLASSIFICATION OF A NUMBER OF LANGUAGES

Table I lists a number of programming languages and their respective categories.

SEE ALSO THE FOLLOWING ARTICLES

Artificial Intelligence Programming • C and C++ • COBOL • FORTRAN • Goal Programming • Java • Object-Oriented Programming • Pascal • Simulation Languages • Visual Basic

BIBLIOGRAPHY

Clark, R., and Wilson, L. (2001). *Comparative programming languages.* 3rd ed. Reading, MA: Addison–Wesley.

Friedman, D., Wand, M., and Haynes, C. (2001). *Essentials of programming languages.* 2nd ed. Cambridge, MA: MIT Press.

Pratt, T., and Zelkowitz, M. (2001). *Programming languages: Design and implementation.* 4th ed. Englewood Cliffs, NJ: Prentice Hall.

Sebesta, R. (2002). *Concepts of programming languages.* 5th ed. Reading, MA: Addison–Wesley.

Project Management Techniques

Bennet P. Lientz *University of California, Los Angeles*
Kathryn Rea *The Consulting Edge, Inc.*

GLOSSARY

collaborative project management Team members work on defining and reporting on their own work; team members work together on tasks and issues.
issues Problem or opportunity within a project.
lessons learned Specific idea that can be used in a project that results from experience and the extraction of knowledge.
project Set of tasks and milestones that produce a specific end product.
project concept Objectives, scope, roles, issues, and general schedule for a project.
project management Techniques used for managing projects.
template High-level schedule consisting of high-level tasks, dependencies, and general resources.

AN APPROACH FOR MODERN PROJECT MANAGEMENT is presented in this article. Modern project management takes advantage of both modern technology and tools as well as lessons learned from experience from past projects. In modern project management you aim for cumulative improvement in projects in terms of results, schedule, and cost. In modern project management there is an emphasis on collaborative work among team members. There is also an emphasis on issues and getting these resolved. Issues are associated with tasks in the project plan. Tasks with issues are labeled as risky. Project plans are built using project templates. Project templates act as a repository for lessons learned and experience. A project template includes high-level tasks, general resources, and dependencies for a class of projects. You then use the template and create detailed tasks with specific resources and times and durations for tasks.

I. INTRODUCTION

A. Review of the Nature of IS Projects

Surveys have shown that the rate of success of information systems (IS) and related types of projects is very low. This continues despite new technology and tools and despite experience from many projects of the past.

1. Why IS Projects Are Different Than Standard Projects

IS and information technology (IT) projects are quite different from standard projects. A major reason for failure of IS projects has been that IS projects have been treated the same as standard projects. Here are some key differences of IS projects.

- IT projects have ill-defined boundaries. In many IS projects the scope of the project keeps expanding as work on the project goes on. There is an active need to control the project of scope creep and project drift out of control.

- The role of business units is critical in IS projects. Many IS projects attempt to get a few business staff involved in projects full time to collect requirements and business rules. Business rules are very critical to IS projects. This is less so in many other projects.
- Staff in IS projects often have other work. There is on-going maintenance, enhancement, and operations support of current systems. There is network and systems software support. The same people who perform these tasks are critical to the IS projects. People often must be shared among projects as well. If IS staff on a project are treated as full time on the project, the project will deteriorate.
- There are more opportunities for teamwork. In a construction project, by contrast, people tend to work in small groups or alone at assigned tasks. If the IS project does not exploit teamwork opportunities, there is a greater chance of failure.
- There are more opportunities for parallel work in IS projects. Projects in IS are not composed of rigid sequential tasks. While there are some dependencies, there are also many opportunities for parallel effort. If an IS project does not take advantage of this, a valuable opportunity is lost.
- Progress and milestones are sometimes more difficult to measure. Unlike physical construction, IS project milestones are more obscure. A systematic method for assessing IS milestones is needed.

This article applies these differences to define an alternative approach for managing IS projects.

2. Characteristics of IS Projects

Information systems projects are composed of many tasks. Tasks have risk because of organizational, technical, technological, business, and external factors. It is critical to be able to manage the risk. With people being shared across normal work and other projects there is a need to manage resources across multiple projects and work.

General management often has limited IS/IT knowledge. This makes communications between management and the IS team and project leader very important and risky. There are dangers of misunderstandings. Management can reset priorities on short notice. Based on the promise of technology, management has raised expectations for IS projects. With new methods and tools IS projects should be easier to do. The new technologies should also ease the workload in the IS project. Both of these statements are wrong, but are believed by many business managers.

Business units already have systems, procedures, and policies in place at the start of an IS project. The business staff believe in these business processes. When the IS project comes along, it is not surprising that the business managers and staff in a business unit are resentful and fearful of the IS project. They see their way of life threatened. More than that, they fundamentally do not perceive the need for change.

New technology brings the promise of greater productivity and easier maintenance of the systems. Yet, there are problems. First, the IS project team must not only understand the new technology, but become proficient in its deployment and use. There is often insufficient time in the project schedule for this. People must learn the technology as they go.

A second factor is that new technologies often create technology gaps between the existing technology and the new. With products from a variety of vendors complexity of the gaps is increased. Problems in system integration arise.

When an IS project is begun, the project faces several daunting challenges, including:

- How to adequately test the new system.
- How to convert the data from the current systems to the new system.
- How to gain end-user support for change.
- How to cope with management changes in direction, schedule pressure, and expectations.

3. Five Reasons Why IS Projects Fail or Overrun

There are many reasons why IS projects fail. Some of the more frequent sources of failure are:

- The IS project leader does not address issues and risk in the project. Issues continue unresolved while work in the project goes on. Eventually, a critical issue can destroy the project and bring it to its knees.
- The business department continues to resist the IS project. Business staff are pulled from the project to do their normal work. People who are not fully qualified are assigned to the IS project. Even when implemented, there is a lack of commitment. In some cases, the business department continues to use the old business process with the new system. There are then few, if any, benefits.

- The combination of management pressure, resource depletion due to the demands of normal work, and project and technology complexity act to subvert the schedule of the IS project.
- The IS project team members may not feel a commitment to the project. The project leader may not involve them in issue management or project management. Without commitment, they do not feel compelled to deliver 100%.
- The scope and purpose of the project were not thought through and agreed to at the start of the project. Different people have different views of the project. These different views are not reconciled in the project—paving the way for failure.

Many try to blame the technology, the business unit, or management for these problems. Yet, the root source of failure lies in the methods of the project and project management.

II. STEPS IN IS PROJECT MANAGEMENT

At the start of a project in IT there is an idea. Before the idea can be turned into a project, you must first develop some essential information about the project. This will be discussed in the project concept. It includes the purpose, scope, roles of departments in the project, issues regarding the project, overall schedule, and general benefits. The project concept solidifies the project idea sufficiently to enable a plan to be developed.

The next series of steps includes: the identification of the project leader, the development of a project plan, and identification and assembly of the project team. It is an essential goal of an IT project to prevent the recurrence of problems in the past and to capture and use lessons learned from past and current projects. Thus, a database of issues and another database of lessons learned are often useful for project management.

After you have the plan developed with the team, you can present it for review to management. Work now begins. However, unlike standard projects that have dedicated resources, resources are shared in IS projects. This means that a constant effort must be made to effectively allocate resources among projects and nonproject work. Tracking of work includes the tracking of resources. Another critical task is the review of milestones of the project.

III. PROJECT CONCEPT

A. What Is the Project Concept?

At the very beginning there is a project idea. Someone saw the need for the project and got people's attention. Instead of plunging in and developing the project plan, you should develop a project concept.

What is a project concept? The project concept is composed of the following:

- *Purpose of the IS project*—An IS project has both business and technical goals. You must define goals in both areas and relate these to each other. For example, suppose that you are going to implement a new point-of-sale system for a retail chain. This is the technical purpose. The business purposes can be some combination of increased sales, improved productivity, reduced errors, more accurate data, etc. Unless you identify the business goals, people will tend to have different expectations as to what the system will accomplish.
- *Scope of the project*—This is critical. The scope must be argued about and agreed to by everyone touched by the IS project. Otherwise, there is a greater chance for scope creep later. You should define the scope in terms of the following dimensions.
  - *Technology*—What technology will be employed in the project?
  - *Business processes*—Which business processes will be involved in the project? To what extent will these be changed?
  - *Organization*—To what extent will the business organization be changed through the IS project?
  - *Systems*—What systems will be impacted by the IS project?
  - *Policies*—How will policies of the business be altered by the IS project?
- *Risk and issues*—Based on past experience, what are the potential issues that the IS project face? Identifying issues before the project starts raises awareness of the business and management decisions that will be needed later.
- *General schedule*—What is the overall schedule and deadline of the IS project? What dates can the business unit live with?
- *Priorities*—How will the IS project get and retain resources for the project?
- *Costs and funding*—What monies are really needed for the project?

- *Roles in the IS project*—What is the role of the business unit in the project? What duties will contractors and consultants perform? Answering these questions and getting agreement can go a long way toward project success.

B. Why the Project Concept Is Important

The project concept is important for many reasons. First, through the discussions and review of the project concept, expectations of the project are made more realistic. Second, management review of the project concept means that management is more involved in the project at the start. Through understanding there will be greater support. Third, by participating in the definition and review of the project concept, business unit management will be more supportive and committed to the IS project.

C. Issues Database

An issue can be a problem or an opportunity. Some examples of issues are

- How to gain continued involvement of the project team.
- How to effectively communicate with management.
- How to carry out testing of the new system.
- How to effectively use new tools and technology.

In order to identify the initial potential issues in a project, it is useful to have a list of issues to start with. When the project is underway, the project leader will spend much of his time dealing with issues.

To reduce the effort in project management, you should establish an issues database for issues across all projects. The data elements of the issues database include:

- Title of the issue
- Type of the issue
- Description of the issue
- Project identifier
- Priority of the issue in the project
- Status of the issue in the project
- Date that the issue appeared in the project
- Impact of the issue if not solved
- Who the issue is assigned to
- How the issue was resolved
- Date of the resolution of the issue
- Impact of solving the issue

The type of the issue can be tied to the source. Potential sources of issues include: other projects, existing systems, technology, project team, contractors, business unit, management, and external factors. The impact of the issue if not resolved is critical because it helps set the priority for the issue.

D. Lessons Learned Database

The lessons learned database consists of data elements that capture experience for reuse and later applications. Data elements of the lessons learned database include:

- Name of the lesson learned
- Description
- Type of lesson learned
- Projects to which the lesson learned is applicable
- Date of the creation of the lesson learned
- How to use the lesson learned
- The expected benefits of using the lesson learned

Each time a lesson learned is employed, there is an obligation of the user to provide feedback in terms of experience. This may in turn lead to a refinement of the lesson learned and the creation of new lessons learned.

E. How to Use the Project Concept to Gain Consensus

The benefits of the project concept are clear. Furthermore, it does not take long to develop the project concept. In order to gain understanding and support, you should identify several alternative project concepts. One alternative is one concept in which the goals and scope of the project are very narrow. The system is implemented, but the underlying processes are not changed. At the other extreme, you have complete reengineering. The processes and organization as well as policies are redefined. In the middle is the alternative of changing the process and system without changing the organization. It is this middle ground where you often want to gain consensus. By having the extreme alternatives which have, respectively, fewer benefits and much greater risk, you can support for the middle.

IV. DEVELOPING THE PROJECT PLAN

There are several ways to go about building a task list. One approach is to use a work breakdown structure (WBS). A WBS is a detailed list of tasks to address various types of projects. A more advanced approach is to use a project template. A project template goes beyond a WBS in that it includes not only high-level tasks, but also dependencies and general resources for these high-level tasks. The project template provides more structure than a WBS and allows you to do multiple project analysis since all projects share the same resource pool and have high-level tasks in common. This is critical in IT projects where resources are often shared across multiple projects and regular, nonproject work.

A. Building a Project Template

Many IS projects are started with a blank piece of paper or a computer screen. The effort to create the schedule is enormous and tedious. Moreover, you are more likely to leave out critical tasks because, after all, you cannot think of everything. To get around this problem, you should define project templates for classes of projects. A project template is a successor to a work breakdown structure. A project template consists of the following:

- High-level tasks that are the same for all projects in the class. For example, all intranet development projects would share the same high level tasks.
- Dependencies among these high level tasks.
- Assignment of general resources to the tasks in the template. Examples of resources might be programmer, network support, analyst, and business unit staff.

The project template is employed as you start to build the schedule for the project. Lessons learned can be attached to specific tasks in the project template. This makes the lessons learned more available and usable.

B. Benefits of a Project Template

There are many benefits of a project template approach.

- By creating the templates, the IS staff gets a common understanding of the work at a high level.
- There is an increased awareness of the lessons learned since they are present in the template.
- The cost and effort to build the project plan from the template are reduced.
- The effort to review and understand the project plan are reduced.
- Over time and with experience the templates can be made more detailed. Estimated durations may also be able to be included. This means that there is cumulative benefit for later projects.

C. Creating the Lessons Learned Database

You should gather lessons learned from current and recent projects. Make these as concrete as possible so that they can be of more use. The lessons learned should then be linked to the tasks in the project template.

D. Developing the IS Plan in a Collaborative Manner

When you begin to develop a plan from a template, there is a tendency for the project leader to start identifying tasks. This is not a good idea for several reasons. First, you want to get the participation of the team members. Second, the project leader does not know all of the technical and business areas of the project. The project leader is really unsuited to define the detailed tasks within the plan.

The first step is to assign areas of the template to the team members. The team members go away and define the detailed tasks over the next three months. The tasks in the template that will be done more than three months in the future should remain. Later, during the project you will have the team define detailed tasks in the future on a rolling three-month basis. Each team member will review the tasks with the project leader. The project leader will now better understand what is involved in the work and where the uncertainty and risk will be.

Following this review, the team member will define dependencies and identify specific resources that are necessary to do the work. After review, the next step is to define the dates and durations for the work.

This approach has been shown to get results. First, by the sequential approach you better understand the nature of the work. Second, the team members are more committed to the project since they are more involved. Third, you identify potential issues and areas of risk in the schedule. Figure 1 gives a sample GANTT chart of some of the planning tasks.

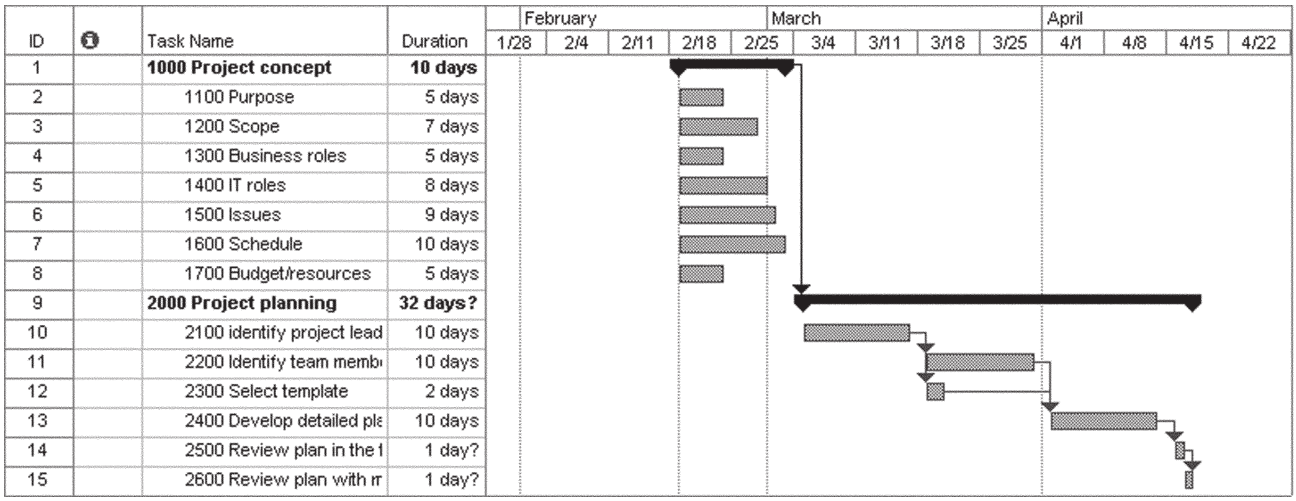


Figure 1 Sample GANTT chart of some planning tasks. Note the highest level tasks show dependencies at the same level of the outline in the plan.

E. Identifying Risk in the Project and Associating It with Issues

You have probably been taught to treat risk in some abstract mathematical way. This is not very productive in IS projects. Here is what you should do. First, take the list of issues in the issues database and find tasks in the plan that have these issues. You will find that there are some valid issues for which there is no task. This is good because you have now found some missing tasks.

Second, work the reverse way. Look at the tasks and see which ones are risky. Each risky task has one or more issues behind it. Find the issues that correspond to this risky task. You will find tasks that have issues but the issues are not in the issues database. Add the issues. This approach helps you to validate the schedule.

When estimating durations in the project plan, you will encounter either detailed tasks or template tasks in the future that you cannot estimate. Identify the issue behind this problem. Then chop up the task into smaller parts. You will eventually isolate the detailed tasks that you cannot estimate easily. Identify the issues here.

This approach has several benefits. First, you define issues in the project more precisely. Second, you associate these with tasks in the schedule. Third, you raise the level of awareness of the issue to the project team.

V. ORGANIZING THE PROJECT TEAM

Information systems projects suffer from several misconceptions related to the project team. The first myth is that you need to get the team members into the project early. This is a really flawed idea. Early in the project, you really do not know what skills you need. Getting people too early means that their productivity is reduced.

A second myth is that you should get the best people for the IS project team. This is a bad idea for several reasons. First, the best people are in demand everywhere. In the business unit if you get a senior business person on the project, you can cripple the business unit. Second, if you are given the best people, management expectations are raised. Third, even if you are given the best people, it is likely that they will be later taken away for other work.

A. Focus on Part-Time Team Members

You realize that you cannot get people full time on the project in most cases. Therefore, you will only get people part- or full-time for a limited period. Accept this and use it in negotiating for resources. Realize that as the project continues, you will want to probably change project team members as the requirements of the project change.

B. Dynamic Team Commitment

Never try to get people committed for more work than several months. The uncertainty in most IS projects means that you must have flexibility in staffing. People tend to be more enthusiastic in the short term and then lose the energy later. Capitalize on this in project staffing.

C. Importance of Sharing Information

From the start you want people to share information. One way of doing this is to have the team participate in scheduling. Not only do they define their own tasks, but they also update their tasks.

Up to 35% of the tasks in the project plan should be jointly assigned. This supports sharing information among the team and providing the project with a backup. It also provides different perspectives. In addition, it helps team members improve their skills.

Projects employ project management software. The project files should be available on a server so that everyone has read-only access to all schedules. Next, the project team members can access their schedules and update these on-line. They have write-access to their schedules. The issues and lessons learned databases are also available on-line for everyone. This information visibility ensures that there is an openness and awareness of what is going on in other projects.

D. Role of the Team Members in Project Management

To summarize, team members should be able to define their own work and schedules subject to management review and revision. Team members should be able to identify issues in their work. They should be able to take advantage of lessons learned. During the project they should update their own work. They should also become more aware of the business processes that they are addressing. There is no room for an ivory tower or academic approach.

Let's turn to the business units. Rather than relying on only a few people from the business unit, you should involve as many people as possible. Why do this? A few people do not have knowledge of all of the business rules and the transactions. Another reason is that you want to build grass roots support for the change and for the new business process and system. If people are shut out of the project, they are not aware of the new process or system until it is sprung on them at the end of implementation.

VI. OBTAINING PROJECT APPROVAL

A. Structure of a Management Presentation for an IS Project

In the past, management presentations were based on the benefits of the project. The project leader often cites benefits such as improved productivity, reduced paper, fewer errors, greater flexibility, and lower costs. Yet, management has heard this so often that emphasizing these will not work. Instead, in addition to benefits, you should stress the following:

- What happens if the project is not done? How will the business be impacted? What are effects on the organization?
- What happens if the project is deferred? What is the effect of the deferment on the business?

These are negative, but very powerful motivating factors. And they are certainly more credible than raw benefits.

B. Linking the IS Project to the Business Process

The IS project should be linked to the business processes that are affected. The technology and technical methods and tools are secondary—a distant second. Management probably does not understand the technology anyhow and stressing it will just estrange management from the project.

By linking the project to the business processes, you shift the benefits to the business. After all there are no benefits from IS and IT per se. The effects only come true when the business units make adjustments due to the new system.

C. Outline of the Management Presentation

The presentation to management begins with an overview of the current business processes and the problems that they have. This motivates people for change. The presentation ends with the specific actions to be taken after the presentation.

An effective outline of the management presentation of the project is as follows:

- Summarize the current business process. Highlight the problems in the current process. Give an example of a simple transaction in the process.
- Summarize the new business process in terms of how it would work. This part does not include the technology, but how the work would be different. Use the same transaction that you employed above.
- Identify the benefits of the new process over the current process. Use the example transaction to

reinforce this. At this point management understands what will happen as a result of the project.
- Define what will happen if the project is not done or deferred. Discuss the impacts in terms of the example transaction.
- Generally discuss the approach to the project. Here summarize the roles, scope, and purpose of the project from the project concept. Discuss the schedule and costs also.
- Highlight potential issues that the project faces. This shows management that the project will not be a "walk in the park."
- Identify specific actions that are needed in the short term. Make sure that the initial actions do not involve spending much money. The more money you ask for, the less likely you are to get management approval right away. The initial tasks can be in the form of identifying the new process in detail and getting requirements.

D. Followup after the Presentation

The most immediate followup is to get all of the team working. It is at the start of the project where you set the pattern of behavior for the rest of the project. It is also here where you reinforce the involvement and commitment of the team.

VII. GETTING AN IS PROJECT STARTED

Begin with establishing a project binder or file. You should have each project use the same tabs for the binder. After the project is completed, place the project binder in a library so that future project leaders and team members can use the information for their projects. This is another example of shared information.

The tabs of the binder include the following:

- Project concept
- Notes from project meeting
- Project GANTT charts—keep a record of these beginning with the baseline plan and all changes
- Issues in the project
- Relevant lessons learned
- Management communications
- Benefits of the project
- Requirements of the project
- Communications with the business units
- Communications with contractors and consultants

A. Importance of the Initial Project Tasks

The initial project tasks set the stage for the behavior of the team and how they work together for the entire project. If you push for individual work, you will inhibit later efforts to instill team work. The initial project tasks should be reviewed carefully by the project leader. This will show the team what to expect and the standards of their work.

B. Team Task Assignments and Work

Earlier we indicated that a substantial amount of the work should be assigned to more than one person. This is especially true at the start of the project. By having people work together they will identify a common approach for using methods and tools in the project. This is also an effective way to have senior members of the team impart knowledge to junior team members.

How should you handle project meetings? Gathering status in such meetings is almost meaningless. It is also a waste of time. Each person discusses what he or she did. Everyone else goes to sleep. Team members are reluctant to discuss issues or problems in front of the team. Gather status separately and individually prior to the meeting. Devote the meetings to either dealing with issues or sharing lessons learned. This makes the meetings more productive and interesting.

The second guideline is to vary the frequency and timing of meetings based on the urgency of the issues. If there is no burning issue, do not hold a meeting. If there are many issues, hold meetings more frequently.

There are a number of benefits to this meeting approach. First, team members learn more through the issues and lessons learned discussions. Second, you are focusing the project on resolution of issues or sharing knowledge—both very positive. A third benefit is that the meetings reflect the true status of the project. You are not meeting, for example, weekly when there are many issues to solve.

C. Evaluating Initial Work Products

You should have each person present and summarize their work to the team. By presenting their work they gain confidence in their methods. They share information with the team. Based on tone of voice as well as content, you can see where further review is needed. Most importantly, this approach gets the people to work together.

D. Communications with Management

Management communications are often thought of as formal methods where you go in and make a presentation to management on status. While this occurs, the most important management communications are informal. You want to meet one-on-one with individual managers and keep them informed on issues. This approach means that there is no risk to the managers. Also, the approach alerts managers to potential issues prior to the surfacing of the issues.

VIII. MANAGING THE IS PROJECT

A project leader has no real money and no resources. These belong to the organizations. The project leader has no permanent structure. With these limitations in mind, you have to wonder what assets a project leader has. There is the problem-solving ability of the project leader. However, there are two other assets. The first is information. The project leader has the most complete picture and knowledge of the project and everything related to it. Knowledge is power. The second asset is timing. With the knowledge that the project leader has, there is an opportunity to coordinate timing of issues and other items in the project. If you are going to be an effective IT manager, you must take advantage of these assets.

A. Tracking the Status of a Single IS Project

How do you track the status of a single IS project? Traditional methods have focused on percentage of the project complete and budget versus actual. However, these are trailing indicators. The project could be in real trouble when these numbers are bad. Moreover, percentage complete can be meaningless in an IS project.

What are better measures of status? The first question to answer is: What is the status of the oldest outstanding major issue? Answering this question will give you an idea of the degree of risk in the project today. The second question to answer is: Of the tasks that remain to be done, what percent have significant issues and, hence, high risk? Answering this question will tell you about the future risk in the project. After all, a project can be 70% complete in terms of hours, but only 40% complete in terms of risky tasks since 60% of the tasks that have risk lie in the future. Such is the case often in IS projects where much of the project risk is toward the latter stages when integration, conversion, testing, and user acceptance occur.

How do you get the status of the project? We already said that meetings were a bad way to gather status. An alternative and better approach is to go to each team member after they update their tasks and ask them how their work is going. This will lead to a discussion of issues that in turn will lead to the schedule and status. This approach gives you a much better idea of what is going on in the project.

B. Management Reporting of Projects

Let's begin with reporting on one project. Here is a one-page summary outline that has proven to be useful in many projects.

- Project title
- Purposes of the project—both business and technical
- Scope of the project—both business and technical
- Summary GANTT chart
- Cumulative budget versus actual expenditures
- Major milestones achieved
- Major upcoming milestones
- Status and discussion of unresolved issues

The listing of the purpose and scope in both business and technical terms serves to reinforce to management what the project is supposed to accomplish. The summary GANTT chart shows the overall progress and status of the project. Both actual and planned dates are shown. The discussion of unresolved issues is, perhaps, the most important part of the report since it focuses attention on issues and reinforces issue awareness.

For multiple projects there are some very useful charts and tables to present. These include:

- *Summary GANTT chart across all projects.* This gives a summary view of all active projects.
- *Summary GANTT chart that includes the tasks that have major issues.* This chart draws management attention to the issues in the projects.
- *Table of issues versus projects.* Issues are given as rows and projects as columns. The table entry is the impact of the individual issue on the specific project.
- *GANTT chart of all tasks across all projects that pertain to a specific issue.* This chart allows management to address a specific issue along with the preceding table.
- *Table of processes versus projects.* The rows are the business processes and the columns are the projects. The table entry is how the project supports the specific process. Take care here. If

there is a row with no entries, then the process in this row has no project. Either the process is perfect and there is no need for a project, or there is an opportunity for a new project. If there is a column with no entries, then there is a project which contributes nothing to the processes. Maybe it should be dropped.
- *Table of projects versus business units.* The rows are projects and the columns are the business units. The entry is the role of the business unit in the project.

Have these charts ready at all times. You will then be prepared for a management meeting on short notice.

C. Tracking Multiple Projects

How do you track multiple projects? You have the project reporting for the individual projects. Since the projects are all following templates, they have the same customization of the project management software and the same resource pool. This allows you to combine the projects using the project management software. You can do what-if analysis. You can filter or reduce the visible tasks to those that pertain to:

- The same issue
- The same resource
- The interrelated tasks and milestones of the project

D. Managing Risk across Projects

Risk is a fuzzy term. Here risk and issues are interrelated. You will want to use the table of issues and projects as well as the GANTT charts pertaining to issues presented above. Once you present these, you can have the project managers discuss what to do about the specific issue.

This approach also allows you to aggregate or group issues. You can then combine the related issues and try to address them as a group. When you come up with a solution, you can test it out on the schedules by doing what-if analysis with the project management software.

IX. MANAGING MULTIPLE PROJECTS

A. Key Goal Is Resource Allocation

Earlier we indicated that a major challenge in a project is to get, retain, and manage the resources of the project. We also indicated that these resources are spread across other work. Thus, one of the most important goals in managing multiple projects is to allocate the resources across the normal work and projects.

Let's consider the situation where there is no active management across multiple projects. Each project and normal work are managed separately. Now examine the lot of an individual programmer. This person has their line manager as well as project leaders coming to them and telling them what to do. Each manager tells the programmer that their work is the most important. What is the programmer to do? Who does the programmer listen to? Based on our own experience as programmers, we find that the programmer calls the shots. The conflicting signals given by various managers provide the programmer to play one manager off against another. The work is then managed bottom up.

B. Dynamic Resource Allocation among Project Leaders

How do we allocate resources? Here is a method that works. Each week line managers and project leaders get together and review what critical resources have been doing during the past week. They set priorities for the people for the next week. A manager is assigned to detail the priorities of the work to the staff. This makes sure that the message given to the person is consistent.

There are several benefits to this approach. First, the staff are managed top down. There is no bottom-up behavior that was mentioned earlier. A second benefit is that the meeting allows the managers the opportunity to perform analysis and trade-offs.

What do you need for these meetings? Every manager should have their schedules handy. It is also useful to have project management software and a projector so that the managers can do what-if analysis for the person' time.

C. Identifying and Resolving Issues across Projects

You have tables and GANTT charts that highlight specific issues. With this information at hand, how do you resolve an issue? Consider some alternatives for the issue, including:

- Apply more resources to the project. This in general in an IS project is a stupid solution, but useful to consider. It is stupid because when you bring more people into a project, the project work slows down while the new people are acclimatized into the project.
- Pull resources from the project. This is a sometimes useful approach.

- Put more money into the project. Money can buy technical, internal resources, or external resources. These resources may create more problems.
- Restructure the project to address the issue.
- Change the scope or purpose of the project.
- Do nothing.

Most of the time you should in fact do nothing. Often, you only know symptoms of an underlying problem. It is too early then to address the issue. The issue may not have reached the stage where management attention is needed. You then may not be able to get management support to address the issue. There also may not be urgency to address the issue.

D. Ending a Project

You might think that a project ends when the new system is installed and working. However, if you stop here, you do not ensure that you will have benefits. After all, benefits only come when the business processes are adjusted to take advantage of the new system. This can only come after the new system is installed.

When should a project end? The project should end after the business processes have been changed and after the results have been measured. This means that the scope of the project has to include these tasks. Measurement of the project at the end must work to validate the benefits that were identified earlier. This often can only be accomplished through reviewing individual transactions in the new business processes.

Using this approach you can answer the question: What is project success? Success in a project is not when the system is installed. The system could be a success but if the users do not use the system properly and change their own methods, the overall project can be a failure.

X. CONCLUSIONS

A. Critical Success Factors in IS Projects

Experience has shown the following to be critical success factors in IS projects.

- End-user involvement in the IS project must be continuous and widespread throughout the life of the project.
- Success of an IS project occurs only when the new business processes are in place and delivering benefits. The new systems support the business processes. Their completion does not signal success.
- The goals, scope, roles, and issues must be understood by all players at the start of the IS project. This avoids many problems later.
- When managing an IS project, you must manage issues. Issues are what can undermine a project.
- IS project management must be a collaborative effort. Through involvement in the project, the team members commit to the project and work. They are more enthusiastic and will give your project a higher priority.
- The same resources are shared across normal work and IS projects. Thus, it is important to manage multiple projects. It is also important to proactively allocate resources on a regular and continuing basis across projects.
- Management communications must be both informal and formal.
- Standard project templates and lessons learned provide for standardization, flexibility, and release the project manager to address issues.

B. Use of Standardized Project Templates

The project templates have been shown to have specific and tangible benefits. Templates provide for both standardization and flexibility. At the highest level of the project there is standardization to support analysis and management control. Below the template the tasks can be anything that the team members and project leader want them to be. The result is standardization on top of flexibility.

C. Manage Multiple Projects

People work in multiple projects and work. You should consider having normal work recorded in summary form in the project management software. This allows you to do multiple project analysis. Managing multiple projects involves several important actions.

- Allocating staff time across the various projects and their normal work on a regular basis.
- Managing and addressing issues across multiple projects.

D. Gather Lessons Learned

At the heart of the method that has been outlined here has been the need to generate cumulative improvements in IS projects. We just have to get better with future projects. The use of templates provides for one means of using experience and knowledge. The templates can be

improved over time. Addressing issues and documenting them means that you are building to your base of lessons learned. With the lessons learned attached to the template, you can improve future projects.

SEE ALSO THE FOLLOWING ARTICLES

Benchmarking • Enterprise Resource Planning • Outsourcing • Staffing the Information Systems Department • Strategic Planning

BIBLIOGRAPHY

Hallows, J. E. (1997). *Information systems project management.* Amacom.

Kerzner, H. (2000). *Project management.* New York: J. Wiley and Sons.

Lientz, B. P., and Rea, K. P. (2000). *Breakthrough technology project management,* Second Edition. San Diego: Academic Press.

Lientz, B. P., and Rea, K. P. (2001). *Project management for the 21st century,* Third Edition. San Diego: Academic Press.

Prototyping

Merle P. Martin
California State University, Sacramento

GLOSSARY

adaptive prototype (level 3) A working model of the final system.
bottom-up prototyping An approach where prototyping is used, not as an overall methodology, but as an optional tool within each SDLC stage.
bread boarding Building products by patching together existing product components.
heuristic prototype (level 2) An enhancement of the input/output (level 1) prototype to include important system processing functions, particularly those related to updating of a database.
input/output prototype (level 1) Generation of the most frequently used reports and screens with emphasis on screen flow sequence (navigation path) and screen option/entry sequence.
internal controls Special procedures and programs designed to protect an organization from human error and fraud related to resource use.
iterative prototype A prototype model that becomes the final system after a series of evolutionary changes based upon user feedback.
menu map A diagram that organizes the system into the separate interactive screens and the flow between these screens that the user will encounter.
navigation path The flow between screens in an interactive system.
object-oriented prototype A patched-up prototype consisting of preconstructed objects.
patched-up prototype A prototype model patched together from existing programming code.
program stub An incomplete program module that, when called, passes control back to the calling module.
radical top-down design A top-down design methodology emphasizing timely delivery of a functional system, even if that system is not complete.
rapid application development (RAD) A methodology combining joint application development (JAD), automated design tools, and prototyping to quickly develop application systems.
throwaway prototype A prototype that becomes a model for the final system and that is abandoned for coding (construction) purposes.

PROTOTYPING is the process of quickly building a model of the final software system in order to elicit system requirements from and actively involve ultimate system users.

I. WHY PROTOTYPING?

There are many reasons why prototyping may be used in a systems development project.

A. Understanding System Unknowns and Risks

When you're working with new system ideas with your users, you don't want to go through the cost of

> developing a gigantic system (which might take years); you'll build a mock-up of it, which might take weeks.
>
> *Brian Kilcourse, Chief Information Officer*
> *Long's Drug Stores*

A prototype model provides the analyst with a vehicle to understand the problem environment. It is useful particularly for "noncookie cutter" applications such as expert systems, decision support systems, and executive information systems. These systems, unlike fairly standard transaction processing systems, are unique and often new to the organization—they explore new territory. Prototype models allow the analyst to "surround" system unknowns and to better gauge system risks and costs.

The State of California's Department of Parks and Recreation created a logical design for automated kiosks at the entrances to their most frequently visited state parks. They developed a prototype model of the user interface and sent it to interested vendors to elicit ideas as to the practicality, functionality, and prospective costs for such a system.

B. Analyst/User Communication

In the typical nonprototyping environment, users are presented with an arcane collection of flowcharts, entity-relationship designs (ERDs), data flow diagrams (DFDs), and network diagrams. All of these are difficult for the nontechnical user to understand. It is not surprising then that the final system often is a surprise (or worse) to users.

Prototype models are system requirements and possibilities expressed in the users' everyday world. A prototype includes the keyboarding, navigation path, and report extraction that relates to what the user actually does or will do. In addition, a prototype model can demonstrate to users what (1) is actually feasible with existing technology, and (2) weaknesses exist with this technology.

C. User Involvement

One of the most significant predictors of ultimate application system success is user ownership of that system. Prototyping allows users to become involved early and consistently throughout the development project. Users suggest changes and see those changes made almost immediately. Prototyping can stretch users' imaginations and can elicit more creative input, thus resulting in a more forward looking system. This type of user involvement breeds a sense of user ownership—that this is *their* system rather than one thrust down upon them from above.

D. Earlier Error Detection and Correction

Experiences at AT&T, TRW, and IBM have shown that some 30% of system requirements will change *before* initial system delivery. The longer it takes to discover these changes, the more expensive they will be to correct. This is shown in Fig. 1, which has been derived from assembly line situations such as in the auto industry. This time-to-cost relationship in application systems development is partly due to the fact that there is more module interlinking as the project proceeds.

Prototyping allows errors and requirements changes to be detected earlier in the systems development life cycle (SDLC) (asterisks in Fig. 1). Experiments at the University of Southern California have shown that prototyping can decrease program size and effort by 40%.

E. Earlier and More Extensive System Training

Prototyping allows user training to begin earlier in the development project. Prolonged exposure to change has been shown to decrease significantly users' resistance to that change. Early project training using prototype models can, therefore, facilitate the implementation process.

F. Hand Off from Analyst to Programmer

A prototype is a valid means of documentation that can reduce miscommunication during the transition from logical design to physical design and imple-

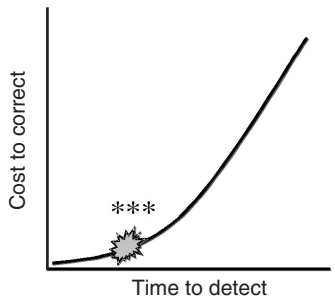


Figure 1 Error correction model.

mentation. The prototype is excellent for demonstrating analyst intent as to the "look and feel" of the ultimate system. In addition, the prototype built during the analysis phase can serve as a "test bed" during the implementation phase—as a vehicle to continually validate user requirements.

II. PROTOTYPING ENVIRONMENTS

A. Prototyping Approaches

Prototyping can be used as a tool within the SDLC or a methodology within which the SDLC proceeds. Three different approaches to using prototyping are rapid application development (RAD), radical top-down, and bottom-up prototyping.

Often RAD is confused with prototyping itself. Yet prototyping is only part of the RAD methodology. Burch (1992) defines RAD as: "a methodology that combines joint application development, specialists with advanced tools (SWAT) teams, computer-aided system and software engineering tools, and *prototyping* to develop major parts of a system quickly."

The *radical top-down design* approach emphasizes timely delivery of a functional system even if that system is not yet completed. It counters traditional approaches where the system is not delivered until complete, but it is far too often delivered later than promised. Radical top-down design proceeds as follows:

- A complete systems hierarchical chart is developed.
- Lower level (primitive) modules are prioritized based upon user needs and criticality to the total system.
- Control (upper-level) modules are developed and tested first to establish the system's navigation path.
- Lower level modules are "stubbed out." A program stub is an incomplete program module that, when called, immediately passes control back to the calling module, displaying a message such as "Under Construction."
- One by one, the lower level modules are developed and "unstubbed" according to module priority.
- This continues until the promised delivery date arrives, when the incomplete but functional application system is delivered to the client.
- Work continues on lower priority modules until the system is complete.

This approach lends itself quite well to use of prototyping since, as discussed later, prototyping is a top-down process.

With *Bottom-up prototyping* some system developers are skeptical of a top-down approach because it may delay the more intricate and complex programming (primitive modules) until too late in the SDLC. Some organizations prefer to use prototyping, not as an overall methodology, but as an optional tool that can be used within each SDLC stage. In this approach, prototypes tend to be of a narrower scope but deeper in detail.

B. Prototype Types

There are three different types of prototypes that can be developed:

1. *Patched-up:* This type of prototype is patched together from existing code. The process of patching in engineering is called *bread boarding*. Generally, patched-up models are developed when both the prototyping and operational languages are a 3GL such as COBOL, where there is an abundance of reusable code. The current extension of patched-up prototypes is constructed of objects, or *object-oriented prototypes*.
2. *Iterative:* This prototype becomes the final system after a series of evolutionary changes based upon user feedback. Often, this type of prototype is referred to as a "first-of-a-series."
3. *Throwaway:* This type of prototype becomes a model of the final system, and is abandoned for coding (construction) purposes. It resembles an architect's drawing of a building that must be converted to a technical blueprint before construction can begin. Some iteration occurs. The steps of analysis, prototype design, coding, testing, and modification may be repeated several times until all user requirements are identified and met. Then the prototype is "thrown away" for system design purposes. It can be used, however, for training and demonstration. Often this type of prototype is referred to as "nonoperational."

1. Selecting the Type

Figure 2 provides guidance as to which types of prototype to construct. If the ultimate operational language will be a 3GL, then the prototype should be a throwaway, if the prototyping language is a 4GL or patched-up (using existing code), if the prototyping

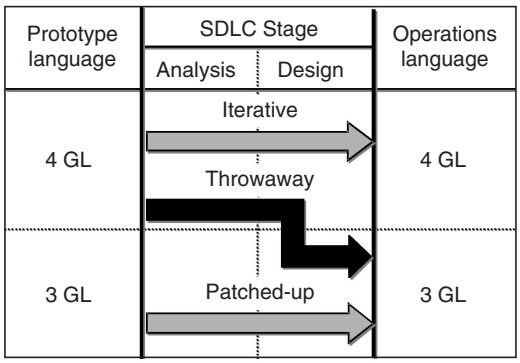


Figure 2 Prototyping type.

language will use the same 3GL as the operational language.

If both the operational and prototyping language will use the same 4GL, then the prototype will be iterative. It will undergo a series of transformations until it becomes the final system.

C. Prototype Stages

Prototyping development proceeds through the following stages, or levels:

- *Level 1 (input/output)*—This is generation of the most frequently used reports and entry screens; emphasis is on screen flow sequence (navigation path) and screen option/entry sequence. The primary purpose of the level 1 prototype is to establish a medium of communication between users and analysts.
- *Level 2 (heuristic)*—The level 1 prototype is enhanced by including important system functions, particularly those related to updating of a database. The most frequent transactions are included. Generally, some 20% of all transaction types account for some 80% of transaction volume. These 20% are the domain of the level 2 prototype. Edit and real (as opposed to simulated) database update capabilities are not included at this stage.
- *Level 3 (adaptive)*—This prototype is a working model of the ultimate system. Often the level-3 prototype is referred to as a "system with training wheels." The goal of this level prototype is to provide a means for fine-tuning the final, deliverable product.

This is a theoretical sequence that may be modified in practice. For example, a database-oriented 4GL will require you to first establish the database (level 2) before designing any screens (forms) or reports.

III. PROTOTYPE DEVELOPMENT

A. Prototype Life Cycle

Figure 3 shows how the prototyping life cycle differs from the traditional SDLC. The prototyping life cycle consists of five stages:

1. *Analyze the problem:* Determine the key aspects of the proposed system. This may require three separate prototyping efforts to determine:
   - The user interface
   - Uncertain or vague system functions
   - Time and memory requirements (in this context, the prototype operates as a type of simulation model)
2. *Design the prototype:* Build the prototype using the selected prototyping language.
3. *Test the prototype:* Have users exercise the prototype using test documents and transaction entry scripts.
4. *Refine the prototype:* Make any changes to correct erroneous functionality or to incorporate user suggested enhancements. This step leads back to stage 1 or 2.
5. *Use the prototype:* Operationalize the prototype as:
   - A model for design of the final system (throwaway prototype)
   - The deliverable system after refinement

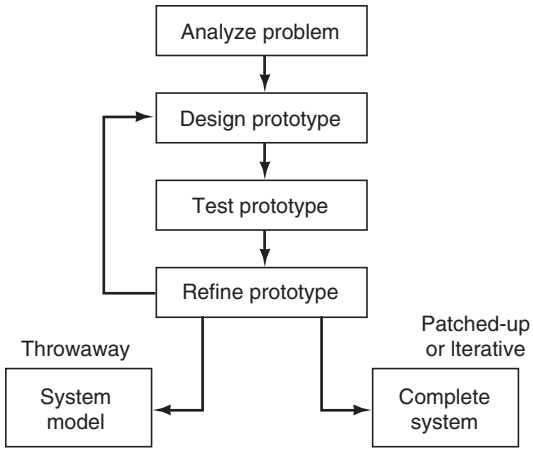


Figure 3 Prototype life cycle.

B. Development Sequence

Each level of prototyping follows the first four steps of Fig. 3. The problem is analyzed in part, then the level 1 prototype is designed, tested, and refined. After a sufficient number of iterations, the problem is analyzed further. Then the level 2 prototype is designed, tested, and refined. The level 3 prototype follows the same sequence with this exception. For a throwaway prototype, the process may be halted after level 2. The throwaway prototype may not be developed to the same depth of detail, as are the iterative or patched-up models.

C. Development Structure

The development structure of a prototype is shown in the Menu Map of Fig. 4. A *menu map* is the application users' view of what the system will be. This diagram, sometimes referred to as a navigation diagram, is similar to a program hierarchical chart. The difference is that the menu map organizes the system into the separate interactive screens and the flow between these screens that the user will encounter. The Program Hierarchical Chart, on the other hand, is organized by functional program module. Prototype development occurs in the following sequence:

- Lower level (primitive) modules are stubbed out.
- Control modules (navigation path) are the targets for level 1 prototype development. This allows users to explore the navigation path and screen characteristics in order to gauge the "look and feel" of the system.

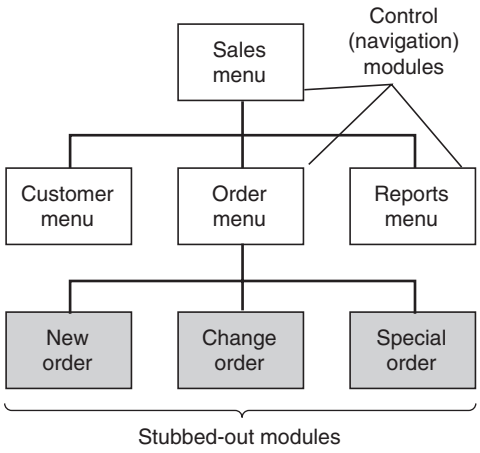


Figure 4 Menu map.

- Primitive modules for the most frequent system activities (e.g., Add New Customer) are unstubbed and made operational during development of the level 2 prototype. This then allows users to simulate most commonly experienced file updates.
- More extensive editing, file update scenarios and report/query functions are added to create the level 3 prototype to give users a "real-world" feel of what the final system will be.

D. Prototype Programming

Any language can be a prototyping language, particularly if breadboarding is used. Surprisingly, Martin and Carey found COBOL to be the most frequently used prototyping language in surveyed manufacturing firms. Some 4GL languages, however, lend themselves better to the ease and speed expected of prototype design. There are five progressive levels of programming that can be used to develop prototype models.

1. *Language platform:* One can develop this prototype directly in languages such as Visual Basic or C++ without any interactive design aids. Many of the manufacturing firms in the Martin and Carey survey used existing COBOL code to develop their prototype models.
2. *Data design language:* A database-oriented language will include a segment that allows tables, screens, reports, and queries to be generated quickly. In Microsoft's ACCESS application (written in Visual Basic), for example, this equates to the Design View where system elements can be constructed without "coaching" assistance.
3. *DDL with coaching:* Many database-oriented languages provide users with coaching assistance in development of system functions. Thus, ACCESS has Wizard functions, which will generate automatically screens (forms), tables, and reports.
4. *Application generators:* Some applications such as ACCESS have complete application systems built into the software package. For example, ACCESS users may select a fully operational purchasing system and easily tailor it to their unique organizational requirements.
5. *CASE tools:* Computer-assisted systems engineering (CASE) tools have capabilities for automatically generating prototype models from data dictionary (repository) specifications. The Visible Analyst

Workbench, for example, has an automatic generation feature called the Visible Prototyper.

E. Best Practices Recommendations

Price Waterhouse posits the following "best practices" recommendations for developing prototypes:

1. Use at least 30% of your development staff for prototyping.
2. Emphasize development of the user interface and application functionality.
3. Don't emphasize maintainable code or formal documentation.

This author adds one other recommendation—*keep it simple*. The prototype should not comprise the full range of system functionality. This would make the prototype too complex and would defeat the purposes of user communication and involvement. The prototype should consist only of the most frequently experienced transactions—that 20% that accounts for 80% of daily volumes.

IV. PROTOTYPES FOR USER TRAINING

A. Training Needs

Carey found that user resistance to change decreased significantly with prolonged exposure to that change. Early training that is continued throughout the SDLC not only creates a more knowledgeable user population, but it also reduces implementation anxieties. Prototypes developed in the early stages of the SDLC are effective vehicles for earlier user training. Even throwaway models, while discarded for design purposes, can become excellent training vehicles.

B. Prototype Training Sequence (Fig. 5)

1. A level 1 prototype is developed during the requirements analysis stage. After this model is used to elicit user needs, it can be focused upon user familiarization. In addition, the level 1 prototype can be used as a demonstration tool to "sell" management on the proposed system.
2. The level 2 prototype is developed during the conceptual design stage of the SDLC. This prototype can be used for user indoctrination, a process that occurs much later in traditional projects not utilizing prototyping.

Prototype Level	SDLC Stage	Type User Training
Level-1 Input/Output	Requirements Analysis	Familiarize
Level-2 Heuristic	Conceptual Design	Indoctrinate
Level-3 Adaptive	Detailed Design	Train in-depth

Figure 5 Prototype training sequence.

3. The level 3 prototype is developed during the detail design stage of the SDLC. This prototype is a workable system, but without infrequently used features. It can be used as a vehicle for intensive, in-depth user training.

C. Prototype Training Problems

There are two problems with using prototype models for user training. The first problem is that, if the prototype model is markedly different from the final system, then prototype training will be counterproductive. The second problem is that early training using prototyping will prove to be ineffective if it is not continued throughout the SDLC. Too much will be forgotten if there is an appreciable gap between the time of prototype training and system implementation. The advantage of prototype training is not only on the earliness of such training, but that the training can be enhanced and sustained throughout the SDLC.

V. PROBLEMS WITH USING PROTOTYPES

Prototyping is not a panacea. It is merely a tool that, if used inappropriately, will have the same effect as using a saw to hammer a nail. There are many potential problems with use of prototypes. Three of the most serious problems are (1) giving users false expectations, (2) bypassing problem identification, and (3) ignoring operational problems.

A. User Expectations

Initial prototype versions may lead user expectations astray in the areas of development time, functionality, complexity, and the operational environment.

- *Development time*—The ability to build a prototype rapidly may raise user expectations about delivery time for the final system. "What do you mean it will take nine months to complete the system? It took you only a week to build this prototype, and it works very well."
- *Functionality*—The final system may look different than the prototype, particularly if the prototype is a throwaway model. The prototype built in a 4GL may have different functionalities (e.g., color, animation) than the final system programmed in a 3GL.
- *Complexity*—The prototype is based on the 20% most frequent transaction types that account for 80% of total transaction volume. A system of this limited scope and complexity is built quickly and experiences few errors. Adding the other 80% of system functionality and the module interlinking required increases complexity exponentially. Not only does development time increase, but the number of system errors and rework required increases also—often beyond what users expected given their prototyping experiences.
- *Operational environment*—Generally, prototype models are developed on a computer isolated from the main business. This "laboratory" allows users to test drive the system without being affected by or interfering with the chaos of the work setting. Minor system inconveniences may later become major problems when phones are ringing, customers are scowling, and a supervisor is exerting production pressure.

B. Problem Identification

The process of prototyping is easy, quick, and often enjoyable. Analysts are tempted to bypass the often unexciting work of problem identification and proceed immediately to prototype development. The result may be a system that "looks good," but does not solve user problems. As Russell Ackoff has stated:

> It has been observed that we more frequently fail to face the right problem than fail to solve the problem we face.

Prototyping is a valuable enhancement to, but not a suitable replacement for problem identification.

C. Operational Requirements

Often a system designed to optimize day-to-day operational efficiency requires some sacrifice of design speed and cost. Cheap and rapid systems design often leads to ineffective operating systems. Use of prototyping too often entices the analyst to consider the prototype as an end to itself, rather than as a means to an end. The state of New Jersey contracted with a consulting company to redesign its drivers license and vehicle registration systems. The consulting firm planned to use a 4GL throwaway prototype and code the final system in COBOL. However, project delays and cost overruns induced the consultants to make the prototype iterative with the final system programmed in the 4GL. The delivered system was too slow; half the motorists in New Jersey ended up with an invalid drivers license, an invalid vehicle registration, or both. Another consulting firm fixed the problem by converting the high volume transaction modules from the 4GL to COBOL. This is an example of analysts focusing on development speed and cost rather than ultimate operational system effectiveness. Often it is easier to select an iterative prototype rather than a throwaway version where efforts must be redirected to another programming language. Yet, if the other programming language (e.g., COBOL) will be more effective for day-to-day processing, then the throwaway model must prevail. Operational requirements must dictate prototyping use, rather than the other way around.

VI. TRANSITION BETWEEN PROTOTYPE AND OPERATING ENVIRONMENTS

Often there are pronounced differences between characteristics of the prototyping environment and those of the day-to-day environment in which the final system will operate. Developers using prototyping must remind themselves continually of these differences—that the prototype is not the real world but only a simulation of that world. Martin and Carey (1991) found that many manufacturing firms preferred to build patched-up prototypes rather than face the challenges of reconciling differences between a 4GL prototyping and 3GL operating environment.

A. Differences

There are six primary differences between prototyping and operational system environments:

1. *Language:* In a throwaway prototype, the prototype language will differ from the language in which the final system is coded. Often the

prototyping language will be a 4GL and the operational system language a 3GL. The 4GL will have functions not always available in a 3GL, such as date handling or graphics user interface (GUI) interfaces. In addition, the 4GL may not have the same self-documenting features, as does a 3GL such as COBOL.

2. *Range of transactions:* The prototype models those 20% most frequently occurring transactions that account for 80% of total transaction volume (Fig. 6). This 20% domain shows users what typically happens, rather than what always will happen. This typical domain must be expanded to the full range of possibilities to meet operational requirements and contingencies.
3. *Documentation requirements:* Both the prototype and the operational system undergo change. However, prototype changes occur in a relatively short time span and typically are made by the original prototype designers. Therefore, system documentation is not as important for a prototype, and often is omitted or minimized. An effective operational system, on the other hand, cannot tolerate shoddy documentation since a different programmer often makes changes at a much later time.
4. *Access control:* The prototype is isolated in a near laboratory setting and uses simulated rather than real data to update files. Thus, use of access controls such as passwords is not a critical prototyping requirement. Access control must be added in the transition between the prototype and the more complex, fraud-prone operational environment.
5. *Procedures:* Written procedures are needed to sequence and integrate all system components, including human operators. Development of such procedures is time consuming and is rarely done for the relatively simple and isolated prototyping environment. However, these procedures must be developed before final system delivery.
6. *Internal controls:* Internal controls are special procedures and programs designed to protect an organization from human error or fraud related to resource use. Auditing of application systems to detect inappropriate resource use is a field of itself. Suffice it to say here that a significant number of internal controls must be added to a prototype model to make it an acceptably auditable operational environment. Such controls may include programming of Imbedded Test Modules, Integrated Test Facilities, and Audit Hooks.

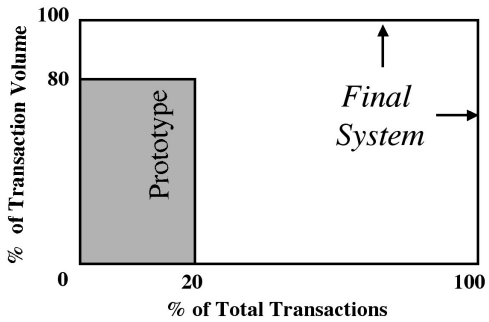


Figure 6 Prototyping realm.

B. Recommendations

Following are some recommendations for facilitating the transition between the prototyping and operational environments.

1. *Build a data dictionary:* A data dictionary should be constructed at the beginning of prototype development. As prototype file structures are initiated, entries should be required for both the prototype and operational model.
2. *Minimize frills:* 4GLs often have convenient features (e.g., automatic editing) that will not translate well to a 3GL, or even to other 4GLs. Such special features (often frills) falsely feed user expectations at the prototyping stage. Prominent among these stretched user expectations are overuse of color and animation. When building the prototype model, the analyst must imagine how features will appear (or maybe not appear) in the operational system.
3. *Include moderate documentation:* Require at least menu maps and program hierarchical charts so that conceptual system structure can be communicated more effectively between the prototype designer and the operational system programmer.
4. *Consult a maintenance programmer:* Include as part of the prototype design team a programmer who will have the responsibility for maintaining the operational system. This will ensure an operational system mindset during prototype construction.

VII. CONCLUSION

Prototyping is an approach or tool that can add significant value to the application system development process. As with any tool, however, it can be counterproductive if used inappropriately. This was shown in the New Jersey motor vehicle example. To avoid the pitfalls of inappropriate prototype use, the developer must remember that the prototype is a means to an end, not an end of itself. Always keep in mind how prototype features will function in the final operational system environment, and use the prototype primarily as a means to communicate with and to involve system users.

SEE ALSO THE FOLLOWING ARTICLES

Database Development Process • Documentation for Software and IS Development • End-User Computing Concepts • Quality Information Systems • Software Process Simulation • Structured Design Methodologies • System Development Life Cyle • Systems Implementation • User/System Interface Design

BIBLIOGRAPHY

Ackoff, R. (1978). *The art of problem solving.* New York: John Wiley & Sons.

Burch, J. (1992). *Systems analysis, design and implementation.* Boston, MA: Boyd & Fraser.

Carey, J. (1987). Understanding resistance to system change: An empirical study. *Human Factors in Management Information Systems* (J. Carey, ed.), pp. 195–206. Norwood, NJ: Ablex Publishing.

Carrol, J., and Carrithers, C. (1984). Training wheels in a user interface. *Communications of the ACM,* pp. 42–47.

Kull, D. (1986). Anatomy of a 4GL disaster. *Computer Decisions,* 18, 58–65.

Martin, M. (1987). The human connection in systems design. *Journal of Systems Management,* 38, 19–22.

Martin, M. (1995). *Analysis and design of business information systems.* Englewood Cliffs, NJ: Prentice Hall.

Martin, M., and Carey, J. (1991). Converting prototypes to operational systems: Evidence from preliminary surveys. *Information and Software Technology,* 33, 351–356.

Price Waterhouse World Technology Center (1997). *Technology forecast: 1997.* Menlo Park, CA: Price Waterhouse.

Romney, M., and Steinbart, P. (1999). *Accounting Information Systems.* Upper Saddle River, NJ: Prentice Hall.

Pseudocode

Lesley Anne Robertson
University of Western Sydney

GLOSSARY

algorithm A set of detailed, unambiguous and ordered instructions developed to describe the processes necessary to produce the desired output from given input.

CASE control structure A structure that extends the basic selection control structure from a choice between two values to a choice from multiple values.

flowchart A graphical representation of program logic, using a series of standard geometric symbols and lines.

pseudocode A subset of English that has been formalized and abbreviated to look like a high-level computer language. Keywords and indentation are used to signify particular control structures.

structure theorem The structure theorem states that it is possible to write any program using only three basic control structures:

1. Sequence: the straightforward execution of one processing step after another.
2. Selection: the presentation of a condition, and the choice between two actions, depending on whether the condition is true or false.
3. Repetition: the presentation of a set of instructions to be performed repeatedly, as long as a condition is true.

I. INTRODUCTION

Pseudocode is a set of English-like words and conventions that are used to represent an algorithm when designing a solution to a computer-programming problem. There are a number of common words and keywords used when writing pseudocode as well as conventions for their use. The Structure Theorem states that it is possible to write any computer program by using only three basic control structures: sequence, selection, and repetition. These control structures can be represented in pseudocode using keywords such as `IF, THEN, ELSE, DOWHILE` and `ENDDO,` and correct indentation. Many pseudocode examples for these control structures are provided in this article.

II. WHAT IS AN ALGORITHM?

An algorithm is like a recipe: it lists the steps involved in accomplishing a task. It can be defined in programming terms as a set of detailed, unambiguous, and ordered instructions developed to describe the processes necessary to produce the desired output from a given input. The algorithm is written in simple English and is not a formal document. However,

to be useful, there are some principles, which should be adhered to. An algorithm must:

- Be lucid, precise, and unambiguous.
- Give the correct solution in all cases.
- Eventually end.

For example, if you want to instruct someone to multiply a list of numbers on a pocket calculator, you might write an algorithm such as the following:

```
Turn on calculator
Clear calculator
Repeat the following instructions
  Key in first number
  Press multiplication (*) key
Until all numbers have been entered
Write down result
Turn off calculator
```

Note that in this algorithm the first two steps are performed once, before the repetitive process of entering the numbers. After all the numbers have been entered and multiplied, the result can be written down and the calculator turned off. These final two activities are also performed only once. This algorithm satisfies the desired list of properties: it lists all the steps in the correct order from top to bottom, in a definite and unambiguous fashion, until a correct solution is reached. Notice that the steps to be repeated (entering and multiplying the numbers) are indented, both to separate them from those steps performed only once, and to emphasize the repetitive nature of their action. It is important to use indentation when writing solution algorithms because it helps to differentiate between the different control structures.

III. WHAT IS PSEUDOCODE?

Traditionally, diagrams such as flowcharts were used to represent the steps in an algorithm. However, pseudocode is now a popular alternative to diagrams because it is easy to read and write and allows the programmer to concentrate on the logic of the problem. Pseudocode is really structured English. It is English that has been formalized and abbreviated to look like high-level computer languages.

There is no standard pseudocode at present. Authors seem to adopt their own special techniques and set of rules, which often resemble a particular programming language. This article establishes a standard pseudocode for use by all programmers, regardless of the programming language they choose. Like many versions of pseudocode this standard version has certain conventions, as follows:

- Statements are written in simple English.
- Each instruction is written on a separate line.
- Keywords and indentation are used to signify particular control structures.
- Each set of instructions is written from top to bottom with only one entry and one exit.
- Groups of statements may be formed into modules, and each group given a name.

IV. HOW TO WRITE PSEUDOCODE

When designing a solution algorithm, you need to keep in mind that a computer will eventually perform the set of instructions written. That is, if you use words and phrases in the pseudocode, which are in line with basic computer operations, the translation from the pseudocode algorithm to a specific programming language becomes quite simple.

There are six basic computer operations with corresponding words and keywords in pseudocode to represent these operations. Each operation can be represented as a straightforward English instruction, with keywords and indentation to signify a particular control structure.

A. Six Basic Computer Operations

1. A Computer Can Receive Information

When a computer is required to receive information or input from a particular source, whether it be a terminal, a disk, or any other device, the verbs `Read` and `Get` are used in pseudocode. `Read` is usually used when the algorithm is to receive input from a record on a file, while `Get` is used when the algorithm is to receive input from the keyboard. Note that a screen prompt instruction is usually required before a `Get` statement. For example, typical pseudocode instructions to receive information are

```
Read customer name
Get system date
Read number1, number2
Prompt for product_code
Get product_code
```

Each example uses a single verb, `Read` or `Get`, followed by one or more nouns to indicate what data is to be obtained. At no stage is it necessary to specify

the source of the data, as this information is not required until runtime.

2. A Computer Can Put Out Information

When a computer is required to supply information or output to a device, the verbs `Print`, `Write`, `Put`, `Output`, or `Display` are used in pseudocode. Print is usually used when the output is to be sent to the printer, while `Write` is used when the output is to be written to a file. If the output is to be written to the screen, the words `Put`, `Output` or `Display` are used in pseudocode with any literals enclosed in single quotes. Typical pseudocode examples are

```
Print "End of Processing"
Write product record to master file
Put out student_name, student_address
Output total_price
Display "End of data"
```

In each example, these data to be put out are described concisely using mostly lower case letters.

3. A Computer Can Perform Arithmetic

Most programs require the computer to perform some sort of mathematical calculation, or formula, and for these, a programmer may use either actual mathematical symbols or the words for those symbols. For instance, the same pseudocode instruction can be expressed as either of the following:

```
Add price to total_cost
total_cost = total_cost + price
```

Both expressions clearly instruct the computer to add one value to another, so either is acceptable in pseudocode. The equal symbol, =, has been used to indicate assignment of a value as a result of some processing.

To be consistent with high-level programming languages, the following symbols can be written in pseudocode:

+	Add
−	Subtract
*	Multiply
/	Divide
( )	Brackets

The verbs `Compute` and `Calculate` are also available. Some pseudocode examples to perform a calculation are:

```
Divide total_price by item_count
sales_tax = cost_price * 0.15
Compute C = (F - 32) * 5/9
```

When writing mathematical calculations for the computer, standard mathematical "order of operations" applies to pseudocode and most programming languages. The first operation carried out will be any calculation contained within parentheses. Next, any multiplication or division, as it occurs from left to right, will be performed. Then, any addition or subtraction, as it occurs from left to right, will be performed.

4. A Computer Can Assign a Value to a Variable or Memory Location

There are three cases where you may write pseudocode to assign a value to a variable or memory location.

1. To give data an initial value in pseudocode, the verbs `Initialize` or `Set` are used.
2. To assign a value as a result of some processing, the symbols = or ← are used. Note that the = symbol is used to assign a value to a variable as a result of some processing and is not equivalent to the mathematical = symbol. For this reason, some programmers prefer to use the ← symbol to represent the assign operation.
3. To keep a variable for later use, the verbs `Save` or `Store` are used.

Some typical pseudocode examples are:

```
Initialize total_price to zero
Set product_count to 0
total_price = cost_price + sales_tax
total_price ← cost_price + sales_tax
Store student_num in last_student_num
```

5. A Computer Can Compare Two Variables and Select One of Two Alternate Actions

An important computer operation available to the programmer is the ability to compare two variables and then, as a result of the comparison, select one of two alternate actions. To represent this operation in pseudocode, special keywords are used: `IF`, `THEN` and `ELSE`. The comparison of data is established in the `IF` clause, and the choice of alternatives is determined by the `THEN` or `ELSE` options. Only one of these alternatives will be performed. A typical pseudocode example to illustrate this operation is

```
IF student_attendance_status is
    part_time THEN
  add 1 to part_time_count
ELSE
  add 1 to full_time_count
ENDIF
```

In this example the attendance status of the student is investigated, with the result that either the part_time_count, or the full_time_count accumulator is incremented. Note the use of indentation to emphasize the THEN and ELSE options, and the use of the delimiter ENDIF to close the operation.

6. A Computer Can Repeat a Group of Actions

When there is a sequence of processing steps that need to be repeated, two special keywords, DOWHILE and ENDDO, are used in pseudocode. The condition for the repetition of a group of actions is established in the DOWHILE clause, and the actions to be repeated are listed beneath it. For example:

```
DOWHILE product_total < 50
  Read product record
  Write product_number, product_name
      to report
  Add 1 to product_total
ENDDO
```

In this example, it is easy to see the statements that are to be repeated, as they immediately follow the DOWHILE statement and are indented for added emphasis. The condition that controls and eventually terminates the repetition is established in the DOWHILE clause, and the keyword ENDDO acts as a delimiter. As soon as the condition for repetition is found to be false, control passes to the next statement after the ENDDO.

V. MEANINGFUL NAMES

It is a good idea to introduce some unique names to represent the variables or objects in the problem, and to describe the processing steps. All names should be meaningful. A name given to a variable is simply a method of identifying a particular storage location in the computer.

The uniqueness of a name will differentiate it from other locations. Often a name chosen describes the type of data stored in a particular variable. For instance, a variable may be one of the three simple data types: an integer, a real number, or a character. The name itself should be transparent enough to adequately describe the variable: number1, number2, and number3 are more meaningful names for three numbers than A, B, and C.

Underscores are useful when choosing variable names, as the underscore is used as a word separator, for example, sales_tax and word_count. Another form of word separation is the use of a capital letter in the second word of a variable name, for example, salesTax and wordCount. Most programming languages do not tolerate a space in a variable name.

When it comes to writing down the processing steps in an algorithm, you should use words that describe the work to be done in terms of single, specific tasks or functions. For example:

```
Read three numbers
Multiply numbers together
Print total number
```

There is a pattern in the words chosen to describe these steps. Each action is described as a single verb followed by a two-word object. Studies have shown that if you follow this convention to describe a processing step, two benefits will result. First, you are using a disciplined approach to defining the problem and second, the processing is being dissected into separate tasks or functions. The simple operation of dividing a problem into separate functions and choosing a proper name for each function is extremely important when considering algorithm modules.

VI. THE STRUCTURE THEOREM

The Structure Theorem revolutionized program design by establishing a structured framework for representing a solution. The Structure Theorem states that it is possible to write any computer program by using only three basic control structures that are easily represented in pseudocode: sequence, selection, and repetition.

A. The Three Basic Control Structures

1. Sequence

The sequence control structure is the straightforward execution of one processing step after another. In pseudocode we represent this construct as a sequence of pseudocode statements.

```
statement a
statement b
statement c
```

The sequence control structure can be used to represent the first four basic computer operations listed previously: to receive information, put out information, perform arithmetic, and assign values. For example, a typical sequence of statements in an algorithm might read:

```
Add 1 to page_count
Write heading line1
Write heading line2
Set linecount to 2
Read product record
```

These instructions illustrate the sequence control structure as a straightforward list of steps, written one after the other, in a top-to-bottom fashion. Each instruction will be executed in the order in which it appears.

2. Selection

The selection control structure is the presentation of a condition and the choice between two actions, the choice depending on whether the condition is true or false. This construct represents the decision-making abilities of the computer and is used to illustrate the fifth basic computer operation, namely to compare two variables and select one of two alternate actions. In pseudocode, selection is represented by the keywords IF, THEN, ELSE and ENDIF:

```
IF condition p is true THEN
  statement(s) in true case
ELSE
  statement(s) in false case
ENDIF
```

The condition in the IF statement is based on a comparison of two items, and is usually expressed with one of the following relational operators:

<	less than
>	greater than
=	equal to
<=	less than or equal to
>=	greater than or equal to
<>	not equal to

The IF statement requires that if condition p is true then the statement or statements in the true case will be executed, and the statements in the false case will be skipped. Otherwise, (the ELSE statement) the statements in the true case will be skipped and statements in the false case will be executed. In either case, control then passes to the next processing step after the delimiter ENDIF.

There are a number of variations of the selection structure, as follows:

a. Simple selection (simple IF statement)

Simple selection occurs when a choice is made between two alternate paths, depending on the result of a condition being true or false.

```
IF bank_account_balance < $300 THEN
  service_charge = $5.00
ELSE
  service_charge = $2.00
ENDIF
```

Only one of the THEN or ELSE paths will be followed, depending on the result of the condition in the IF clause.

b. Simple selection with null false branch (null ELSE statement)

The null ELSE structure is a variation of the simple IF structure. It is used when a task is performed only when a particular condition is true. If the condition is false, then no processing will take place, and the IF statement will be bypassed. For example:

```
IF student_attendance_status =
    part_time THEN
  add 1 to part_time_count
ENDIF
```

In this case, the part_time_count field will only be altered if the student's attendance status is part-time.

c. Combined selection (combined IF statement)

A combined IF statement is one that contains multiple conditions, each connected with the logical operators AND or OR. If the connector AND is used to combine the conditions, then *both* conditions must be true for the combined condition to be true. For example:

```
IF student_attendance_status =
    part_time
AND student_gender = male THEN
  add 1 to male_part_time_count
ENDIF
```

In this case, each student record will undergo two tests. Only those students who are male, and who attend part-time will be selected, and the variable male_part_time_count will be incremented. If either condition is found to be false, then the counter will remain unchanged.

If the connector OR is used to combine any two conditions, then only *one* of the conditions needs to

be true for the combined condition to be considered true. If neither condition is true, then the combined condition is considered false. Changing the AND in the above example to OR dramatically changes the outcome from the processing of the IF statement.

```
IF student_attendance_status =
    part_time
OR student_gender = male THEN
  add 1 to male_part_time_count
ENDIF
```

In this example, if either or both conditions are found to be true, the combined condition will be considered true. That is, the counter will be incremented: if the student is part-time, regardless of gender; and if the student is male, regardless of attendance pattern.

Only those students who are not male, and not part-time will be ignored. So male_part_time_count will contain the total count of male part-time students, female part-time students, and male full-time students. As a result male_part_time_count is no longer a meaningful name for this variable. You must fully understand the processing that takes place when combining conditions with the AND or OR logical operators.

More than two conditions can be linked together with the AND or OR operators. However, if both operators are used in the one IF statement, parentheses must be used to avoid ambiguity. Look at the following example:

```
IF record_code = "99"
OR update_code = delete
AND account_balance = zero THEN
  delete customer record
ENDIF
```

The logic of this statement is confusing. It is uncertain whether the first two conditions should be grouped together and operated on first, or the second two conditions should be grouped together and operated on first. Pseudocode algorithms should never be ambiguous. There are no precedence rules for logical operators in pseudocode, but there are precedence rules in most programming languages. Therefore parentheses must be used in pseudocode to avoid ambiguity as to the meaning intended, as follows:

```
IF (record_code = "99"
OR update_code = delete)
AND account_balance = zero THEN
  delete customer record
ENDIF
```

The IF statement is now no longer ambiguous, and it is clear as to what conditions are necessary for the customer record to be deleted: the record will only be deleted if the account balance equals zero and either the record code = 99 or the update code = delete.

i. The NOT Operator The NOT operator can also be used for the logical negation of a condition, as follows:

```
IF NOT (record_code = "99") THEN
  update customer record
ENDIF
```

Here, the IF statement will be executed for all record codes other than code "99," that is for record codes *not* equal to "99."

Note that the AND and OR operators can also be used with the NOT operator, but great care must be taken and parentheses used, to avoid ambiguity, as follows:

```
IF NOT (record_code = "99"
  AND update_code = delete) THEN
  update customer record
ENDIF
```

Here, the customer record will only be updated if the record code is not equal to "99" AND the update code is not equal to delete.

d. Nested selection (nested IF statement)

Nested selection occurs when the word IF appears more than once within an IF statement. Nested IF statements can be classified predominantly as linear or non-linear.

i. Linear Nested IF Statements The linear nested IF statement is used when a field is being tested for various values and a different action is to be taken for each value. This form of nested IF is called linear because each ELSE immediately follows the IF condition to which it corresponds. Comparisons are made until a true condition is encountered, and the specified action is executed until the next ELSE statement is reached. Linear nested IF statements should be indented for readability, with each IF, ELSE, and corresponding ENDIF aligned. For example:

```
IF record_code = 1 THEN
    increment counter_1
  ELSE
  IF record_code = 2 THEN
    increment counter_2
  ELSE
  IF record_code = 3 THEN
    increment counter_3
  ELSE
    increment error_counter
  ENDIF
  ENDIF
ENDIF
```

Note that there are an equal number of IF, ELSE, and ENDIF statements, and that the correct indentation makes it easy to read and understand.

ii. Nonlinear Nested* IF *Statements A nonlinear nested IF occurs when a number of different conditions need to be satisfied before a particular action can occur. It is termed nonlinear because the ELSE statement may be separated from the IF statement with which it is paired. Indentation is once again important when expressing this form of selection in pseudocode. Each ELSE statement should be aligned with the IF condition to which it corresponds. For example:

```
IF student_attendance_status =
     part_time THEN
  IF student_gender = male THEN
    IF student_age > 21 THEN
      add 1 to mature_male_pt_students
    ELSE
      add 1 to young_male_pt_students
    ENDIF
  ELSE
    add 1 to female_pt_students
  ENDIF
ELSE
  add 1 to full_time_students
ENDIF
```

Note that there are an equal number of IF conditions as ELSE and ENDIF statements. Using correct indentation helps to see which pair of IF and ELSE statements match. However, nonlinear nested IF statements may contain logic errors that are difficult to correct so they should be used sparingly in pseudocode. If possible, replace a series of nonlinear nested IF statements with a combined IF statement. This replacement is possible in pseudocode because two consecutive IF statements act like a combined IF statement which uses the AND operator. Take as an example the following nonlinear nested IF statement:

```
IF student_attendance_status =
     full_time THEN
  IF student_age > 21 THEN
    increment mature_ft_student
  ENDIF
ENDIF
```

This can be written as a combined IF statement;

```
IF student_attendance_status =
     full_time
AND student_age > 21 THEN
  increment mature_ft_student
ENDIF
```

The same outcome will occur for both pseudocode expressions but the format of the latter is preferred, if the logic allows it, simply because it is easier to understand.

e. The case structure

The case control structure in pseudocode is another way of expressing a linear nested IF statement. It is used in pseudocode for two reasons: it can be directly translated into many high-level languages, and it makes the pseudocode easier to write and understand.

Case is not really an additional control structure. It simplifies the basic selection control structure and extends it from a choice between two values, to a choice from multiple values. In one case structure, several alternative logical paths can be represented. In pseudocode, the keywords CASE OF and ENDCASE serve to identify the structure, with the multiple values indented, as follows:

```
CASE OF single variable
  value_1 : statement block_1
  value_2 : statement block_2
    :
  value_n : statement block_n
  value_other: statement block_other
ENDCASE
```

The path followed in the case structure depends on the value of the variable specified in the CASE OF clause. If the variable contains value_1, statement block_1 is executed; if it contains value_2, statement block_2 is executed, and so on. The value_other is included in the event that the variable contains none of the listed values. For example:

```
CASE OF record_code
  1 : increment counter_1
  2 : increment counter_2
  3 : increment counter_3
  other : increment error_counter
ENDCASE
```

3. Repetition

The Structure Theorem introduces repetition as the third control structure. The repetition control structure is the presentation of a set of instructions to be performed repeatedly, as long as a condition is true. The basic idea of repetitive code is that a block of statements is executed again and again, until a terminating condition occurs. This construct represents the sixth basic computer operation, namely to repeat a group of actions. It is written in pseudocode as:

```
DOWHILE condition p is true
  statement block
ENDDO
```

The DOWHILE loop is a leading decision loop, that is, the condition is tested before any statements are executed. If the condition in the DOWHILE statement is found to be true, then the block of statements following that statement is executed once. The delimiter ENDDO then triggers a return of control to the retesting of the condition. If the condition is still true, the statements are repeated, and so the repetition process continues until the condition is found to be false. Control then passes to the statement that follows the ENDDO statement. It is imperative that at least one statement within the statement block can alter the condition and eventually render it false, because otherwise the logic may result in an endless loop.

There are a number of ways of representing the repetition structure in pseudocode.

a. Repeating a set of instructions a known number of times

When a set of instructions is to be repeated a specific number of times, then a counter is used in pseudocode, which is initialized before the DOWHILE statement and incremented just before the ENDDO statement, as follows:

```
Set product_total to zero
DOWHILE product_total < 50
  Read product record
  Write product_number, product_name
     to report
  Add 1 to product_total
ENDDO
```

This example illustrates a number of points:

1. The variable product_total is initialized before the DOWHILE condition is executed
2. As long as product_total is less than 50, (i.e., the DOWHILE condition is true) the statement block will be repeated.
3. Each time the statement block is executed, the variable product_total is incremented.
4. After 50 iterations, product_total will equal 50, which causes the DOWHILE condition to become false and the repetition to cease.

It is important to realize that the initializing and subsequent incrementing of the variable tested in the condition is an essential feature of the DOWHILE construct. Sometimes the keywords WHILE ... DO and ENDWHILE are used instead of DOWHILE and ENDDO to start and end this operation, as follows:

```
Set product_total to zero
WHILE product_total < 50 DO
  Read product record
  Write product_number, product_name
     to report
  Add 1 to product_total
ENDWHILE
```

Note that the format, indentation, and operation of both DOWHILE and WHILE ... DO are exactly the same. The only difference is the actual keywords that are used.

b. Repeating a set of instructions an unknown number of times

i. When a Trailer Record or Sentinel Exists When there is a trailer record or sentinel to signify the end of the input data, the DOWHILE clause must test for the trailer record or sentinel. However, there must be an initial, or priming read before the DOWHILE statement, otherwise the first test for the trailer record will be invalid. Look at the following example:

Example 1 Print Examination Marks

A program is required to read and print a series of names and examination marks for students enrolled in a history course. The class average mark is to be computed and printed at the end of the report. Marks can range from 0 to 100. The last record contains a blank name and a mark of 999 and is not to be included in the calculations.

You will need to consider the following requirements:

- A DOWHILE structure to control the reading of examination marks, until it reaches a mark of 999
- Accumulators for total mark and total students

Solution Algorithm

```
Print_examination_marks
  Set total_mark to zero
  Set total_students to zero
  Read student_name, exam_mark
  DOWHILE exam_mark NOT = 999
    Add 1 to total_students
    Print student_name, exam_mark
    Add exam_mark to total_mark
    Read student_name, exam_mark
  ENDDO
  IF total_students NOT = zero THEN
    average_mark = total_mark /
       total_students
    Print average_mark
  ENDIF
END
```

In this example the DOWHILE condition tests for the trailer record or sentinel (Record_999). However,

this test cannot be made until at least one exam mark has been read. Hence, the initial processing which sets up the condition is a Read statement immediately before the DOWHILE clause (Read student_name, exam_mark). The algorithm will require another Read statement, this time within the body of the loop. Its position is also important. The trailer record or sentinel must not be included in the calculation of the average mark, so each time an exam mark is read, it must be tested for a 999 value, before further processing can take place. For this reason, the Read statement is placed at the end of the loop, immediately before ENDDO, so that its value can be tested when control returns to the DOWHILE condition. As soon as the trailer record has been read, control will exit from the loop to the next statement after ENDDO—the calculation of average_mark.

ii. When a Trailer Record or Sentinel Does Not Exist

When there is no trailer record or sentinel to signify the end of the data, the DOWHILE clause must check for end-of-file. In pseudocode, the following terms can be used and are equivalent:

```
DOWHILE more data,
DOWHILE more records,
DOWHILE records exist,
DOWHILE not EOF,
WHILE success DO
```

In this case, all statements between the words DOWHILE and ENDDO will be repeated until an attempt is made to read a record, but no more records exist. When this occurs, a signal is sent to the program to indicate that there are no more records, and so the "DOWHILE more records" or "DOWHILE not EOF" clause is rendered false. Look at the following example:

Example 2 Process Student Enrollments

A program is required which will read a file of student records, and select and print only those students enrolled in a course unit named Mathematics I. Each student record contains student number, name, address, post code, gender, and course unit number. The course unit number for Mathematics I is M1222. Three totals are to be printed at the end of the report: total females enrolled in the course; total males enrolled in the course; and total students enrolled in the course.

You will need to consider the following requirements:

- A DOWHILE structure to perform the repetition
- IF statements to select the required students
- Accumulators for the three total fields

iv. Solution Algorithm

```
Process_student_enrollments
  Set total_females_enrolled to zero
  Set total_males_enrolled to zero
  Set total_students_enrolled to zero
  Read student record
  DOWHILE records exist
    IF course_unit_number = M1222 THEN
      print student details
      increment total_students_enrolled
      IF student_gender = female THEN
        increment total_females_
           enrolled
      ELSE
        increment total_males_enrolled
      ENDIF
    ENDIF
    Read student record
  ENDDO
  Print total_females_enrolled
  Print total_males_enrolled
  Print total_students_enrolled
END
```

c. Repetition using the REPEAT . . . UNTIL structure

The REPEAT . . . UNTIL structure is similar to the DOWHILE structure, in that a group of statements are repeated in accordance with a specified condition. However, where the DOWHILE structure tests the condition at the *beginning* of the loop, a REPEAT . . . UNTIL structure tests the condition at the *end* of the loop. This means that the statements within the loop will be executed once before the condition is tested. If the condition is false, the statements will then be repeated UNTIL the condition becomes true.

The format of the REPEAT . . . UNTIL structure is

```
REPEAT
  statement,
  statement
    :
UNTIL condition is true
```

You can see that REPEAT . . . UNTIL is a trailing decision loop; the statements are executed once before the condition is tested. There are two other considerations about which you need to be aware, before using REPEAT . . . UNTIL. First, REPEAT . . . UNTIL loops are executed when the condition is false; it is only when the condition becomes true, that repetition ceases. Thus, the logic of the condition clause of the REPEAT . . . UNTIL structure is the opposite of

DOWHILE. For instance, "DOWHILE more records" is equivalent to "REPEAT ... UNTIL no more records" and "DOWHILE number not = 99" is equivalent to "REPEAT ... UNTIL number = 99."

Second, the statements within a REPEAT ... UNTIL structure will always be executed at least once. As a result, there is no need for a priming Read when using REPEAT ... UNTIL. One Read statement at the beginning of the loop is sufficient, however, an extra IF statement immediately after the Read statement must be included, to prevent the processing of the trailer record. Look at the following example:

```
Process_student_records
  Set student_count to zero
  REPEAT
    Read student record
    IF student number not = 999 THEN
      Write student record
      Increment student_count
    ENDIF
  UNTIL student number = 999
  Print student_count
END
```

REPEAT ... UNTIL loops are used less frequently in pseudocode than DOWHILE loops for sequential file processing because of this extra IF statement required within the loop. The above example could have been written as a DOWHILE, as follows:

```
Process_student_records
  Set student_count to zero
  Read student record
  DOWHILE student number not = 999
    Write student record
    Increment student_count
    Read student record
  ENDDO
  Print student_count
END
```

d. Counted repetition

Counted repetition occurs when the exact number of loop iterations is known in advance. The execution of the loop is controlled by a loop index, and instead of using DOWHILE, or REPEAT ... UNTIL, the simple keyword DO is used as follows:

```
DO loop_index = initial_value to
    final_value
  statement block
ENDDO
```

The DO loop does more than just repeat the statement block. It will

1. Initialize the loop_index to the required initial_value.
2. Increment the loop_index by 1 for each pass through the loop.
3. Test the value of loop_index at the beginning of each loop to ensure that it is within the stated range of values.
4. Terminate the loop when the loop_index has exceeded the specified final_value.

In other words, a counted repetition construct will perform the initializing, incrementing, and testing of the loop counter automatically. It will also terminate the loop once the required number of repetitions have been executed.

Example 3 Fahrenheit–Celsius Conversion

Every day, a weather station receives 25 temperatures expressed in degrees Fahrenheit. A program is to be written which will accept each Fahrenheit temperature, convert it to Celsius, and display the converted temperature to the screen. After 25 temperatures have been processed, the words "All temperatures processed" are to be displayed on the screen.

Solution Algorithm The solution will require a DO loop and a loop counter (temperature_count) to process the repetition.

```
Fahrenheit_Celsius_conversion
  DO temperature_count = 1 to 25
    Prompt operator for f_temp
    Get f_temp
    Compute c_temp = (f_temp - 32) *
        5/9
    Display c_temp
  ENDDO
  Display `All temperatures processed'
      to the screen
END
```

Note that the DO loop controls all the repetition:

- It initializes temperature_count to 1.
- It increments temperature_count by 1 for each pass through the loop.
- It tests temperature_count at the beginning of each pass to ensure that it is within the range 1-25.
- It automatically terminates the loop once temperature_count has exceeded 25.

VII. MODULARIZATION AND PSEUDOCODE

Modularization is the process of dividing a problem into separate tasks or functions, each with a single purpose. By dividing a complex problem into separate tasks, or modules, the programmer can concentrate on the overall design of the algorithm before getting involved with the finer details. Each of the subtasks or functions becomes a module within the solution algorithm or program.

A module can be defined as a section of an algorithm, which is dedicated to a single function. The use of modules makes the algorithm simpler, more systematic, and more likely to be free of errors. Since each module represents a single task, the programmer can develop the solution algorithm task by task, or module by module, until the complete solution has been devised. Separate modules, once identified and written, are easily understood, can be reused, and can be independently modified if necessary.

A module must be large enough to perform its task, and must include only the operations, which contribute to the performance of that task. It should have a single entry, and a single exit with a top-to-bottom sequence of instructions. The name of the module should describe the work to be done as a single specific function. The convention of naming a module by using a verb, followed by a two-word object is particularly important here, as it helps to identify the separate task or function which the module has been designed to perform. Also, the careful naming of modules using this convention makes the algorithm and resultant code easier to follow. For example, typical module names might be:

```
Print_page_headings
Calculate_sales_tax
Validate_input_date
```

By using meaningful module names such as these, you can describe the task that the module has been designed to perform.

A. The Mainline

Since each module performs a single specific task, a mainline routine must provide the master control, which ties all the modules together and coordinates their activity. This program mainline should show the main processing functions, and the order in which they are to be performed. In pseudocode, the modules are simply named, or are preceded by the keyword, `Call`, as in the following example.

Example 4 Read Three Numbers

Design a solution algorithm, which will prompt a terminal operator for three numbers, accept those numbers as input, sort them into ascending sequence, and output them to the screen. The algorithm is to continue to read numbers until the number 999 is entered.

In the solution algorithm, the problem has been divided into three separate tasks and a mainline, which will form the modules in the program. The three tasks are to "Get three numbers," "Sort the three numbers," and "Display the sorted numbers."

B. Solution Algorithm

1. Mainline

```
Read_three_numbers
  Get_three_numbers
  DOWHILE NOT (number_1 = X AND
      number_2 = X AND number_3 = X)
    Display_three_numbers
    Sort_three_numbers
    Get_three_numbers
  ENDDO
END
```

2. Other Modules

```
Get_three_numbers
  Prompt the operator for number_1,
      number_2, number_3
  Get number_1, number_2, number_3
END

Display_three_numbers
  Display to the screen number_1,
      number_2, number_3
END

Sort_three_numbers
  IF number_1 > number_2 THEN
    temp = number_1
    number_1 = number_2
    number_2 = temp
  ENDIF
  IF number_2 > number_3 THEN
    temp = number_2
    number_2 = number_3
    number_3 = temp
  ENDIF
```

```
    IF number_1 > number_2 THEN
      temp = number_1
      number_1 = number_2
      number_2 = temp
    ENDIF
  END
```

When the mainline wants to pass control to its submodule, it simply names that module. Control will then pass to the called module and when the processing in that module is complete, the module will pass control back to the mainline. The resultant mainline is simple and easy to read.

VIII. COMMUNICATION BETWEEN MODULES

One method of intermodule communication is the passing of parameters or arguments between modules. Parameters are simply data items transferred from a calling module to its subordinate module at the time of calling. When the subordinate module terminates and returns control to its caller, the values in the parameters are transferred back to the calling module. When a calling module calls a subordinate module in pseudocode, it must consist of the name of the called module with a list of the parameters, enclosed in parentheses, to be passed to the called module, for example:

```
Print_page_headings (page_count,
    line_count)
```

The called module will also have, following its name, a list of parameters, which it expects to receive from the calling module. The parameter names of the respective modules need not be the same (although it helps readability if they are) but their number, type, and order must be identical.

Parameters may have one of three functions:

1. To pass information from a calling module to a subordinate module. The subordinate module would then use that information in its processing, but would not need to communicate any information back to the calling module.
2. To pass information from a subordinate module to its calling module. The calling module would then use that parameter in subsequent processing.
3. To fulfill a two-way communication role. The calling module may pass information to a subordinate module, where it is amended in some fashion and then passed back to the calling module.

Example 5 Read Three Numbers

Design a solution algorithm, which will prompt a terminal operator for three numbers, accept those numbers as input, sort them into ascending sequence, and output them to the screen. The algorithm is to continue to read numbers until the number 999 is entered.

The solution algorithm is exactly the same as in Example 4, except that parameters are passed between the mainline and each module.

A. Solution Algorithm

1. Mainline

```
Read_three_numbers
  Get_three_numbers (number_1,
      number_2, number_3)
  DOWHILE NOT (number_1 = X AND
      number_2 = X AND number_3 = X)
    Display_three_numbers (number_1,
        number_2, number_3)
    Sort_three_numbers (number_1,
        number_2, number_3)
    Get_three_numbers (number_1,
        number_2, number_3)
  ENDDO
END
```

2. Other Modules

```
Get_three_numbers (number_1,
    number_2, number_3)
  Prompt the operator for number_1,
      number_2, number_3
  Get number_1, number_2, number_3
END

Display_three_numbers (number_1,
    number_2, number_3)
  Display to the screen number_1,
      number_2, number_3
END

Sort_three_numbers (number_1,
    number_2, number_3)
  IF number_1 > number_2 THEN
    temp = number_1
    number_1 = number_2
    number_2 = temp
  ENDIF
```

```
    IF number_2 > number_3 THEN
      temp = number_2
      number_2 = number_3
      number_3 = temp
    ENDIF
    IF number_1 > number_2 THEN
      temp = number_1
      number_1 = number_2
      number_2 = temp
    ENDIF
  END
```

This example shows intermodule communication with the use of parameters. The module `Get_three_numbers` prompts for and receives three numbers from the screen and sends them to the mainline in the form of parameters. The mainline then sends these parameters to the module `Sort_three_characters` which sorts the numbers and returns them to the mainline. The mainline then sends these parameters to the module `Display_three_characters` which displays them to the screen. The advantage of using intermodule communication is that each module is completely independent and may even be written by different programmers.

IX. PSEUDOCODE AND OBJECT-ORIENTED DESIGN

Object-oriented design is concerned with identifying the problem as a set of objects rather than a set of functions. An object can be considered as a container for a set of data and the operations that need to be performed on it. An object has the following properties:

- It has a *name* which is unique for the lifetime of the object.
- It has data in the form of a set of characteristics or *attributes,* each of which has a value at any given point in time.
- There is a set of operations or *methods* which can be performed on the data.
- It is an instance (example) of a *class.*

An object is created from a template or pattern, called a *class,* which defines the basic attributes and the methods available to objects of that class. When an object is created it is given a unique name and a copy of all the attributes and methods from its class. The process of creating objects from classes is called *instantiation.* Many objects can be instantiated from a class, each containing the same attributes but not necessarily storing the same values in those attributes.

A. Constructors

The process of instantiating an object from the class template is performed by a special method, or set of instructions, known as a *constructor.* Until the constructor is called, no object exists. As each object is created, it receives its own copy of all the attributes and methods for its class. Every class should have a default constructor, which puts actual values into the attributes. In pseudocode, the format for this constructor is:

```
Create objectName as new Class()
```

The constructor may:

- Have no parameters, in which case a new object is assigned all the default values for its attributes.
- May have parameters that initialize the attributes with those particular values.

For example, the pseudocode

```
Create timeSheet as new TimeSheet()
```

will create a new object called `timeSheet` which is assigned all the default values for its attributes from the class `TimeSheet`. Similarly, the pseudocode

```
Create student as new Student
    (studentNum)
```

will create a new object called student which is assigned all the same attributes as the class Student, but is initialized with a unique student number, called `studentNum`, which is passed to it as a parameter.

More than one object can be instantiated from a class. For example:

```
Create student1 as new Student
    (studentNum)
Create student2 as new Student
    (studentNum)
```

When there is more than one object instantiated from a class, it is important that the program can identify exactly which copy of a method it should be using when a method call is made. This is managed by using a special notation in which the object's name is placed in front of the method, separated by the "dot operator." For example a method called `printFinalMark`, which prints a student's final mark, could be called by each of the student objects, `student1` and `student2`, as follows:

```
student1.printFinalMark()
student2.printFinalMark()
```

X. SUMMARY

Pseudocode is a set of English-like words and conventions that are used to represent an algorithm when designing a solution to a computer-programming problem. This article introduced the keywords and conventions used when writing pseudocode. The three control structures stated in the Structure Theorem, sequence, selection, and repetition, were extensively covered with many examples for their use. Pseudocode for algorithms with modules and object-oriented design were also covered. Much of the material for this article was adapted from *Simple Program Design, 3rd Ed.,* by Lesley Anne Robertson and published by Nelson Thomson Learning, 2000.

ACKNOWLEDGMENT

Portions of this article were adapted by permission from Robertson, L. A. (2000). *Simple Program Design, 3rd Ed.,* Chapters 2 to 5, published by Nelson Thomson Learning, South Melbourne Vic, Australia.

SEE ALSO THE FOLLOWING ARTICLES

Computer Assisted Systems Engineering (CASE) • Data Flow Diagrams • Error Detecting and Correcting Codes • Evolutionary Algorithms • Flowcharting Techniques • Object-Oriented Programming • Structured Design Methodologies

BIBLIOGRAPHY

Baase, S., and Van Gelder, A. (1999). *Computer algorithms, introduction to design and analysis.* Reading, MA: Addison Wesley.

Heaven, S. (1995). *Introduction to software design.* London: Edward Arnold.

Horowitz, E. (1997). *Computer algorithms, pseudocode.* New York: Computer Science Press.

Juliff, P. (1990). *Program design, 3rd ed.* Sydney: Prentice Hall.

Robertson, L. A. (2000). *Simple program design, a step by step approach, 3rd ed.* South Melbourne, Australia: Nelson Thomson Learning.

Psychology

Sean B. Eom
Southeast Missouri State University

GLOSSARY

biological psychology Study of the biological underpinings of behavior to determine how the nervous system, hormones, genes, and other biological entities and processes interact with behavior.
clinical psychology Study of the assessment and treatment of psychological problems.
cognitive psychology Study of the human mind and is concerned with adult's normal, typical cognitive activities of knowing/perceiving/learning.
developmental psychology Study of changes in behavior and behavioral potential over the life span of an individual or group.
educational psychology The design, development, and evaluation of materials and procedures for education.
experimental psychology Study of a restricted set of problems, such as learning, sensation, and perception, human performance, motivation, emotion, language, thinking, and communication.
forensic (legal) psychology Applying psychology to law and legal proceedings (i.e., evaluating incoming prisoners, selecting a jury, providing testimony as expert witnesses, etc.).
industrial (organizational) psychology Study of work-related behavior in industrial organization.
neuropsychology The diagnosis of brain disorders and the design of rehabilitation programs that help patients recover after various types of brain disorders.
rehabilitation psychology Applies psychological theories to facilitate patients' recovery from various physical injuries.
social psychology Study of people interacting, their behavior in a group, and the causes, types, and consequences of human interaction.

PSYCHOLOGY is the scientific study of the mind, mental processes, and behavior of living organisms in order to understand, predict, control, and explain these aspects. A large number of psychologists appear to agree that there is no universal definition of psychology and that most likely, one will never exist. Psychology is a diverse field with many branches, including cognitive, industrial and organizational, social and behavioral, experimental, biological, developmental, clinical, educational, counseling, rehabilitation, philosophical, community, humanistic, population/environmental, and health, etc. All these diversified fields of psychology are interested in studying environmental forces, genetic forces, mental processes, and the free will enacting on the minds of living organisms. This article discusses the intellectual relationship between the management information systems (MIS) subareas and psychology, as well as examining the contributions of psychology to the development of the MIS areas.

An MIS is an integrated system of a set of interrelated elements of data, hardware, software, people, and procedure to increase the efficiency and improve the effectiveness of individuals, groups, departments, and organizations. Broadly speaking, MIS can be divided into two areas: computer information systems and subsystems (hardware, software, communication

technology, data, procedures, and users) and the use of information systems (1) to increase effectiveness/efficiency of the management process of planning, organizing, staffing, controlling, and coordinating; (2) to support each level of management from operational, middle, and top management and nonmanagement personnel, such as secretaries and rank and file workers; (3) to support each of the functional management areas, including marketing, production and operations, human resources, accounting, etc. Like many academic disciplines, the study of information systems is a multidisciplinary (or interdisciplinary) field. There has been a two-way flow of intellectual materials between the MIS area and other academic disciplines, such as management science/operations research, psychology, systems science, cognitive science, communications science, economics, accounting, etc. This article examines the contributions of psychology to the development of MIS subspecialties.

I. MAJOR TYPES OF COMPUTER-BASED INFORMATION SYSTEMS AND PSYCHOLOGY

The general term, computer-based information systems (CBIS), is a constellation of a variety of information systems, such as office automation systems (OAS), transaction processing systems (TPS), management information systems (MIS), and management support systems (MSS). Management support systems consist of decision support systems (DSS), expert systems (ES), artificial neural networks (ANN), and executive information systems (EIS)/executive support systems (ESS). Executive support systems are executive information systems with added decision support capabilities. The types of CBIS are based on the levels of management, ranging from nonmanagement level (secretaries, clerical workers, etc.), operational level management, and middle level management, to top level (senior) management.

Since the purpose of this article is to explain the relationship between psychology and CBIS, one useful method of looking at various types of information systems is to observe the degree of human/computer interactions to achieve the systems' intended objectives, but more specifically, the roles of the user in accomplishing those desired goals. We now examine each type of information system from this perspective of the degree of human/computer interaction. Several systems (OAS, TPS, MIS) require no or little human judgments and/or mental processes (cognitive processes, reasoning, learning, etc.) to interpret and use outputs from each system. (The primary function of MIS aggregates transaction is data compiled by the TPS in order to produce information on an organization's performance between two points in time and historical records.) Management information systems also include structured decision-making tools, such as calculating economic order quantity and reorder points using deterministic inventory management models.

The three systems from the top of the organizational pyramid are labeled as MSS, which are comprised of expert systems, DSS, and EIS/ESS. All these systems necessitate the use of the human mind and cognitive processes to process and use information. Expert systems in general embed human experiences and judgments in the form of rules, frame, and semantic networks as a part of the systems. Therefore, most ES outputs do not require human input to execute the decisions, although some expert systems are used as intelligent support systems.

The other two remaining systems (DSS and ESS) are the systems to which psychology has played a critical role in its design, implementation, and use. As Fig. 1 exhibits, the DSS/ESS area has borrowed theories/concepts from various reference disciplines to establish itself as an academic field. The focus of this section is placed on the role of psychology to the design, development, implementation, and use of DSS/ESS.

Despite psychology being a field with many diverse branches, including the aforementioned, all branches are concerned with studying the forces of environment, genetics, mental processes, and free will on the human mind. Of these numerous branches of psychology, cognitive psychology and social psychology are found to be the two most influential fields that have affected the establishment of DSS/ESS as an academic discipline.

II. INTELLECTUAL RELATIONSHIPS BETWEEN THE INFORMATION SYSTEM AREAS AND REFERENCE DISCIPLINES

The MIS area is a relatively young field of study as compared to, for example, economics, physics, philosophy, organizational behavior, etc. As a field of study continues to grow and become coherent, study of the intellectual development of the field is important. In a relatively new field such as MIS, understanding the process of intellectual development and evolution of thought is even more beneficial because

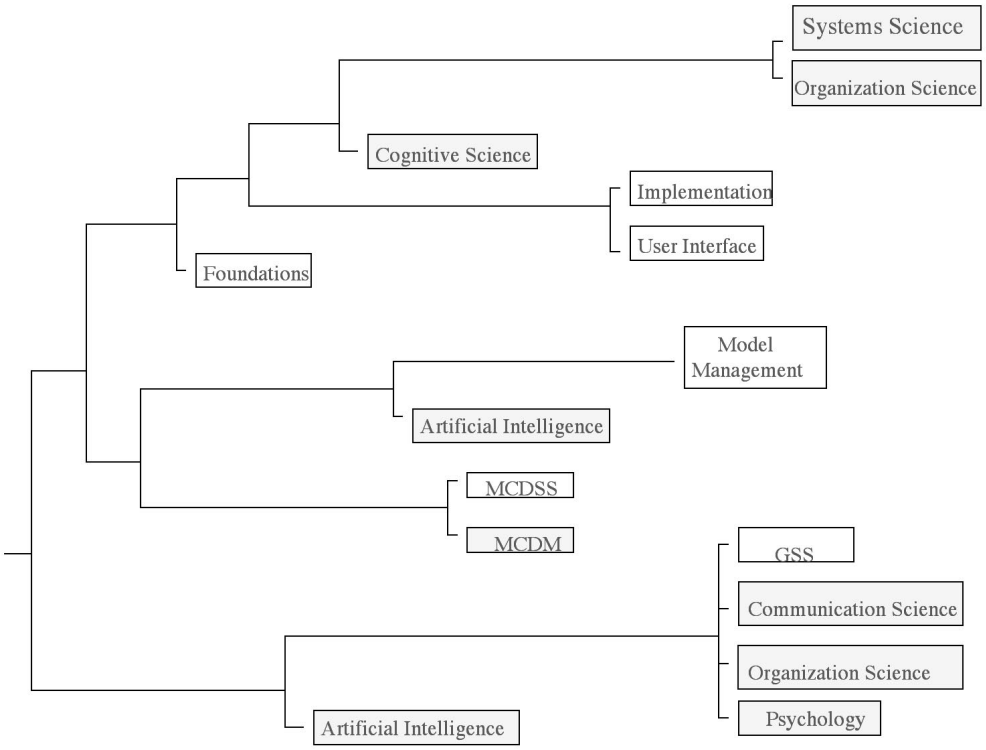


Figure 1 Dendogram illustrating hierarchical clustering. The dendrogram illustrates hierarchical clustering of six DSS research subspecialties (white rectangles) and eight reference disciplines (shaded rectangles). It also shows external heterogeneity between clusters.

it identifies the basic commitments that will serve as the foundations of the field as it matures.

Eom has conducted a series of studies to infer the intellectual structure of the DSS field by means of an empirical assessment of the DSS literature from 1971 through 1995. The cluster analysis of his study resulted in a dendrogram (tree graph), which illustrates hierarchical clustering (Fig. 1). The dendrogram shows that the DSS tree consists of two main branches. The first branch of the DSS tree represents all DSS subspecialties and related references disciplines, excluding group support systems (GSS) and their reference disciplines. The first branch holds five reference disciplines: artificial intelligence (AI), cognitive science/cognitive psychology, multiple criteria decision making (MCDM), systems science, and organization science. It also includes five major areas of DSS research: foundations, implementation, individual differences/user interfaces, model management, and multiple criteria/negotiation DSS.

The second branch consists of GSS and related reference disciplines (communication science, organization science, psychology, and AI).

III. REVIEW OF IMPORTANT PSYCHOLOGY THEORIES/CONCEPTS

Of these numerous fields of psychology as discussed earlier, cognitive psychology and social psychology have significantly influenced the formation of information systems subspecialties. The central component of cognitive psychology, which is the study of the human mind, is concerned with adults' normal, typical cognitive activities of knowing/perceiving/learning. Cognitive scientists view the human mind as an information processing system that receives, stores, retrieves, transforms, and transmits information (the computational or information processing view). The central hypothesis of cognitive psychology is that "thinking can best be understood in terms of representational structures in mind and computational procedures that operate on those structures." Cognitive science originated in the mid-1950s when researchers in several fields (philosophy, psychology, AI, neuroscience, linguistics, and anthropology) began to develop theories of the workings of the mind—particularly knowledge, its acquisitions, storage, and use in intelligent activity.

A major concern of cognitive psychology deals with the cognitive architecture (Fig. 2), referring to information processing structure and its capacity. The cognitive architecture is comprised of three subsystems: sensory input systems (vision, hearing, taste/smell, touch), central processing systems, and output systems (motor outputs and decision and actions). The central processing systems perform a variety of activities to process information gathered from the sensory input systems. They include memory and learning, such as processing/storing/retrieving visual (mental imagery) and auditory/ echoic information, and representing that information in memory. Other areas of central processing activities include problem solving (any goal-directed sequence of cognitive operations), reasoning, and creativity. Specifically, they include cognitive skills in problem solving; reasoning including that about probability; judgment and choice; recognizing pattern, speech sounds, words, and shapes; representing declarative and procedural knowledge; and structure and meaning of languages, including morphology and phonology.

The central processing system can be viewed for the information processing perspective as having a three-stage process:

1. Sensory memory can be characterized by large capacity, very short duration of storage, and direct association of with sensory processes. The initial attention process selects information for further processing.
2. Short-term memory (also known as working memory) has limited capacity where information selected from sensory memory is linked with information retrieved from long-term memory (past experiences and world knowledge) that forms the basis for our actions/behavior.
3. Long-term memory is characterized by large capacity, long duration. Our episodic (every day experiences) and semantic (world knowledge) information is stored in the long-term memory. Much of the research dealing with connective models, such as neural nets, focus on the structure of this long-term memory, although some researchers also expand the neural net perspective to include sensory and short-term memory. Topics such as language, concept formation, problem solving, reasoning, creativity, and decision making are usually associated with the structure and processes of this long-term memory. Also, the issue of "experts" versus "novices" is almost always linked with the structure of secondary memory.

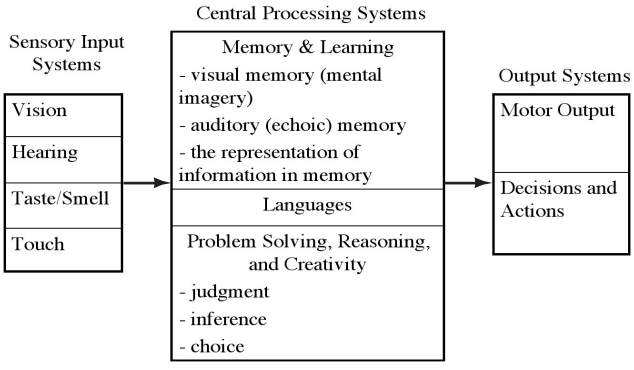


Figure 2 A global view of the cognitive architecture [adapted from Stillings, *et al.* (1995). *Cognitive Science: An Introduction,* (Second Edition). Cambridge, MA: MIT Press].

Perhaps, behavioral decision theory may be the most influential theory developed by cognitive scientists that has contributed toward the developments of DSS research subspecialities. Behavioral decision theory is concerned with normative and descriptive decision theories. The normative decision theory aims at "prescribing courses of action that conform most closely to the decision maker's beliefs and values." Behavioral decision theorists proposed decision theories solely on the basis of behavioral evidence, without presenting neurological internal constructs on which to base these mechanisms. Descriptive decision theory aims at "describing these beliefs and values and the manner in which individuals incorporate them into their decisions." Descriptive decision theories have focused on three main areas of studies: judgment, inference, and choice. The study of judgment, inference, and choice has been one of the most important areas of cognitive psychology research, which has been referenced most frequently by DSS researchers.

A. Judgment and Inference

The fundamental factor distinguishing DSS from any other CBIS is the use of judgment in every stage of the decision-making process, such as intelligence, design, choice, and implementation. The crucial part of cognitive psychology is the study of internal mental processes of knowing/learning/decision making, etc., mental limitations, and the impacts of the limitations on the mental processes.

Many DSS can help decision makers generate numerous decision alternatives. The decision makers use intuitive judgment of probability of future events, such as annual market growth rate, annual rate of in-

flation, etc. Tversky and Kahneman uncovered three heuristics employed when making judgment under uncertainty (representativeness, availability of instances, and adjustment from an anchor), which are usually effective, but often lead to systematic and predictable errors. Due to the cognitive limitations, the reliance on judgmental heuristics often leads to cognitive biases that eventually may cause ineffective decisions. The representativeness heuristic is applied when people are asked to judge the probability that object/event A belongs to class/process B. According to Tversky and Kahneman, the judgment of probability can be biased by many factors related to a human being's cognitive limitations, such as (1) insensitivity to prior probability of outcomes, (2) insensitivity to sample size, (3) the misconception of chance, (4) insensitivity to predictability, (5) the illusion of validity, and (6) the misconception of regression. The availability heuristic is used when people are asked to estimate the plausibility of an event. The employment of the availability heuristic may lead to predictable biases due to (1) the retrievableness of instances, (2) the effectiveness of a search set, (3) imaginableness, and (4) illusory correlation. The anchoring and adjustment effects can also bias the estimation of various quantities stated in percentage or in the form of probability distribution due to insufficient adjustment and/or biases in the evaluation of conjunctive and disjunctive events, etc.

Hogarth is another cognitive scientist whose profound insight on human judgment has made a substantial contribution to the DSS area. A substantial part of his research is devoted to compiling a catalogue of human judgmental fallibilities and information-processing biases in judgment and choice. He presented a conceptual model of judgment in which judgmental accuracy is described as a function of both individual characteristics and the structure of task environment within which the person makes judgments. Human judgments are based on information that has been processed and transformed by the limited human information processing capacity. He further decomposed the information processing activities of decision makers into: (1) acquisition of information; (2) processing information; (3) output; (4) action; and (5) outcome. He emphasized that judgmental biases can occur at every stage of information processing and that judgments are the result of interaction between the structure of tasks and the nature of human information processing systems. Decision aids are necessary in structuring the problem and assessing consequences, since intuitive judgment is inevitably deficient.

Einhorn and Hogarth reviewed behavioral decision theory to place it within a broad psychological context. In doing so, they emphasized the importance of attention, memory, cognitive representation, conflict, learning, and feedback to elucidate the basic psychological processes underlying judgment and choice. They concluded that decision makers use different decision processes for different tasks. The decision processes are sensitive to seemingly minor changes in the task-related factors.

B. Choice

Cognitive psychologists have long been interested in the area of problem solving. Payne and his colleagues attempted to understand the psychological/cognitive process that led to a particular choice or judgment using two process tracing methods—verbal protocol analysis and information acquisition behavior analysis. The verbal protocol is a record of the subject's ongoing behavior, taken from continuous verbal reports from the subject while performing the decision task (rather than through later questionnaires or interviews).

1. Four Information Processing Strategies

Payne and his colleagues focused on the identification of the information processing strategies/models and the task characteristics of the decision situation when choosing among multidimensional/multicriteria alternatives. Four of the most important decision models discussed in the cognitive psychology literature are the (1) additive/linear model of choice, (2) conjunctive model, (3) additive difference model, and (4) elimination-by-aspects (EBA) model.

The additive model allows the decision maker to choose the best candidate, the one with the greatest aggregated overall value. The conjunctive model assumes that an alternative must possess a certain minimum value on all dimensions in order to be chosen. The additive difference model directly compares just two alternatives at a time and retains only the better one for later comparison in order to reduce memory load. The two alternatives are compared directly on each attribute/dimension to determine a difference, and the results are added together to reach a decision. The selected alternative becomes the new standard against which each of the remaining alternatives is to be compared. The EBA model, like the additive difference model, is also an intradimensional strategy. The decision maker selects the most important dimension/attribute based on its relative importance. The first step of this process eliminates all alternatives for that attribute with values below the cut-off value.

This procedure will be repeated until we have all but one of the alternatives.

2. Choice of Specific Decision Strategy

Another important focal point of behavioral decision theorists' research has been the selection of a specific information processing strategy and the study of the factors that could change the choice of the specific processing strategy. These factors include information availability, time pressure, incomplete data, incommensurable data dimension, information overload, and the decision maker's intention to save time or increase accuracy.

Payne argued that the choice of specific decision strategy is contingent upon two task determinants: number of alternatives available and number of dimensions of information available per alternative. Analysis of the decision maker's information search pattern and verbal protocols suggested that task complexity is the major factor that determines a specific information processing strategy. When dealing with a two-alternative choice task, either additive or additive difference models are used, requiring the same amount of information on each alternative. On the other hand, both the conjunctive and elimination-by-aspects models were proposed to deal with a complex decision task consisting of more than six alternatives. These conjunctive and elimination-by-aspects models are good ways to eliminate some available alternatives quickly so that the amount of information being processed can be reduced in complex decision making.

3. Cost-Benefit (Effort-Accuracy) Framework

Payne examined cost-benefit framework investigating the effects of task and context variables on decision behavior. According to cost-benefit framework, a possible reason for choosing a specific decision model in a specific task environment is to maximize the expected benefits of a correct decision against the cost of using the process. This theory states that "decision makers trade-off the effort required to make a decision vis-a-vis the accuracy of the outcome." The strategies used vary widely based on small changes in the task or its environment. The decision maker frequently trades off small losses in accuracy for large savings in effort.

Payne and others further investigated effort and accuracy considerations in choosing information processing strategies, especially under time constraints. They found that under time pressure, several attribute-based heuristics, such as EBA and the lexicographic choice model, were more accurate than a normative procedure, such as expected value maximization. Under the severe time pressure, people focused on a subset of information and changed their information processing strategies.

IV. CONTRIBUTIONS OF PSYCHOLOGY TO THE DEVELOPMENT OF INFORMATION SYSTEMS SUBAREAS

A. Contributions to Information Systems Design/User Interfaces from Psychology

1. From Cost-Benefit Theory to Cognitive Fit Theory

The cost-benefit framework discussed above has been widely accepted by DSS researchers to assess the impact of decision aids used on effort expenditure by decision makers. Todd and Benbasat conducted numerous laboratory experiments to conclude that the use of a decision aid may result in effort savings, but not improved decision performance. In other words, DSS may aid the decision maker to be more efficient, but not to be more effective. Therefore, they suggested that DSS designers must consider the decision maker's trade-off between improving decision quality and conserving effort. Cost-benefit theory views problem solving in general as a trade-off between the effort to make a decision and the accuracy of the outcome, regardless of the characteristics of the tasks that must be performed.

Cognitive fit theory developed by Vessey and Vessey and Galletta can provide a useful guideline specifically for the designers of DSS for the tasks involving graphical and/or tabular representation of data in the decision-making process. Vessey and Galletta argue that supporting the task to be accomplished with the appropriate display format leads to minimization of both effort and error. Therefore, system designers should concentrate on determining the characteristics of the tasks that problem solvers must address, and supporting those tasks with the appropriate problem representation and support tools.

Cognitive fit theory explains "under what circumstances one representation outperforms the other." The theory states that:

1. Graphical and tabular representations emphasize spatial and symbolic information respectively.
2. Tasks can also be divided into two types—spatial and symbolic, based on the type of information that facilitates their solution.
3. Performances on a task will be enhanced when there is a cognitive fit (match) between the

information emphasized in the representation type and that required by the task type; that is, when graphs support spatial tasks and when tables support symbolic tasks.
4. The processes or strategies a problem solver uses are the crucial elements of cognitive fits since they provide the link between representation and task.
5. So long as there is a complete fit of representation, processes, and tasks type, each representation will lead to both quicker and more accurate problem solving.

2. Fallible Judgment: A Basis of DSS Design Theory

Behavioral decision theory is also used as a basis of proposing a theory of DSS design for user calibration, so that "the confidence a decision maker has in a decision made using the aid equals the quality of that decision." Behavioral decision scientists such as Einhorn and Hogarth and Tversky and Kahneman concluded that people are quite confident in very poor decisions and in their fallible judgment. Kasper's proposed DSS design theory for user calibration prescribes properties of a DSS (expressiveness, visibility, inquirability) needed for users to achieve perfect calibration, meaning that one's belief in the quality of a decision equals the objective quality of the decision.

3. Human Cognitive Limitations: A Basis of the ROMC Approach

The limitations of the human information processing system (a relatively slow serial processor with small short-term memory) and the study of cognitive biases contributed to the development of the ROMC approach to the user interface design. The ROMC approach emphasizes that a focus for user interface design is to provide users with familiar representations (graphs, plots, maps, charts, etc.) in order to communicate some aspect of the decision to other persons and that several types of memory aids should be provided to extend the users' limited memory.

4. Applied Psychology Research Aided the User-Interface Design

An applied information-processing psychology project team at the Xerox Palo Alto Research Center conducted requisite basic psychological research to generate a set of design principles, which aid in the design of computer systems for human-computer interaction. Their framework for applying psychology to computer system design consists of:

1. The structure and performance of the human/computer system
2. Performance of models for predicting the performance of the human/computer system
3. Design functions for using the performance models in the design process

Card and colleagues viewed that the role of an applied psychology is to supply performance models for the designer. The performance (independent variable) of a human/computer system can be specified as being determined by its structural (dependent) variables (task, user, and computer).

B. Contributions to Implementation from Cognitive Psychology

Researchers in the DSS implementation area have attempted to systematically identify the implementation success factors and the relationship between user-related factors (cognitive style, personality, demographics, and user-situational variables) and implementation success. The majority of DSS implementation researchers have expanded the implementation success factors to include other user-related factors, such as personality, demographics, and user-situational variables, in addition to cognitive styles. They asserted that the cognitive style of users/managers did affect the chances of implementations.

A theory of problem solving by Newell and Simon described the cognitive mechanisms and the nature of human information-processing systems. A theory of problem solving sheds some light on understanding how intelligent adults solve short (half-hour), moderately difficult problems of a symbolic nature, such as those in chess, symbolic logic, and algebra-like puzzles. According to their theory, the organization of the problem representation significantly influences the structure of the problem space and the problem-solving processes decision makers use. Therefore, when their problem-solving processes are adapted to the problem representation, decision makers make effective decisions, which lead to successful implementation of DSS.

C. Contributions to GSS from Psychology

1. Group Idea Generation and Cognitive Psychology

Group idea generation is an important part of GSS activities. Osborn argued that most human mental

capacities, such as absorption, retention, and reasoning, can be performed by computers, with the exception of the creative ability to generate ideas, and that nearly all humans have some imaginative talent. Osborn identified two broad classes of imagination—controllable and uncontrollable by the will of the individual. GSS researchers have focused on extending his idea concerning how human imagination that can be driven at the will of the individual can be further developed by GSS. Brainstorming is one of the most widely known approaches to idea generation. Osborn improved a traditional, unstructured process of group idea generation technique and provided a set of systematic rules for brainstorming groups to overcome several social psychological factors that usually inhibit the generation of ideas.

Numerous research has shown that nominal groups of noninteracting individuals have outperformed verbally brainstorming groups. Many psychologists investigated the most likely causes of productivity losses of brainstorming groups (production blocking, free riding, evaluation apprehension, etc.). A series of experiments by psychologists Diehl and Stroebe concluded that "individuals brainstorming alone, then pooling afterwards produces more ideas of a quality at least as high as do the same number of people brainstorming in a group" due to several possible reasons, such as evaluation apprehension, free riding, and production blocking. A significant finding of Diehl and Stroebe's experiments was their recognition of the magnitude of the impacts that production blocking has on the productivity loss of brainstorming groups. By manipulating blocking directly, Diehl and Stroebe were able to determine that production blocking accounted for most of the productivity loss of real brainstorming groups. Therefore, their findings suggest that it might be more effective to ask group members first to develop their ideas in individual sessions; then, these ideas could be discussed and evaluated in a group session.

Several studies conducted in the early 1990s showed that electronically brainstorming groups produced superior results to verbally brainstorming groups and nonelectronic nominal groups in terms of number of unique ideas generated. Group support systems researchers tried to answer the question of why electronically brainstorming groups generated a higher number of unique ideas, adopting the numerous research results of cognitive scientists. Nagasundaram and Dennis argued that "a large part of idea-generation behavior in electronic brainstorming (EBS) can be explained by viewing EBS as an individual, cognitive (rather than social) phenomenon from the human information processing system perspective."

Janis and Mann analyzed psychological processes involved in conflict, choices, commitment, and consequential outcomes and provided a descriptive conflict theory. Their theory is concerned with "when, how, and why psychological stress generated by decisional conflict imposes on the rationality of a person's decisions" and how people actually cope with the stresses of decisional conflicts. Based on the theoretical assumptions derived from extensive research on the psychology of stress, Janis and Mann (1977) provided a general theoretical framework for integrating diverse findings from psychological/behavioral science research and reviewed the main body of psychological/behavioral science research concerning the determinants of decisional conflicts.

An important goal in the study of group DSS (GDSS) is to minimize the dysfunctions of the group interaction process, such as evaluation apprehension, cognitive inertia, domination by a few individuals, etc. In designing GDSS to minimize the dysfunctions, GDSS researchers have sought to build on/extend the research results of group dynamics, which seeks the answer to the following question: How is behavior influenced by others *in a group?* In the area of group dynamics, Shaw and McGrath provided an integrative conceptual framework for synthesizing the voluminous body of group research and presented approaches to the study of groups. They examined factors that facilitate/inhibit group behavior and problem solving as an interrelated process of social interaction. The factors include the physical environment of groups, personal characteristics of group members, group composition, group structure, leadership, group tasks and goals, etc. According to McGrath, all groups can be classified as: vehicles for delivering social influence, structures for patterning social interaction, or task performances systems. He focused on the nature, the causes, and the consequences of "group interaction processes," defined as "dynamic interplay of individual and collective behavior of group members."

Siegel and others investigated the behavioral and social implications of computer-mediated communications and sought to answer the question: Do computer-mediated communications change group decision making? The results of their experiments suggest that simultaneous computer-mediated communication significantly affected efficiency, member participation, interpersonal behavior, and group choice, when compared to the face-to-face meeting. Using computerized communication, it took more time for group consensus, and fewer remarks were exchanged. However, more decision proposals were introduced.

Communication via the computer showed more equal participation of group members and more uninhibited communication; in addition, decisions deviated further from initial individual opinions. These results suggest computer-mediated communication is somewhat inefficient compared to face-to-face communication. Distraction and frustration in having to read and type messages simultaneously could provoke more uninhibited behavior.

D. Contributions to Model Management from Cognitive Psychology

Since 1975, model management has been determined to encompass several central topics, such as model base structure and representation, model base processing, and application of AI to model integration, construction, and interpretation.

1. Cognitive Psychology Aids in Problem Structuring

A group of DSS researchers is continuing to build DSS to support the problem-structuring phase. In this line of research, while building toward an interactive graphics-based problem-structuring aid, such as the Graphical Interactive Structural Modeling Option (GISMO), cognitive scientists have made important contributions in developing imagery theory, dual coding theory, and a theory of problem solving. According to Anderson, there are two competing theories of memories—propositional versus dual-code. The dual-code theorists believe that although other codes exist for other modalities, such as touch, taste, and smell, all memory is tied to a particular sensory modality, with the verbal and visual codes being the dominant ones stored in long-term memory. On the other hand, the propositional code theorists claim that representations in memory are abstract, and therefore, are not tied to a particular sensory modality.

Using GISMO, Loy found that the user's ability to create and use visual images is positively related to better problem-solving and -structuring performance. His findings imply that further DSS research is necessary to develop DSS tools which can provide effective support for decision makers who do not possess highly developed visual thinking skills.

2. Development of Graph-Based Modeling

In the same line of research, graph-based modeling is an emerging research area. Jones presented NETWORKS, a prototype system of graph-based modeling which allows the user to represent a wide variety of decision problems in a graphical form, such as bar chart, decision tree, decision network, etc. Further, the users manipulate the models (e.g., deleting/adding subtrees for decision trees) using a graph-grammar by applying a set of operations (or productions).

E. Contributions to Intelligent Information Systems/DSS from Cognitive Psychology

1. Development of Neural Networks

A significant contribution from cognitive psychology is the development of artificial neural networks. This area of research is often called connectionism, parallel distributed processing, or neurocomputing. Numerous individuals have contributed to the advancement of neural network systems. Among them, Rumelhart, McClelland, and the PDP research group (the research group formed to understand the nature of cooperative computation and to develop "neurally inspired" computational architectures) have been most influential in providing foundational concepts for the development of neural networks.

One of the essential study questions the cognitive scientists had in mind is to study the architecture of mind. Comparatively, the human brain works slowly, as it contains billions of brain cells. In addition, the brain must deploy the processing elements in the brain cooperatively and in parallel to carry out its activities. Human brains have the capability to think in parallel and in serial for any task requiring attention. Today, intelligent DSS combines traditional quantitative DSS tools with neural networks. Such DSS combined with neural networks produce synergistic impacts in improving the effectiveness of decision-making activities via enhancing the cognitive support for the user.

As reviewed, cognitive psychologists identified the cognitive limitations and biases associated with presenting data to a decision maker and within information processing. To overcome these human limitations, many DSS researchers suggested an expert system with embedded DSS to ameliorate these cognitive limitations.

2. Intelligent Agent Research

Intelligent agents (known also as intelligent interfaces, adaptive interfaces) research is an emerging

interdisciplinary research area involving researchers from such fields as ES, DSS, cognitive psychology, computer science, etc. According to Riecken, the primary purpose of agent research is to "develop software systems which *engage and help* all types of end users" in order to reduce work and information overload, teach, learn, and perform tasks for the user. In the early 1990s, the conversational framework for decision support was introduced as a basis of a new generation of active and intelligent DSS and EIS. The active DSS will be equipped with the tools (stimulus agents) that will act as experts, servants, or mentors to decide when and how to provide advice and criticism to the user, while the user formulates and inquires about his or her problems under the continuous stimulus of electronic agents.

F. Contributions to MCDSS from Cognitive Psychology

Behavioral (descriptive) decision strategies proposed by cognitive psychologists, such as Payne, and normative (prescriptive) strategies in the MCDM area are further incorporated into the design of multiple criteria decision support systems (MCDSS) to deal with multiple criteria decision problems. Minch and Sanders recommended a framework for building the MCDSS and showed that the conjunctive, disjunctive, lexicographic, and elimination-by-aspect models proposed by cognitive psychologists can be easily implemented in the MCDSS using rudimentary database query facilities.

V. SUMMARY AND CONCLUSION

We have discussed the intellectual relationships between the DSS subarea and examined the contributions of psychology to the development of the MIS areas. The fundamental factor distinguishing DSS from any other CBIS is the use of judgment in every stage of the decision-making process. The crucial part of cognitive psychology is the study of internal mental processes, mental limitations, and the impacts that the limitations have on the mental processes, such as revealing judgmental heuristics and exploring their impacts on decision making. Cognitive scientists emphasized that judgmental biases can occur at every stage of information processing and that judgments are the result of interaction between the structure of tasks and the nature of human information processing systems. Decision aids are necessary in structuring the problem and assessing consequences, since intuitive judgment is inevitably deficient. We have traced how concepts/theories in psychology have been further extended and refined in the development of information systems theories and concepts.

SEE ALSO THE FOLLOWING ARTICLES

Data, Information, and Knowledge • Ergonomics • Forensics • Health Care, Information Systems and • Human Side of Information, Managing the Systems • Intelligent Agents • Knowledge Representation • Law Firms • Medicine, Artificial Intelligence in • Neural Networks • Sociology • Systems Science

BIBLIOGRAPHY

Anderson, J. R. (1985). *Cognitive psychology and its implications,* 2nd ed., San Francisco, CA: W.H. Freeman and Co.

Beach, L. R., and Mitchell, T. R. (July 1978). A contingency model for the selection of decision strategies. *Academy of Management Review,* 3:3, 439–449.

Card, S. K., Moran, T. P., and Newell, A. (1983). *The psychology of human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Diehl, M., and Stroebe, W. (September 1987). Productivity loss in brainstorming groups: Toward the solution of a riddle. *Journal of Personality and Social Psychology,* 53:3, 497–509.

Einhorn, H. J., and Hogarth, R. M. (1981). Behavioral decision theory: Processes of judgment and choice. *Annual Review of Psychology,* 32, 53–88.

Eom, S. B. (1997). Assessing the current state of intellectual relationships between the decision support systems area and academic disciplines. *Proceedings of the Eighteenth International Conference on Information Systems,* (K. Kumar and J. I. DeGross, eds.), Atlanta, GA, December 15–17, 167–182.

Gallupe, R. B., Bastianutti, L.M., and Cooper, W. H. (1991). Unblocking brainstorms. *Journal of Applied Psychology,* 76:1, 137–142.

Hogarth, R. M. (1980). *Judgement and choice: The psychology of decision.* New York: Wiley.

Janis, I. L., and Mann, L. (1977). *Decision making: A psychological analysis of conflict, choice, and commitment.* New York: Free Press.

McGrath, J. E. (1984). *Groups: Interaction and performance.* Englewood Cliffs, NJ: Prentice-Hall.

Nagasundaram, M. I., and Dennis, A. R. (1993). When a group is not a group: The cognitive foundation of group idea generation. *Small Group Research,* 24:4, 463–489.

Newell, A., and Simon, H. A. (1972). *Human problem solving.* Englewood Cliffs, NJ: Prentice-Hall.

Payne, J. W. (1976). Task complexity and contingent processing decision making: An information search and protocol analysis. *Organizational Behavior and Human Performance,* 16:2, 366–387.

Payne, J. W. (1982). Contingent decision behavior. *Psychological Bulletin,* 92:2, 382–402.

Payne, J. W., Bettman, J., and Johnson, E. J. (July 1988). Adaptive strategy selection in decision making. *Journal of Experimental Psychology: Human Learning, Memory and Cognition,* 14:4, 534–552.

Payne, J. W., Braunstein, M. L., and Carroll, J. S. (1978). Exploring predecisional behavior: An alternative approach to decision research. *Organizational Behavior and Human Performance,* 22:1, 17–44.

Rumelhart, D. E., McClelland, J. L., and the PDP Research Group. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations.* Cambridge, MA: MIT Press.

Russo, J. E., and Dosher, B. (1983). Strategies for multiattribute binary choice. *Journal of Experimental Psychology: Learning, Memory, and Cognition,* 9:4, 676–696.

Shaw, M. E. (1981). *Group dynamics: The psychology of small group behavior, 3rd ed.,* New York: McGraw-Hill.

Siegel, J. J., Dubrovsky, V., Kiesler, S., and McGuire, T. W. (1986). Group processes in computer-mediated communication. *Organizational Behavior and Human Decision Processes,* 37:2, 157–187.

Slovic, P., Fischloff, B., and Lichtenstein, S. (1977). Behavioral decision theory. *Annual Review Psychology,* 28, 1–39.

Stillings, N., Feinstein, M. H., Garfield, J. L., Rissland, E. L., Rosenbaum, D. A., Weisler, S. E., and Baker-Ward, L. (1995). *Cognitive science: An introduction* (Second Edition). Cambridge, MA: MIT Press.

Tversky, A., and Kahneman, D. (1974). Judgment under uncertainty: Heuristics and biases, *Science,* 185, 1124–1131.

Public Accounting Firms

Mary S. Doucet and Thomas A. Doucet
California State University, Bakersfield

GLOSSARY

assurance services Independent professional services that improve the quality of information or its context, for decision makers.
attestation services Services where a practitioner, such as a public accountant, is engaged to issue a conclusion about the reliability of a written assertion of another party.
audit The process of objectively obtaining and evaluating evidence regarding assertions about economic actions and events to determine the extent of correspondence between the assertions and established criteria and communicating the results to interested users.
collaborative computing technologies Software technologies which facilitate the cooperative work environment and include such technologies as groupware, workflow technology, and knowledge management software.
computer-assisted audit tools and techniques Computerized tools and techniques that can be used to: (1) retrieve and analyze information about the system and about specific applications within the system and (2) test transactions of specific application systems.
continuous audit A methodology that allows auditors to provide assurance on a subject matter using a series of auditors' reports issued simultaneously with, or a short period after, the occurrence of events underlying the subject matter.
decision support systems Rule based to assist the accountant in making decisions as opposed to making the decision for the accountant.
enterprise resource planning systems Fully integrated software packages that include individual modules which perform functions in the following areas: human resources, accounting, finance, various aspects of manufacturing, marketing, and distribution.
intelligent systems Software tools that enable decision makers to draw on the knowledge and decision processes of experts in making decisions.

PUBLIC ACCOUNTING FIRMS provide a variety of services to their clients. These services are typically categorized as assurance services such as audits, tax services such as tax planning and tax return preparation, and consulting (management advisory) services which encompass a wide range of activities including information systems consulting. Advances in information systems technology have had a significant impact on how public accounting firms provide these services and have created new and expanded service opportunities, thereby increasing the range of services that public accounting firms offer. In particular, the need to keep up with their clients' technology has been a major factor in the increased use of technology by public accounting firms. As their clients have become more advanced in their use of technology, public accounting firms have had to adapt and become more technologically sophisticated in providing their services in order stay competitive.

I. BACKGROUND

Accounting information systems have evolved from simple manual systems to complex computerized systems.

Manual systems take accounting data and convert it into useful information primarily for financial statements. While manual systems can provide other information that is of value to management, the cost of extracting such information from a manual system can be prohibitive. Computerized systems can provide the information needed for financial statements as well as satisfy an organization's other information needs and do so effectively and efficiently. As a result of the many advances in information systems technology, combined with the decreasing cost of the technology, most organizations (business, government, and nonprofit) have made the transition from a manual system to a computerized system.

II. IMPACT OF TECHNOLOGY

Advances in information technology have had two primary impacts on public accounting firms. The first has been on how public accounting firms provide their traditional services such as audit, tax, and consulting services. The second is the new service opportunities available to public accounting firms. In addition, public accounting firms and the accounting profession have welcomed the Internet as a means to enhance the services they provide their clients and the public. Many public accounting firms, state societies of Certified Public Accountants, state boards of public accountancy, and the American Institute of Certified Public Accountants have web sites which provide extensive information for professionals and their clients.

A. How Services Are Provided

To stay competitive public accounting firms must continually improve the quality, effectiveness, and efficiency of the services they provide as well as adapt to their clients' technology. Many public accounting firms have adopted the use of computerized audit working papers to document their work and as a means to improve the quality, effectiveness, and efficiency of the audit services they provide. Other examples of technology that have helped public accounting firms improve their services are collaborative computing technologies, computer-assisted audit tools and techniques, intelligent systems, and decision support systems. A brief description of each of these technologies follows.

1. Collaborative Computing Technologies

Collaborative computing technologies include groupware, workflow technology, and knowledge management software. Groupware is a suite of software tools specifically designed to be utilized by groups to enhance and promote teamwork. Key features of groupware include e-mail, to-do lists, sharable calendars and scheduling systems, and discussion groups. Sharable calendars and scheduling systems allow for more efficient scheduling either a face-to-face meeting, chat room discussion group, or conference call. An example of this type of software is Lotus Notes.

Collaborative workflow software allows each team member to contribute to the final product from a variety of locations. Data may be modified as it flows through the system and team members can change or take action on what others have already done.

Knowledge management software allows for the communication and sharing of information among users who might not be in a workgroup together, but have common needs. The use of knowledge management software, such as Arthur Andersen's KnowledgeSpace, fosters collaboration and learning among disparate groups of people. KnowledgeSpace is a web-based product that gives users access to some of Arthur Andersen's knowledge resources, along with news, insights, and diagnostic tools.

2. Computer-Assisted Audit Tools and Techniques

Computer-assisted audit tools and techniques (CAATTs) include a variety of computerized tools and techniques that can be used to: (1) retrieve and analyze information about the system and about specific applications within the system and (2) test transactions of specific application systems. They also include tools and techniques to test the security of the system and to analyze operating system features, including the analysis of operating systems logs.

Auditors can often use systems-vendor supplied software packages such as access control software and utility program tools to gather evidence about the operations of the system. If these software packages do not provide the auditor with needed information, the auditor may have to write specialized audit programs to gather the evidence needed regarding system operations.

To substantiate the balances in the financial statements the auditors must retrieve and analyze information from application systems. Auditors can use embedded audit modules (EAM) and generalized audit software (GAS) packages to accomplish this. The use of EAMs requires that the auditor be allowed to include an EAM in the client's application program and generally requires that this be done during the application development stage. Then EAMs identify se-

lected transactions as they are processed and extract copies of them in real time. This allows for concurrent auditing to occur where evidence is gathered throughout the period thereby allowing auditors to react on a more timely basis if material errors are detected.

Generalized audit software is software that can be used on a variety of computer platforms to assist auditors in completing routine audit tasks such as performing calculations, selecting and reporting detailed information, selecting statistical samples, printing confirmations, comparing multiple files, and formatting results of tests into reports. It does this by allowing auditors to easily access and manipulate data stored on client computers. It can also increase both the effectiveness and efficiency of an audit by increasing accuracy and greatly reducing the time needed to complete routine audit tasks. There are proprietary GAS packages that have been developed by large public accounting firms and commercially available GAS packages. Two popular commercially available GAS packages are Interactive Data Extraction and Analysis (IDEA) by the Canadian Institute of Chartered Accountants (CICA) and ACL by ACL Limited.

Many CAATTs test application processing to determine whether controls such as edit routines and processing controls are functioning properly. For example, parallel simulation can be used to test the application processing under review. The auditor writes a program to simulate an aspect of the application under review and then processes transactions that were previously processed by the application and compares the results.

Three other methods of testing application processing controls use some form of test data. In each case the auditor creates test data that represent both valid and invalid transactions. The first test data method processes test data using a copy of the current version of the application program under review. The results of this processing are then compared with predetermined results to assess whether the application program and its controls are operating effectively. The second method is the integrated test facility (ITF). With this method test data are processed during the application program's normal operations. The auditor then analyzes the processing of these transactions to assess application program operating effectiveness. In order to use the ITF method, the ITF audit modules have to be created during the development of the application program. The third method is called tracing. This method requires that the application program under review be compiled to activate the tracing option. Test data transactions are then traced through all processing stages of the program. The auditor then reviews the listing of all instructions that were executed during processing.

3. Intelligent Systems

Intelligent systems refers to different software tools that enable decision makers to draw on the knowledge and decision processes of experts in making decisions. Many public accounting firms have implemented intelligent systems of one sort or another to guide auditors and improve efficiencies. The most widely used intelligent system in public accounting firms is expert systems (ES). Research has led to neural networks that have the ability to "learn" patterns as they process data. Auditors have begun to use these systems to monitor large on-line transactions processing systems. However, their use is not as pervasive as the use of ES.

Expert Systems are knowledge-based systems that capture the knowledge of experts and make that knowledge available to other accountants. The goal is that access to this knowledge will lead accountants to make the same decisions that the experts would make. The result is faster, more consistent, and more complete decisions. Another benefit of ES is that it can help train accountants by providing the reasoning used in the decision-making process. There are commercially available ES and some large public accounting firms have developed their own ES. An example of an ES is ExperTAX. ExperTAX was developed by the accounting firm Coopers & Lybrand, which is now part of PricewaterhouseCoopers. ExperTAX is used to facilitate tax accrual for financial reporting purposes.

4. Decision Support Systems

Unlike ES, which are knowledge based and make the decision for the accountant, decision support systems (DSS) are rule based and assist the accountant in making decisions as opposed to making the decision for the accountant. Decision support systems can help improve efficiency, but the main benefit of DSS is in helping accountants to make consistent and effective decisions. Many public accounting firms have developed DSS for a number of different applications.

B. New Service Opportunities

Historically, the majority of attest services provided by public accounting firms were financial statements audits. However, public accounting firms have been

increasingly asked to provide assurance on written representations other than historical financial statements. At first auditors responded to these requests for other attest services by applying the concepts underlying Generally Accepted Auditing Standards (GAAS). However, as the breadth of attest services expanded, it became increasingly difficult to apply GAAS to these and other attest engagements. An attest engagement is defined as "one in which a practitioner [public accountant] is engaged to issue or does issue a written communication that expresses a conclusion about the reliability of a written assertion that is the result of another party." Attestation standards were first published by the American Institute of Certified Public Accountants (AICPA) in 1986 to provide guidance to public accountants in providing these other attestation services. Public accounting firms have built a reputation for independence, objectivity, and integrity in providing attestation services, including the financial statement audit, and this reputation will enhance their ability to be the provider of choice for new assurance services.

The concept of assurance services is a broader range of services that include attestation services. The definition of assurance services does not require a written report, nor does it require that the assurance be provided regarding a written assertion. Therefore, the services that public accounting firms are being asked to provide include this broader category of assurance services, many of which are related to information technology. Advances in information technology have changed and will continue to change how much information is available, how accessible the information is, and how this information is used to conduct business. Examples would be the use of electronic data interchange (EDI), to enhance the flow of information between organizations, electronic funds transfer (EFT), to transfer funds between institutions or accounts within an institution, and the use of e-commerce to facilitate transactions.

Advances such as EDI, EFT, and e-commerce have created opportunities for public accounting firms to expand the services they provide. Users of EDI, EFT, and e-commerce must have assurance that the information used is reliable, accurate, and secure. In response the AICPA and the CICA have established guidelines for two assurance services: SysTrust and WebTrust. The SysTrust assurance service is designed to provide assurance regarding the availability, security, integrity, and maintainability of systems including those used in EDI, EFT, and e-commerce. The WebTrust assurance service is designed to provide assurance related to the privacy, integrity, and protection of information in e-commerce transactions. These are just two examples of the many new assurance service opportunities available to public accounting firms.

C. Impact on Auditing Services

One of the services that is most often associated with public accounting firms is the financial statement audit. Most people have some familiarity with the financial statement audit, even if they might not completely understand its purpose and scope. The purpose of the financial statement audit is to add credibility to the financial statements prepared by organizations. Auditors add credibility by providing an independent opinion on the fair presentation of an organization's financial statements in conformance with generally accepted accounting principles (GAAP).

The audit services provided by public accounting firms are very important to the stakeholders of business, government, and nonprofit organizations. These stakeholders rely on the audit to provide assurance that the information used for investment, lending, resource allocation, and other decisions is reliable. Thus public accounting firms have to stay abreast of the changes in technology if they are to successfully serve these stakeholders' needs and the needs of their clients.

Prior to the use of computerized systems, auditors spent the majority of their time examining documentation to test controls and to substantiate account balances. As their clients began to use more advanced information technology, public accounting firms had to change the way they provided audit services. When clients first began using simple computerized systems, public accountants were able to audit by auditing "around the computer." This meant that auditors would assess the reliability of data that went into the computer and the data that came out of the computer. There was no assessment made of the reliability and integrity of the processing of the data by the computer. This approach is still valid today for relatively simple, and time-tested applications.

To test controls and to substantiate account balances in complex applications auditors have to have extensive understanding of the internal logic of the applications and must use CAATTs to audit "through the computer." To test controls in these more complex applications auditors can use a variety of CAATTs, including the test data method, tracing, or ITF discussed previously. To substantiate account balances auditors can use a variety of CAATTs, including parallel simulation or GAS discussed previously.

Another advance in technology that has had a significant impact on how public accounting firms provide audit services is the increased use of integrated software systems such as enterprise resource planning (ERP) systems. These systems are fully integrated software packages that include individual modules which perform functions in the following areas: human resources, accounting, finance, various aspects of manufacturing, marketing, and distribution. They allow flexibility while maintaining standardized control systems and generally have more audit modules built into the various applications than the typical legacy system or nonintegrated software systems. This allows for the concurrent collection of audit information throughout the period.

Another impact that ERP systems have had on auditing is the startup cost of gaining the expertise regarding the ERP system used by a client and developing and supporting the audit tools to be used in auditing the system. However, this startup cost is probably no more significant, and may even be less, than the startup cost for a client with an in-house developed system. The advantage of the ERP system over the in-house developed system for the auditor is that once the audit team understands the control structure of the ERP and develops the audit tools to be used, these tools can be applied to other clients who use the same ERP system because the set of controls is fairly standard.

D. Impact on Tax Services

While the inherent nature of tax planning and tax return preparation is static, the tax regulations are constantly changing. Public accounting firms use information systems technology to keep abreast of the changing tax regulations, to conduct tax research, to make difficult decisions related to tax issues, and to efficiently file returns.

Access to on-line resources through the Internet has greatly enhanced the ability of public accounting firms to keep up with changing tax regulations as well as conduct appropriate research related to a variety of client issues. Information concerning tax regulations and related court cases is available from several providers via the Internet.

Given the complexity of the tax code and the fact that tax regulations are constantly changing, some public accounting firms use ES to make tax-related decisions. An example of such an ES is ExperTax which was discussed previously.

Information technology has also allowed public accounting firms in the United States, in cooperation with the Internal Revenue Service (IRS), to increase the effectiveness and efficiency of filing tax returns. This is made possible by making tax returns forms available electronically and by allowing the electronic filing of tax returns. This not only saves costs for accounting firms and their clients but also for the IRS. The IRS actively encourages the electronic filing of tax returns.

E. Impact on Consulting Services

A significant impact of the advances in information technology has been in the area of consulting services, which have become an increasingly important source of revenue for public accounting firms. The majority of consulting revenues are related to information technology. Additionally, many organizations have turned to public accounting firms to provide other technology-related services. Some large public accounting firms have also formed alliances with major ERP providers to help organizations acquire and implement these systems.

III. DISCUSSION OF FUTURE IMPACT OF TECHNOLOGY

Advances in information technology have had a significant impact on how public accounting firms provide their services. Advances in information technology have provided and will continue to provide many opportunities for new services. This is especially true for assurance and consulting services. Public accounting firms continue to experience substantial growth in each of these service areas. In the future, advances in technology should continue to reshape how public accounting firms provide services.

Rapid improvements in the reliability of highly automated application processes, the automation of audit procedures, and the ability to report the results of these automated audit procedures to auditors in real time, make the probability of continuous audits a reality. A continuous audit is defined as "a methodology that enables independent auditors to provide written assurance on a subject matter using a series of auditors' reports issued simultaneously with, or a short period after, the occurrence of events underlying the subject matter." These reports could be issued on a periodic basis (daily, weekly), could be continuously available ("evergreen reports"), or could be issued when specifically requested by a user ("reports on demand"). In order to be able to provide continuous au-

dit services, auditors would have to be technically proficient in information technology.

While technology continues to improve the availability, quality, and reliability of accounting information, technology also creates challenges for public accounting firms. In addition to changing how public accounting firms provide services and increasing the different services provided, advances in technology have, of necessity, demanded that the professionals in public accounting firms develop new skills. When the new skills necessary to provide the services are extensive, professionals with different educational backgrounds and expertise must be hired. Because of this the traditional view of public accounting firm professionals has evolved from primarily auditors and tax preparers to multiskilled professionals who provide services far beyond audit and tax.

The increased use of paperless technologies such as EDI, EFT, and e-commerce have also created challenges for public accounting firms. These paperless technologies create challenges in terms of security and confidentiality. In providing consulting services related to these technologies, public accounting firms have to ensure that security and confidentiality controls are designed into the systems. In providing assurance services related to these technologies, public accounting firms have to provide assurance regarding security and confidentiality controls to users of these systems.

In addition, as the information systems of public accounting firms become more distributed and the use of paperless technologies to perform their services increases, the security and confidentiality of client information becomes a particular challenge. As confidentiality is one of the cornerstones of the public accounting profession, this challenge is not to be taken lightly.

As advances in information technology continue, the challenge for public accounting firms will be to adapt to these advances to stay competitive. Public accounting firms have demonstrated the ability to meet this challenge by changing how they provide their services and by identifying new service opportunities that result from the advances in information technology.

ACKNOWLEDGMENTS

We thank Lyn Graham and Bill Powers of BDO Seidman, and Rose Martin and Robert Nehmer for their useful insights.

SEE ALSO THE FOLLOWING ARTICLES

Accounting • Control and Auditing • Decision Support Systems • Enterprise Resource Planning • Management Information Systems

BIBLIOGRAPHY

American Accounting Association (AAA). (1971). Committee on Basic Auditing Concepts. Sarasota, FL.

American Institute of Certified Public Accountants (AICPA). (1997). *Professional Standards, Vol. 1.* New York: American Institute of Certified Public Accountants.

American Institute of Certified Public Accountants (AICPA). Web site at www.aicpa.org/.

Association of Chartered Certified Accountants (ACCA). Web site at www.acca.co.uk/.

Association of Government Accountants (AGA). Web site at www.agacgfm.org/.

Canadian Institute of Chartered Accountants (CICA). (1999). *Continuous Auditing.* Toronto: Canadian Institute of Chartered Accountants.

Canadian Institute of Chartered Accountants (CICA). Web site at www.cica.ca/cica/cicawebsite.nsf/public/homepage.

Information Systems Audit and Control Association & Foundation (ISACA). Web site at www.isaca.org/.

Institute of Management Accountants (IMA). Web site at www.imanet.org/.

International Federation of Accountants (IFAC). Web site at www.ifac.org/index.tmpl.

Public Health

Denis J. Protti
University of Victoria, Canada

THE FIELD OF PUBLIC HEALTH has greatly benefited in the past and will benefit even more in the future from the effective application of the principles of information science and information management and the effective implementation of information technology. Public health practitioners have at times been required to avail themselves of technology and systems designed to meet the requirements of the private sector or the acute care medical sector. Public health information requirements are different and their needs unique. In the traditional clinical setting, the focus is on the single patient; in the public health setting, the focus is on the population.

Numerical information systems developed for patient care or the clinical laboratory is typically oriented towards facilitating the entry and review of a single record or of several hundred records of subjects in a study. By contrast, public health practitioners often need to examine thousands of records, although they may not need detailed information for each individual, only summary information about the population. In addition, holders of data are often eager to share selections of their data with others and to engage in collaborative studies. Textual information systems that describe the experimental medical literature are easily accessed through MEDLARS and software packages such as Grateful Med. By contrast, searching the corresponding public health literature is difficult because government publications at all levels are not listed in Index Medicus, are not centrally stored, and have extremely variable formats and lengths. A further complicating factor is the paucity of public health oriented keywords in the Medical Subject Headings (MeSH) system; hence making the public health literature available in full text searchable form is an important way to provide access to it.

The data analysis needs of the clinically based epidemiologist often differ from those of public health professionals in health departments. While the clinically based epidemiologist collects and analyzes data from chart reviews and clinical trials and needs software that supports nonparametric statistics and time-series analysis, the public health worker collects data from surveillance systems, population-based surveys, and outbreak reports and needs software that can be used to perform standardization, fit mathematical models to disease patterns, analyze data from complex surveys, and draw maps.

However, the most significant difference is perhaps in the area of communications. The clinician needs to communicate with patients, with the clinical laboratory, and with colleagues who are typically close by. The amount of information to share is often small (a status, a recommendation) and urgency is often high; telephones and beepers fit these needs. The public health practitioner on the other hand needs to communicate with colleagues in the state or district laboratory, with federal agencies, and with research collaborators at many geographically separated sites; large groups may be called upon to make decisions. The amount of information to share is often large but urgency is rarely high. Electronic mail and video teleconferencing are often more appropriate technologies that fit these needs.

I. INFORMATION SCIENCE

Before exploring the application of information management principles and the impact of information technology in the field of public health, it is important to first understand the foundations on which information science is built.

A. Information

The most universal definition of information comes from philosophy—information is knowledge for the purposes of taking effective action. In his original treatise on cybernetics, Wiener in 1948 compared the acquisition and processing of information in human beings and animals with the similar activity in the control of machines and other activities. Many have attempted to define information, a few examples being:

1. An increment of knowledge
2. An interpretation of external stimuli
3. Increasing the state of knowledge of a recipient
4. A reduction in uncertainty, following communication
5. Any physical form of representation, or surrogate of knowledge, or of a particular thought, used for communication
6. A measure of one's freedom of choice when one selects a message
7. Recorded experience that is, or can be, used in decision making

The last definition is that of Churchman who postulated in 1971 that recorded experience becomes information only when it is or can be applied to a decision process. Hence it is possible to have access to large amounts of descriptive raw data but yet have little or no information. In an engineering or information theory sense, information is the capacity of a communications channel, a measurable quantity that is independent of the physical medium by which it is conveyed. Applying this theory to Churchman's definition enables the measurement of the amount of information than can be obtained from a particular piece of raw data or descriptive material. In an "information system" sense, information is generally considered to be data (raw material) that have been processed into a form that is meaningful to the recipient and is of real or perceived value in current or prospective decisions. Wiener regarded communication between the component parts of a community as vital to its activities. He saw an information system as the means by which the necessary communication can be established and maintained.

There appears to be no consensual definition of information. Information is a complex concept and simplistic views of it lead to simplistic decisions. In a world of uncertainty, information reduces uncertainty. It changes the probabilities attached to expected outcomes in a decision situation and therefore has value in the decision process. It is so closely related to the concepts of thought, values, knowledge, and environment, that it is often difficult to isolate and adequately define "information." Cybernetics, which is concerned with the use of information to effect certain control actions, is but one of many fields which claims to study information.

B. Information Theory

Wiener in 1948 first suggested information theory. His contention that any organism is held together by the possession of means for acquisition, use, retention, and transmission of information denotes a biological sense of information. Shannon and Weaver in 1960 provided the foundation for measuring information (in a nonsemantic sense). Their concern was not with meaning, or the semantic aspects of information, but with the engineering problems of transmitting it. Information theory, as ascribed to by Shannon and Weaver, is only concerned with the factors that determine whether or not a message has been exactly or approximately transferred between a source and the destination. The principal elements of Shannon and Weaver's communications system can be delineated as follows:

1. Information source—The originator of the messages, which are to be transferred to the destination. There are an almost unlimited variety of permissible message types. Typewriter-like systems use sequences of letters. Bank checks are messages, which are composed of letters and numbers.
2. Transmitter—Operates on the message to transform it into a signal form, which can be transmitted over the communication channel (path).
3. Channel—The communication path over which the signal is transmitted to the receiver.
4. Receiver—Usually performs the inverse function of the transmitter to yield a reconstruction of the message.
5. Destination—The intended termination of the message transfer.

A noise source is included in the model as unwanted signals in one form or another perturb all systems. There is a distinction between noise and distortion. Distortion is caused by a known (even intentional) operation and can be corrected by an inverse operation. Noise is random or unpredictable interference. Shannon and Weaver identified three levels of problems in a communication system, namely:

1. Technical Accuracy. Just how accurately are the message symbols transferred from the message source to the destination?
2. Semantic Accuracy. How accurately is the semantic meaning of the messages transferred from the message source to the destination? These semantic problems are concerned with how closely the destination interprets the knowledge conveyed by the message to the knowledge intended by the sender.
3. Effectiveness. How effectively does the received message control the system in the intended fashion?

The primary motivation for communication within a system is to instruct selected subsystems to take some course of action. Effectiveness is closely related to semantic accuracy, and the two problems cannot always be completely dissociated. In fact, it is not uncommon to discover situations in which it is either entirely impossible or meaningless to separate the three problem levels. No real communication can take place unless the transmitter and the receiver are making use of compatible codes or schemes for symbolic representation of information, for example, a bridge player failing to "catch" and respond to in an appropriate manner his partner's bidding signal.

C. Information Science

Although information science is not a direct descendant of information theory, many of its practitioners do, however, attempt to retain the spirit of information theory by making information the central concept and by providing precise definitions of what it is. In information theory, the concept of information was never meant to express the meaning of a message; Shannon and Weaver, in fact, clearly stated that semantic concepts were quite irrelevant to the problem. Yet to many, information science is interested in the meaningfulness of information and in the usefulness of information to the user. It is a field of study, which investigates how systems, humans, and/or machines retrieve information rather than just receive information. Humans are active rather than passive; they search for information for a specific purpose and do not just wait to process it, should it happen to pass by.

Webster's dictionary defines "information science" as the collection, classification, storage, retrieval, and dissemination of recorded knowledge, treated both as a pure and an applied science. Although Webster's considers "informatics" to be synonymous with information science, the literal translation of the French term "informatique" and the German term "informatik" is the rational scientific treatment, notably by computer, needed to support knowledge and communications in technical, economic, and social domains. The significant difference between the English and European definitions is the latter's inclusion of the computer. It should also be noted that the European definitions of informatics make no explicit claim to being a science.

The field of information science is perhaps best exemplified by Meadow who views it as a study concerned with the:

1. Nature of information and information processes
2. Measurement of information (including its value) and information processes
3. Communication of information between humans and information machines
4. Organization of information and its effect on the design of machines, algorithms, and human perception of information
5. Human behavior with respect to the generation, communication, and use of information
6. Principles of design and measurement of the performance of algorithms for information processing
7. Artificial intelligence applied to information processing

Before discussing the broader concept of information science in public health, a historical review of the term medical informatics is in order.

D. Medical Informatics

Over the past 25 years, many have published their impression opinions as to what constitutes the field of medical informatics. One of the first to use the term was Reichertz, a physician, who in 1973 defined medical informatics as the science of analysis, documentation, steering, control, and synthesis of information processes within the health care delivery system,

especially in the classical environment of hospitals and medical practice.

Over 20 years ago, Shires and Ball confidently wrote, "The year 1975 will be noted as the year in which medical informatics became accepted as a legitimate term used to describe activities involved in assembling, correlating and making effective use of information and decision making in health care delivery." Moehr *et al.* in 1979 also observed that informatics as a science does not fit into the conventional classification of sciences: it neither belongs to the natural sciences—its objects are not phenomena of nature—nor is it a part of mathematics. It is not a human science nor is it one of the classical engineering sciences. In their opinion, informatics deals with investigating the fundamental procedures of information processing and the general methods of the application of such procedures in various application areas.

Another physician, Levy, defined medical informatics in 1977 as the acquisition, analysis, and dissemination of information in health care delivery processes. He concluded that on the grounds of relevance and direct appropriateness to modern medicine, informatics is a proper basic medical science. Van Bemmel expressed a similar view in 1984, writing that medical informatics comprises the theoretical and practical aspects of information processing and communication, based on knowledge and experience derived from processes in medicine and health care. This definition is tempered somewhat by Hannah who reported in 1985 that nurses continue to consider the term medical to be synonymous with the word physician. A relatively new but highly related term is that of nursing informatics. In using the term, Hannah refers to the use of information technologies in relation to any of the functions which are within the purview of nursing and which are carried out by nurses. As evidenced by the above definitions, the term medical informatics on the one hand appears to be confined to the clinical practice of medicine while on the other it encompasses the broader notion of health and health care delivery. The term public health informatics is now surfacing as evidenced by the National Forum "Accessing Useful Information: Challenges In Health Policy And Public Health" held at The New York Academy of Medicine in March 1998.

E. Health Information Science

After an extensive review of the literature and a critical analysis of the aims and objectives of the University of Victoria's new baccalaureate degree program in health information science, Protti in 1982 defined health information science as the study of the nature of information and its processing, application, and impact within a health care system. This definition was not intended to be unique and mutually exclusive of the work of others. It was rather an attempt to broaden the Reichertz domain of hospitals and medical practice to encompass all of health care. Health information science is to information science as health economics is to economics. An economist is one who specializes in the social science concerned chiefly with the description and analysis of the production, distribution, and consumption of goods and services. A health economist, upon familiarizing himself with the institutions, participants, and concepts of health, illness, and disease, analyzes economic phenomena in health care delivery and resource management settings. Similarly, a computer scientist is concerned with the science of properties, representation, construction, and realization of algorithms. A medical computer scientist is concerned with the application of these concepts to medical science and medical practice. To be effective, he or she must have more than a passing acquaintance with concepts of diagnosis and treatment of disease.

An engineer is concerned with the application of science and mathematics by which the properties of matter and the sources of energy in nature are made useful in structures, machines, products, systems, and processes. A biomedical engineer is concerned with the capacity of human beings to survive and function in abnormally stressful environments and with the protective modification of such environments.

In keeping with Meadow's views, a health information scientist or health informatician should therefore be concerned with:

1. The nature of information and information processes in all aspects of health promotion, detection, and delivery of care
2. The measurement of information and information processes
3. The organization of information and its effect on the performance of health practitioners, researchers, planners, and managers
4. The communication of information between patients, health care providers, administrators, evaluators, planners, and legislators
5. The behavior of patients, healthcare providers, administrators, planners, and legislators, in respect to the generation and use of information

Many of health information science's conceptual foundations are borrowed from other fields such as mathematics, economics, psychology, engineering, sociology, and biology. It is a discipline, which is not distinctively different from those in subject content, but different in outlook. Information science in health is concerned with the individual and group behavior of health care personnel in their interaction with information and with the technology which processes information.

II. INFORMATION TECHNOLOGY

What are the major issues surrounding this complex and rapidly changing subject? To what extent will information technology affect the public health profession over the next 20 years? Rather than presume to have definite answers, this section will raise questions, which readers will have to answer for themselves.

The section is structured so as to develop the following premises:

1. Information technology is part of the larger domain of "technology."
2. The impact that informatics will have on public health can be seen by observing the impact technology is having on society.

Although technologies such as hydroponics, genetic engineering, and nuclear fission are important in the overall scheme of things, they will not be discussed in this section.

One must resist the temptation to predict unrealistically. A little more than a 100 years ago, The American Press Association organized a group of 74 leading authors, journalists, industrialists, business leaders, engineers, social critics, lawyers, politicians, religious leaders, and other luminaries of the day to give their forecasts of the world 100 years later. Among the most striking features of the 1893 forecasts is the remarkable paucity of predictions that actually came true. Some of them seem outlandish, completely disconnected from reality—but fervently believed by their authors. Predictions of what the world will be like 30 to 40 years from now are easy; the predictor need not worry about being around to defend his or her views. The nearer to the present the more difficult the task because the political, social, economic, and emotional issues which influence change are much more apparent. This section will attempt to identify the issues which will probably affect public health over the next 5 to 10 years. The extent to which one agrees with someone else's views of the future is very much influenced by one's own view of the past and present. How well the individual will govern depends on how each reader answers the questions, understands the issues, and is challenged or threatened by their implications. One consolation is that all health professionals have to wrestle with the same questions.

A. The Evolution of Information Technology

Information technology is not a new phenomenon. It has been around since the beginning of time. It entails people communicating with each other and recording their thoughts, ideas, and actions for others to read or hear. The broad definition of information technology includes:

- Computers (mainframes to workstations, desktop personal computers, and multimedia)
- Telecommunications (switching systems to faxes)
- Networks (local area and wide area)
- Document reproduction
- Artificial intelligence and speed recognition expert systems

In coming to understand information technology in a modern context, it is important to realize that the electronic computer is only one component in an elaborate and highly differentiated infrastructure. This infrastructure has grown through a succession of generations of computers, each of which represents a major change in technology. During the 8-year span of each computing generation (the first generation started in the late 1940s and the fifth in the early 1980s), revolutionary changes have taken place that correspond to those taking place over some 70 years or more in the aircraft industry. If we were to draw parallels to the rapid and massive advancements, aircraft would be able to go 100 times faster, a $200,000 home would cost $20,000, and color televisions would cost $20.

The definition of generations in terms of electronic device technology captures important aspects of computing technology such as cost and size decreases, power increases, and so on. However, it fails to account for the qualitative changes that have given computing its distinct character in each generation. The change from mechanical to electronic devices made it possible to store programs as data and enabled the use of computers as a general-purpose tool and then the development of programming language compilers. The transistor made reliable operation possible and enables routine electronic data processing and

interactive time-sharing. Integrated circuits reduced costs to the level where computers became commonplace and made possible the personal computer dedicated to the single user.

Each generation represents a revolution in technology with a qualitatively different impact. Each generation subsumes the capabilities of that preceding it, providing much better facilities at a much lower cost and adding new capabilities not possessed by the previous generations. One of the innovative new capabilities has been in the area of knowledge-based systems. The products stemming from breakthroughs in this area are expert systems, which simulate some of the processes of the human mind, knowledge representation and inference, allowing expertise to be encoded for a computer and made widely available. This has generated a new industry based on creating expert systems to make the practical working knowledge of a human expert in a specific subject area such as medicine widely available to those without direct access to the original expert.

B. Technology and Society

Society is experiencing its second major revolution in less than 200 years. The first was the Industrial Revolution of the 19th century, which saw the substitution of mechanical processes for human muscles. It changed the nature of work, though not the size of the workforce, and with it society's view of human values. The spinning jenny may have done the work of 1000 women, but hundreds of thousands were eventually needed in the mills. The automobile may have put the horse out of business, but Henry Ford saw to it that many more mechanics were needed than blacksmiths, many more oil industry personnel than haymakers. Although it had a significant impact on the nature of work, the Industrial Revolution did provide untold opportunities for the individual to hold a job at some level. Even if the job was classified as unskilled labor, the person still had an identity as a breadwinner and could feel a sense of worth from that. If an industrial job was classified as skilled labor, the person had not only the benefits of the unskilled laborer, but in addition a higher job status. As a rule, every major technological advance destroys the civilization that existed at the time of its introduction into everyday life. The steam engine pushed us out of the field, in huge crowds in darkened halls; television returned us to our own darkened living rooms. The compass and chronometer made intercontinental travel possible, the airplane makes it trivial, and advances in communications technology may make it unnecessary.

The second major revolution is the so-called electronic or information revolution in which electronic circuits are being substituted for human mental skips. The electronic revolution is not only replacing the mental processes of the unskilled laborer, it is creating a genuine human value dilemma for technologists, managers, and professionals. Technology is changing everyone's job. What is both exciting and frightening is that the rate of change does not appear to be diminishing. As stated by Kaiser in 1999, "We are at the cusp of a new century, but the alteration we are about to undergo is much more than a change in digits. It is far greater than the incremental steps—in science, art or engineering—that each century has so far brought us. It will be a quantum leap in consciousness, a dramatic step forward. The Internet, the electronic global brain, is behind this revolution. It will bring us to a new consciousness because it will allow us to share all the information we are able to gather from cultures past and present. And this sharing of information, this global conversation, will change the consciousness of the planet."

The fundamental economic activities of our society—agriculture and the multitude of extractive, manufacturing, and service industries—continue, but a new decision-making process increasingly influences them. Vastly more information (on markets, costs, techniques, other options) is being made available to decision-makers because of the information technology now available. This information is being eagerly sought because more informed decisions, be they in politics, operating factories, hospitals, public health agencies, or any organization, are likely to produce better results. The electronic revolution has made robotics a reality. The development and use of intelligent robots, which perform delicate tasks that once could have been done by thinking human beings, is increasingly commonplace in manufacturing sectors of society. Robotics is beginning to be introduced into health care. How long before they become commonplace?

While the industrial age found its symbol in the factory, the symbol of the information age is the computer, which can hold all the information in the Library of Congress in a machine the size of a small refrigerator. Alternatively its proper symbol may be a robot, a machine capable of supplementing age-old manual labor and liberating human beings from the most arduous and repetitive tasks. Perhaps its symbol is the direct broadcast satellite, which can send television programmers directly into homes around the

globe. Telephone companies the world over are joining forces under the banner of the Integrated Services Digital Network (ISDN), which is described as the key to linking all the elements of the information age. ISDN is several things at the same time, but it will allow every home and organization to receive simultaneously voice, computing, and video signals on a telephone line.

A popular way of looking at information technology is in terms of its utility. The most frequently used reasoning to justify purchases of information technology goes as follows: labor expenses are high and getting higher; computer expenses are low and getting lower. It then logically follows that one should always trade an expensive commodity, such as labor, for an inexpensive commodity, such as computers. One of the resulting dilemmas is that value has become less personal and more social or group oriented. In a technological society, the individual has the potential of becoming insulated against ethical and moral decisions as these responsibilities are projected onto society itself. For people whose identities have been imbedded in their jobs, traditional culture provides no guidelines to help them value themselves after they have been more or less excluded from the productive parts of society.

C. The Evolution of the Health Care Industry

The future of the health care industry is not the same in all parts of the world. In many parts of the world, the health care industry is struggling to satisfy the most basic and fundamental of needs. In other parts of the world, the rapid advances in medical science are putting strains on governments to provide the best possible care, given the limited resources available. In the United States and the United Kingdom, the future of the health care industry is quite clear, the future is competition. Competition is not new, nor is it unique to the Americans and the British; competition has always existed in terms of institutional pride, quality of care, staff prestige, and reputation. In the United States and the United Kingdom, competition is being redefined to include price and marketing as important factors and the key to being competitive is how well information is provided and used.

The increasing emphasis on competition has spurred the movement toward "alternate-site" medicine that is the delivery of health care outside the traditional, and costly, hospital setting. Of particular interest is the role technology plays in this new movement. Diagnoses that were once run in the hospital or in the large clinical laboratories are now being performed in doctors' offices in minutes and at fraction of the cost charged by the big automated laboratories. Increasingly, more and more surgical procedures are now performed routinely in outpatient day surgery units and in private surgicenters. Technological advances such as the lithotripper replace complex and costly major surgery, along with its 10- to 12-day hospital stay, with a 1-day procedure which "shatters" rather than removes kidney stones. Medical costs are also being reduced by treatments that can be performed by many patients at home.

The market—which includes not only the drugs used in the treatment but also auxiliary equipment, such as small programmable pumps—now consists primarily of special nutritional products and services (aimed at patients with abnormal digestive systems), kidney dialysis, and continuous intravenous drug administration. Home therapies will almost certainly embrace such intractable disorders such as Alzheimer's disease and many forms of physical rehabilitation. One company markets a home chemotherapy system for cancer patients, many of whom would normally have to receive anticancer drugs in a hospital or physician's office. Patients who are healthy enough to live at home can often use a continuously administered, prepackaged drug or combination of drugs. The drugs are contained in a small plastic pouch that is attached to a catheter. A portable programmable pump delivers the drugs at a slow constant rate. One of the advantages to this approach is that the steady infusion of such drugs often eliminates the side effects, such as nausea, that usually accompany large doses.

The benefits of such procedures, moreover, extend well beyond lower costs. Recovery or remission rates for many patients are dramatically reduced in the familiar and comfortable home environment. A lens implant in the eyes of a 75-year-old person allows her to continue to live independently in the home and community, which is meaningful to her. The quality of life is infinitely "better" than moving to a home for the blind in a nearby town or city. Neonatal intensive care units are allowing life to be continued in hundreds of cases in which death would have been a certainty 25 years ago. Microcomputing technology is providing artificial voices for those who cannot speak, workstations for the sightless, and communication for those paralyzed by stroke. The elderly, handicapped people, and others with high-risk medical conditions can find a new level of security when their hospitals use a computer and the telephone system to guarantee them

almost instant response in an emergency. The list grows with each passing year. What are the implications of these trends on the public health practitioner?

D. Communications Technology

Perhaps the form of modern information technology which will most affect the public health practitioner in the future is communications technology. The advances in telecommunications are matching the speed of those of computing technology. More importantly the cost of this form of technology is now dropping—after years of overpricing. This drop in cost is no more evident than in the explosion of the Internet.

III. THE INTERNET—WHAT IS IT?

> What railroads were to America in the 19th century and superhighway systems were in the 20th, high bandwidth networks are in the 21st century.
>
> Mitchell Kertzmay, CEO
> Powersoft Corporation

Ask for a definition of the Internet and, depending on whom you ask, you will get either a simplistic answer or one that is long, detailed, and mainly incomprehensible. The simplest way to describe the Internet is with one word—communication. The Internet is often called a network of networks. The Internet provides a vehicle for networks of all kinds and individual standalone computers to intertwine to form a global network, which connect people, the world over. Exactly how many people is not easy to determine. In 1994, it was reported that the number of users with access to the Internet was growing at 10% per month; forecasts were that by the turn of the century there may be 1 million networks, 100 million computers, and 1 billion users on the Internet. In 1998, it was reported that the number of Internet users outside the U.S. is growing at an average annual rate of 70% and will surpass users in the U.S. by 2002. At that time, there were currently 60 million Internet users worldwide, of which 68% are in the U.S. and Canada. Worldwide, the total number of Internet users will reach 228 million by 2002.

As of July 1999, 205 countries or territories had at least one connection to the Internet. Thus only 4 new countries joined the Internet in the first 6 months of 1999. This is a diminished Internet spread rate, because there are not many new countries to join. Estimates of the number of people on the Internet seem to range between 50 and 80 million people worldwide, with 3–5 million users in Europe. But all that is available are estimates (just as there are only estimates of how many people watched a particular TV show). There is no precise way to count the number of people on the Internet.

The users are connected to small local area networks (LANs) in their offices where they share files and electronic mail (e-mail). Increasingly, these LANs are being connected to form groups of thousands of computers that are linked across large areas sometimes referred to as wide area networks (WANs). The speed at which one can do things on the Internet is remarkable; not because the Internet is particularly speedy, but because it enables one to travel around the world in seconds. The lure of the Internet is communication and access. People who want to exchange ideas and develop knowledge are increasingly doing it on the Internet (for example, librarians whose job it is to find documents, books, and other materials, now share their catalogues through the Internet).

The system that has grown into the Internet was originally designed by the United States military in 1969 under the name ARPANET (Advanced Research Projects Agency). The first ARPANET configuration involved four computers and was designed to demonstrate the feasibility of building networks using computers dispersed over a wide area. Each computer communication point or node on the Internet is able to pass information on to the next node. Information on the Internet is controlled using a set of data communications standards know as the TCP/IP protocol. Protocols are agreed upon methods of communication used by computers similar to the way people have protocols for communicating. The specifics of TCP are complex (and highly technical) and beyond the scope of this text. Let it be left to say that the TCP/IP protocol is designed to ensure that every piece of information finds the most direct route to its destination. The hundreds of thousands of nodes around the Internet form a web in which information can travel, thus eliminating the need for central communication switches. This means that as long as at least two nodes are in contact, the network will remain operational.

There is no single owner, or even a formal coalition that actually "owns" the Internet. The various subnetworks have owners who recognize that having connections to other networks either enhances their mission or makes their services more desirable. The only group that "runs" the Internet is the Internet Society (ISOC). The ISOC is a professional membership society with more than 150 organizational and 6000 individual members in over 100 countries. It provides leadership in addressing issues that confront the fu-

ture of the Internet and is the organization home for the groups responsible for Internet infrastructure standards, including the Internet Engineering Task Force (IETF) and the Internet Architecture Board (IAB). A common stake in maintaining the viability and global scaling of the Internet binds the Society's individual and organizational members. They comprise the companies, government agencies, and foundations that have created the Internet and its technologies as well as innovative new entrepreneurial organizations contributing to maintain that dynamic. Its Board of Trustees elected by its membership around the world governs the society.

A. Internet Tools

There are many different ways to send and receive information across the Internet. There are also many recent publications that provide in-depth descriptions. Many are available for free around the Internet or can be purchased in the ever-growing number of computer sections in bookstores. The ability to access different tools depends on the type of Internet account and the sophistication of the interfaces users employ to log on (connect) to the Internet.

1. E-Mail and Usenet

Electronic mail (e-mail) is and probably always will be the most common use of the Internet. It allows Internet users to send and receive messages from around the world. Requests for database searches and the result posted to an account can also be done by e-mail. E-mail is also used to join electronic mailing lists (called listservers) on specific topics of interest. E-mail is used to transfer text, program files, spreadsheets, and even photographic images. Messages can be sent and received in hours and often within minutes; it is no wonder that most e-mail users refer to the regular postal service as "snail mail."

E-mail is based on the fundamental concept of store-and-forward technology. The store part refers to a message being added to a storage system by the message's originator. When the recipient is ready the message is forwarded for retrieval. The beauty of this technique is that the recipient does not have to be available when the originator sends the message. This enables the e-mail system to select how the message will move from the place where it is first stored to the place where it is retrieved (forwarded to the user).

It is becoming increasingly easy to find anyone on the Internet—even if one does not know his or her e-mail address. Internet addresses are in two parts, a "domain" name and a user name separated by a @ sign. The domain name (more correctly called a hierarchical name) consists of the name of the machine on which the user has an account, along with the network groups and subgroups leading to that computer, thereby giving that machine a unique identification, which enables the Internet software to determine where to deliver the message. Delivering the message to the addressee is then up to the named computer. The computer's name is chosen locally and is often colorful or thematic. User names can be cryptic. They are often composed of a first initial and last name but can be shortened to a nickname or identifying numbers. All Internet alphanumeric addresses are actually aliases for numeric addresses, such as 134.6.4.187. The alphanumeric addresses are used because, even though they can be hard to interpret, they are easier than the numeric names. Machines on the Internet called name servers handle the translation of alphanumeric names into numeric addresses. To get around the cumbersome Internet address of a person or persons, many mail programs enable users to create aliases. Aliases are particularly helpful when e-mail is sent to a group of people.

Because electronic mail is such an inexpensive form of communication and it is so easy to send copies of messages to long distribution lists, recipients may get much mail, which is of little or no value to them. As a result, new filtering software is being developed to help sort the wanted from the unwanted mail. Users can develop their own filtering rules (such as "if from boss, display immediately") and can modify them at any time. Techniques such as assigning points to messages to indicate their importance is another variation of the same theme to make e-mail communications among groups more effective—to get relevant information to the recipients with less waste of time on the part of both senders and recipients.

2. Usenet

Usenet newsgroups are Internet bulletin boards that are similar to listservers but require the use of software known as a newsreader. There are thousands of Usenet groups and listservers for discussion of medical and health related topics. A number of groups have already put up information relating to community and public health issues. The Institute of Maternal and Child Health policy at the University of Florida has provided information through the Maternal and Child Health Network. The topics include items such as vaccination requirements, injury prevention summaries,

school health information, and child health policy documents. Internet resources for the hearing impaired including newsletters, software, and demographic data are available through a number of sites. As well as providing services for the hearing impaired, the Internet can be used as an enabling technology, especially for individuals who are homebound or live in an institution. Other types of community health information available include breast cancer support and poison control information.

The potential of the Internet for public health cannot be overstated. The WHO says it receives as many as five unconfirmed rumors a week of new infectious-disease outbreaks, by telephone, newspaper, or e-mail. Each rumor is then investigated and a rumor/outbreak list is sent out electronically on a need-to-know basis to relevant personnel at the WHO, its collaborating centers, and other public health authorities. These reports, however, are not intended for public consumption. The WHO will only post news of an outbreak on the EMC's public Web page after confirmation. Because confirmation often requires sending specimens to a laboratory that may be outside the country of origin, the WHO system is notoriously slow at alerting the world at large to outbreaks.

Public heath experts who want their news as soon as possible have learned to rely on a Web site known as ProMED-Mail. Founded by New York State Health Department epidemiologist Jack Woodall, ProMED-Mail (the name represents Program for Monitoring Emerging Diseases) is an Internet, e-mail based system connected by satellite to ground stations and Internet nodes throughout the world. Anyone can subscribe. In the 4 years since it went online, ProMED-Mail has grown from 40 subscribers in 7 countries to 15,000 in 150 countries and is now considered by experts to be an indispensable, although not wholly reliable, medium for transmitting news of outbreaks and connecting health experts to the far corners of the globe (ProMED-Mail, available at http://www.healthnet.org.

It is however not devoid of controversy. The fact that ProMED provides such rapid access is a great strength; yet that same rapidity of access can breed problems with quality control—they are the flip side of the coin in the electronic age. Callisher and his monitors are overworked and under funded, and they cannot fact-check every entry that comes their way. This unavoidable situation has led to criticism that ProMED-Mail disseminates rumors. David Heymann, who directs the WHO's Division of Emerging and Other Communicable Diseases (EMC), calls ProMED a "very valuable service," but adds that the WHO does not participate in their discussions, because the organization is "not in a position to discuss rumors with the general public." The WHO goal, he says, is to get the rumors and check them out.

One increasingly common application of the Internet for public health is Public Health Focus Team (http://pelican.gmpo.gov/pubhealt.html) whose purpose is to facilitate actions to minimize adverse health effects resulting from consumption of seafood harvested from the Gulf of Mexico or from contact with its waters. The Health Canada Web site (http://www.hc-sc.gc.ca/english/promo.htm) is but one of many similar ones around the world designed to assist the public at large to readily access information on healthy living and health promotion.

3. Telnet

Another Internet tool, *Telnet,* provides a means by which users can log in to other computers around the Internet. Through Telnet a person can access other computer sites using their own computer as a terminal. This is particularly useful for accessing medical libraries and other health care database systems that are linked to the Internet.

4. FTP

File Transfer Protocol (FTP) is the method by which a specific computer transfers data or files around the Internet. Files can be simple text—usually known as ASCH files—or more complex data such as graphics or computer programs, known as binary files. The ability to pup down a file to get data or run a program (if the file is executable) is vital for people doing research and development work. The Internet transfers files at a rate of millions of bytes per second, and with the coming of the National Research and Education Network (NERN), that will soon be upgraded to gigabytes (thousands of millions of bytes) per second. FTP can do more than just retrieve files. It can be used to transfer files to remote machines from a given computer. To make it a practical tool, FTP includes commands of listing directories, listing files in directories, changing directories, getting information about what is being done, and setting parameters for how the operations will be done. Many pieces of free software can be obtained from around the Internet via anonymous FTPs, which allows users to log in to FTP sites where they do not have accounts. These anonymous FTP sites together contain millions of files that add up to terabytes of information.

5. WWW

The World Wide Web (WWW) is the newest and perhaps the most powerful Internet service. It provides links to information via hypertext and for those who have the proper type of Internet access, it can bring multimedia Internet to the desktop. Hypertext provides links to other information sources through selected or highlighted words within a text. A person simply chooses the highlighted word to get further facts on the topic of interest. The links can be of data located on the same machine or anywhere else on the Internet. According to the results of a study published in the July 8, 1999, issue of *Nature,* the World Wide Web is estimated to contain approximately 800 million pages of publicly accessible information. As if the Web's immense size were not enough, the Web continues to grow at an exponential rate: tripling in size every 2 years, according to one estimate.

The two basic approaches to searching the Web are search engines and subject directories. Search engines allow the user to enter keywords that are run against a database (most often created automatically, by "spiders" or "robots"). Based on a combination of criteria (established by the user and/or the search engine), the search engine retrieves WWW documents from its database that match the keywords entered by the searcher. It is important to note that the search engine is not searching the Internet "live," as it exists at this very moment. Rather, it is searching a fixed database that has been compiled some time previous to your search. While all search engines are intended to perform the same task, each goes about this task in a different way, which leads to sometimes amazingly different results. Factors that influence results include the size of the database, the frequency of updating, and the search capabilities. Search engines also differ in their search speed, the design of the search interface, the way in which they display results, and the amount of help they offer.

In most cases, search engines are best used to locate a specific piece of information, such as a known document, an image, or a computer program, rather than a general subject. Examples of search engines include:

AltaVista (http://www.altavista.com)
Excite (http://www.excite.com)
FAST (http://www.alltheweb.com)
Google (http://www.google.com)
HotBot (http://www.hotbot.com)
Infoseek (http://infoseek.go.com)
Northern Light (http://www.northernlight.com)

The growth in the number of search engines has led to the creation of "meta" search tools, often referred to as multithreaded search engines. These search engines allow the user to search multiple databases simultaneously, via a single interface. While they do not offer the same level of control over the search interface and search logic, as do individual search engines, most of the multithreaded engines are very fast. Recently, the capabilities of metatools have been improved to include such useful features as the ability to sort results by site, by type of resource, or by domain, the ability to select which search engines to include, and the ability to modify results. These modifications have greatly increased the effectiveness and utility of the metatools. Popular multithreaded search engines include:

Dogpile (http://www.dogpile.com)
Metacrawler (http://www.go2net.com/search.html)
ProFusion (http://www.profusion.com)
SavvySearch (http://www.savvysearch.com)

Subject-specific search engines do not attempt to index the entire Web. Instead, they focus on searching for Web sites or pages within a defined subject area, geographical area, or type of resource. Because these specialized search engines aim for depth of coverage within a single area, rather than breadth of coverage across subjects, they are often able to index documents that are not included even in the largest search engine databases. For this reason, they offer a useful starting point for certain searches.

Subject directories are hierarchically organized indexes of subject categories that allow the Web searcher to browse through lists of Web sites by subject in search of relevant information. They are compiled and maintained by humans and many include a search engine for searching their own database.Subject directory databases tend to be smaller than those of the search engines, which means that result lists tend to be smaller as well. However, there are other differences between search engines and subject directories that can lead to the latter producing more relevant results. For example, while a search engine typically indexes every page of a given Web site, a subject directory is more likely to provide a link only to the site's home page. Furthermore, because their maintenance includes human intervention, subject directories greatly reduce the probability of retrieving results out of context.

Because subject directories are arranged by category and because they usually return links to the top level of a web site rather than to individual pages, they lend themselves best to searching for information about a general subject, rather than for a specific piece of information.

Examples of subject directories include:

LookSmart (http://www.looksmart.com)
Lycos (http://www.lycos.com/)
Magellan (http://magellan.excite.com/)
Open Directory (http://dmoz.org)
Yahoo (http://www.yahoo.com)

Due to the Web's immense size and constant transformation, keeping up with important sites in all subject areas is humanly impossible. Therefore, a guide compiled by a subject specialist to important resources in his or her area of expertise is more likely than a general subject directory to produce relevant information and is usually more comprehensive than a general guide. Such guides exist for virtually every topic. For example, Voice of the Shuttle (http://vos.ucsb.edu) provides an excellent starting point for humanities research. Just as multithreaded search engines attempt to provide simultaneous access to a number of different search engines, some Web sites act as collections or clearinghouses of specialized subject directories. Many of these sites offer reviews and annotations of the subject directories included and most work on the principle of allowing subject experts to maintain the individual subject directories. Some clearinghouses maintain the specialized guides on their own Web site while others link to guides located at various remote sites.

Examples of clearinghouses include:

Argus Clearinghouse (http://www.clearinghouse.net)
About.com (http://about.com)
WWW Virtual Library (http://www.vlib.org)

B. Hooking Up to the Internet

Not all Internet connections are the same; some allow users only to access certain types of Internet tools and are available in host-based dial up interfaces. Other, usually commercial, Internet connections can enable users to employ Windows or Macintosh based software to access the Internet. Although these commercial connections can be more expensive than other options, the ease of use that these interfaces offer make them an attractive option, especially for first time users. The three most common methods of accessing the Internet are through a university affiliation, via a community access bulletin board known as Freenet, or through a commercial service provider.

1. Universities

Universities were the first large-scale users of the Internet; thus virtually all schools within a typical university have some form of Internet connectivity. Students, faculty, and groups with university affiliations may be eligible for some level of Internet connectivity through their institution. Most universities will give those who are eligible an account on a computer, which has a connection to the Internet. Dialing in to another machine at the university generally accesses university Internet accounts over a modem on a personal computer either at home or on campus. University accounts can be a good place to start to navigate the Internet, especially if they are available for free as part of a university affiliation—most universities do not expect their users to pay for the Internet access. However, difficulties mastering unfriendly interfaces and lack of user support have been known to cause problems for university-based users.

2. Freenets

Freenets are community based electronic bulletin boards that allow users Internet access. Freenets are relatively new tools, but more are coming online every month. The Freenet system is menu driven and is set up with the novice user in mind. They are as the name implies free to use; however, donations are strongly encouraged to help offset operating costs. As well as being able to attain local information, including health care resources, registered users can access e-mail and other Internet tools. Freenets are not intended for business or commercial ventures and users are limited in the scope of tools they can use. For example, the use of FTP and Telnet are limited on most Freenets.

3. Commercial Internet Accounts

Commercial service prices can vary greatly depending on the provider and the type of services used. The four most common types of Internet access that commercial providers supply are:

- Dial up host access—Similar to a university account with access through a text based computing environment; however, interfaces and user support may be better.
- Dial up SLIP and PPP access—Serial Line Internet protocol (SLIP) and point to point protocol (PPP) allow full Internet access over a modem

and telephone line, thus users can employ interfaces that reside directly on their own computer. This is especially useful for neophytes because it means they can make use of Windows and Macintosh graphical interfaces, many of which are free to access Internet tools. SLIP and PPP also can be used with multimedia Internet through the WWW and browsers such Mosaic.
- Dedicated SLIP and PPP access—In this case a dedicated SLIP and PPP account is open for use 24 hours per day.
- Dedicated link—Used to connect an entire LAN to the Internet and/or be connected to a computer(s), which will act as Internet information servers. These links can be quite expensive and are usually only feasible at an organizational level.

C. Conclusion

The Internet was originally developed so that science and research could share resources. To a great extent, communications in the form of e-mail and discussion groups have overshadowed the Internet use for resource sharing. Although the traditional methods of scholarly communication—presentations at conferences, publishing of papers in journals, and so on—have not been eliminated, they are being recognized as inadequate for current research needs. The Internet distributes information in a way that is infinitely more flexible and timelier. Findings, papers, and information can be instantly shared and discussed.

The Internet can provide an innovative solution for meeting a variety of communication needs within the public health communications. As services and tools expand and improve, more ways of applying the technology will continue to be found; the Internet will probably become a service from the telephone company before too long. It is, however, important to remember that the Internet is not about technology, it is about people. The tools and applications are only as valuable as the people they enable and empower to communicate.

There are a varied and growing number of computer-based group support systems, including computer conferencing, video and audio teleconferencing, document interchange services, meeting support tools, and group decision support. Perhaps the most common group support system aimed at increasing the effectiveness of communication among individuals is computer conferencing.

1. Computer Conferencing

Computer conferencing is a teleconference that uses computers, software, and communications networks to allow groups of people to exchange ideas, opinions, and information. The people in a teleconference may all be located in the same building, or they can be scattered worldwide. Each user signs on to the teleconference (via a terminal or personal computer) at his or her own convenience; there is no need for members of the group to be using it simultaneously, although they may do so if they choose.

Although similar in some ways, computer conferencing systems are different from other types of computerized communication, such as electronic mail, bulletin boards, and information retrieval services. E-mail is essentially a one-to-one (or one-to-many) form of communication. Moreover, after a message has been read, it is generally deleted; there tends to be little or no storage of messages. Bulletin boards provide storage but are designed mainly for posting notices for other people to read, a one-to-many type of communication. Information retrieval services provide a stored database that users can retrieve from but cannot change.

Computer conferencing systems typically include not only electronic mail and bulletin boards but also many-to-many communications, by allowing all participants to join topics and enter comments on the subject being discussed. They provide storage; comments are not deleted after being read. In joining one or more conferences, each time the person logs on, the system tells them how many messages have been entered in those particular conferences since the last time they were logged in and will deliver those messages, one at a time. If one gets tired of a conference, one can leave it and not get any more messages from it. Computer conferencing allows the setting up of subconferences for discussing some aspect of the general subject in more detail. Some systems also allow voting, to indicate consensus of opinion.

The benefits of computer conferencing include:

- Fast exchange of information
- Less formality
- Encourages more stimulating ideas
- Provides a written record of discussions
- Convenient; use it at anytime
- Noninterruptive
- Avoids telephone tag and slowness of mail
- Handles a dispersed group as easily as a local one
- Branching allows for special interest discussions and limits junk mail

- Late joiners can catch up easily
- Users can settle matters without face-to-face meetings
- Users like its collaborative nature
- Supports group interaction; valuable for project management
- Encourages chance meetings of people with shared interests
- Fosters cross-fertilization of ideas
- Managers can participate and be more proactive
- Allows large numbers of people to interact as equals.

2. Video Conferencing

Video conferencing technology is becoming an affordable reality that can substantially increase communications productivity. The products being developed to support "personal conferencing" exploit the power and availability of the workstation and the capabilities of interfaces such as Microsoft Windows, presentation Manager, and X Windows, and Motif. They allow users on either end to share moving video images, voice communications, documents, and even applications across the "new" digital ISDN phone service or over common high-speed LANs and WANs such as Ethernet or token ring. They now even work over public data networks such as the Internet. In New Zealand 5 years ago it was estimated that there were around 30–40 users of video conferencing. These days it is estimated around 300 Kiwi organizations dial into video conferencing with systems that range from around $17,500 to $70,000. Payback time has become more attractive as managers identify savings in travel time, which can be put into other business areas.

The multimedia revolution is making the presence of speakers and microphones on workstations commonplace. The new video cameras, the size of a stack of 10 silver dollars, are less intimidating than camcorders. The picture that these cameras produce and the sound quality from the audio equipment are surprisingly good. The price of these systems ranges from $500 to $20,000, depending on the features and the platform supported.

Video conferencing still has network problems and, until 1993, when desktop units came along, conference room systems went for upwards of $60,000; rollabouts started at $25,000. A research report estimated that there were just 14,000 desktop units installed worldwide at the end of 1993, but it expects that the number will soar to 1.96 million by the end of 1998. Even in view of AT&T's arrangement with Intel to create a personal conferencing gateway, universal video calling does not seem likely for a few years. However, innovative use of the technology is giving early users a competitive edge. As an example, most banks have television cameras at their automated teller machines for security purposes. Some are turning that camera into a two-way videoconference application and staffing the bank remotely. They let customers interact with bank personnel and do all of their banking around the clock at considerable less expense than keeping branch offices open. MEDITrust pharmacy (http://www.meditrust.com/), a Canadian mail-order pharmaceutical firm, has created a virtual pharmacy. It puts a video/phone/data kiosk in a convenience store, connects it to a pharmacist at a remote site over ISDN lines, and sends medicines by mail. The kiosk scans the prescription so that the pharmacist can give advice, stamps the prescription filled, and processes the credit card order. Medicine is sent to a customer's home by two-day special mail service. The convenience store may be in a rural village too small to support a full pharmacy. Employers are considering offering their staff a similar service from the company offices.

3. Group Decision Support Systems

Up until recently most of the work in decision support systems has been to help individuals make decisions. However, increasingly in all sectors, and particularly so in public health, decisions are not made by individuals, instead groups of people are involved. Rather than support only communication between members of a group, group decision support systems (GDSS) have features and functions that help these groups form a consensus or come to a decision.

The desired design of a GDSS typically includes several features or characteristics. Each participant is able to work independently of the others, and then publicly release or demonstrate his or her personal work. When personal work is released, all group members are able to retrieve and view it. Similarly, each member is able to retrieve and view work performed by the group as a whole.

Elements of a GDSS include a database, a GDSS model base, specialized application packages, a good user interface, plus the "people" component. The people include not only the participants but also a group facilitator who is responsible for the smooth operation of the group and who may also serve as the operator of the computerized system. Additional features typically include numerical and graphical summarization of ideas and votes, programs for specialized group procedures (such as calculating weights of different alternatives), anonymous recording of ideas,

formal selection of a leader, handling progressive rounds of voting, and eliminating redundant input.

Undoubtedly the supporting of communications and decision making will merge as researchers and developers from both areas begin supporting both. For example, rooms for video conferencing focus on communication support and seldom have workstations to support consensus building. Likewise, group decision rooms with workstations for each participant seldom have video conferencing capability to support dispersed groups. As the tools and technology improve, it is expected that group support will move beyond problems or decisions that are familiar to the participants to sensitive decisions, crisis decisions, confrontation decisions, and even "regular" day-to-day decisions.

D. Data Capturing Technology

When the first portable computers (now called notebooks or laptops) arrived on the scene 10 years ago, these 15-kg units were considered saviors for users that needed to take their computers with them. As size began to decrease and the portable computer's power began to increase, more and more people began to enjoy the benefits of computing on the go. The notebook is now full-featured enough to function as the main computer for many users—all in a package that can weigh approximately 2 kg. Future notebook users can expect to see more features added with no gain in size or weight.

First lighter materials will be developed that will keep the weight down and more functionality will be integrated into the motherboards, so that additional add-ons are not required. The monochrome screen has disappeared as production yields increase and prices decrease on active-matrix color LCD displays. Second, while nickel-metal hydride batteries are replacing traditional nickel-cadmium cells, lithium batteries are just emerging and are offering longer life and less weight. One of the most important developments in notebooks is the advent of the PCMCIA (personal Computer Memory Card International Association) card. PCMCIA presents a new paradigm of computing that extends the portability of notebook computers.

These credit-card-sized devices offer plug-and-play convenience. They can also be inserted and removed without turning off the computer. PCMCIA cards are available for a variety of functions, including network cards, memory expansion, storage fax/modems, sound cards, etc. PCMCIA cards have become a regular feature on desktop machines as well. One future scenario has users simply moving their data and applications back and forth between machines on a PCMCIA. Another popular option is the notebook docking station combination. The docking station will become more of a common accessory as users take the components out of their notebooks and plug them into the station at the office or home to obtain the benefits of a larger monitor, external keyboard, alternate pointing devices, better sound, or link onto a network. One aspect of notebook computing is the palmtop computer, which is an even smaller device capable of data connection in the field. Nurses are already using this form of technology, as well as physiotherapists, chiropodists, and health visitors in the North West Anglia HealTheare Trust in Peterborough, England. Users, once they adjust their work patterns and behaviors, report that the palmtop leads to a more professional and business-like approach to the job, enabling one to rationalize the workload and manage one's time more efficiently, thereby offering a higher standard of patient care than was previously possible.

Finally, voice recognition is not far away. This technology has been used for sometime in the field of radiology and is now beginning to appear in other aspects of medicine. Computer systems are now being delivered with speech boards. For dictation systems to run effectively a multitasking system is required that lets several programs run at the same time. One of the programs just sits and listens to what one is saying, looking for either command phrases or phrases to dictate. Physicians testing these new systems find the system accurate and capable of supporting dictation at 60 to 70 words per minute. They are finding that it saves time and provides more control over clinical notes, bypassing the normal transcription process.

E. Data Storage and Retrieval Technology

The power of information technology rests in its ability to process instructions very quickly. One aspect of this quickness relates to a computer's ability to quickly store and retrieve data. Primary storage is part of the computer's central processing unit (CPU) and is generally referred to as memory. Secondary storage, on the other hand, is physically separated from the CPU. A type of secondary storage that is becoming increasingly common and affordable is optical disks.

One type of optical disk shares the same technology as the digital compact disk (CD) players used with stereo systems and is referred to as CD-ROM, the ROM

referring to read-only memory. Originally, data could be written on them only one time; however, there are now erasable versions. The primary advantage of optical disks is large storage capacity at low costs; some of them cost less than 10 dollars and hold 200 to 2000 megabytes (one gigabyte) of data. Five hundred megabytes is the equivalent of 300,000 double-spaced typewritten pages, or the entire Encyclopedia Britannia several times over.

Optical disks are used for storing large volumes of data, including photographs that are not changed often. The Medline CD-ROM available in most medical libraries is a well known medical example. A number of medical specialties receive their journal references on CD-ROMs and access the material at home on their personal computers. SAM-CD is the CD-ROM version of Scientific American's reference book on Internal Medicine. Scientific American sends out a CD-ROM every 3 months with the new data (including photos) in place and ready to use. The Compendium of Pharmaceuticals and Specialties (CPS) is available on CD-ROM, which offers users the power of the computer in conducting complex searches in a matter of seconds. New and more flexible software is being developed, as CD-ROM technology will soon become a standard component of all personal computers, much as hard disks became standard in the late 1980s. CD-ROM drives are, in the world of high-speed technology, considered slow; the average access time is one second and the average transfer rate is only 300,000 bits of data per second. The newer WORM (write-only read-many) technology is being packaged in multifunction drives or "jukeboxes" and has access times as fast as 45 ms. Still, this is also "too slow." Not far away is holography, a technique for recording and reproducing a complete image of a three-dimensional object, the next great technology in data storage. In August 1994, researchers at Stanford University reported the first digital holographic storage system. The team believes 120 billion bytes can be stored per cubic centimeter using digital holographic storage and access rates will be significantly faster than today's technology.

1. Information Management

Given the "information revolution" our society is experiencing, it is not uncommon to assume that "information" infers only the involvement of computers and communication technology. In organizational settings, one often further assumes that the major issue involved is the introduction of information technology within the organization. What is often overlooked is that the introduction of information technology in an organization is much more of a social process than it is a technical one. If the people involved in information management are to coordinate the acquisition and provision of information effectively, they must understand how people process information, both as individuals and as members of organized groups or units. The real challenges in implementing successful information systems are those of managing people and their perceptions.

2. Information Systems

An information system connects, classifies, processes, and stores data and retrieves, distributes, and communicates data to decision-makers. This processed data may or may not then be transformed into information by the human decision-maker. In an organizational setting such systems are often called management information systems. This view was promulgated by Davis in 1983 when he defined a management information system as an integrated, man machine system for providing information to support the operations, management, and decision-making functions in any organization. The system uses computer hardware, software, manual procedures, management and decision models, and a database. In many ways, information systems are an extension of the study of organizations, organizational systems, organizational behavior, organizational functions, and management. An organization is an administrative and functional structure of human resources, material, and natural and information resources coordinated in some manner to achieve a purpose.

In the 1990s, the once traditional organization was quickly replaced by the "virtual corporation." Whether real or virtual, any organization is held together by the methodologies of acquiring, processing, retaining, transmitting, and utilizing information. The purpose of an information system is to support managerial activities of all types at all levels of an organization. An organizationally based information system acquires, processes, stores, and transmits raw material which is usually a mixture of (a) factual data, (b) material that has been subjected to interpretation in its passage through the system, and (c) other content that is openly acknowledged to be the opinions, judgments, and observations of individuals both within the organization and outside it. The value of this material, that is information, depends upon the use to which it can be put. Measuring information, decision making, and productivity in information processing is

an unresolved problem. Information is an essential commodity and a unique resource. It is often not depreciable and a "purchaser" may not be able to determine the value of an information item without examining it. Information is not a "free good." It is a resource no less essential to the survival of an organization than are personnel, material, and natural resources. Information is a resource that must be conserved, recycled, and protected. As with any other resource it must be managed.

Increasingly, organizations are coming to accept this premise and hence look for people who view information management from an "information science" versus a "computer science" perspective. The two perspectives are related but by no means the same. People with computer science backgrounds tend to be more concerned with computer hardware and software. Their formal education had a strong theoretical and mathematical basis, with particular emphasis in algorithm development. They probably have a thorough grounding in the study of the implementation of algorithms in programming languages which operate on data structures in the environment of hardware. They usually have had little exposure to information requirement analysis and organizational considerations. They have greater expertise in programming, system software, and hardware. People with such a technical background tend to be more machine and technology focused.

People with an information science background or orientation tend to be more concerned with people and the nature of information and information processes in the organization. They are more likely to assess the value of information and its effect on the performance of the decision-makers within the organization. In a health care setting, they are more likely to be aware of how and why information is communicated between patients, clients, health care providers, epidemiologists, administrators, evaluators, and planners. The use to which these people put information is, in the end, the most critical criteria of success of information systems, be they computer based or not.

3. Managing Information

If information is to be managed, someone has to be the information manager. The future will place new demands on information systems in public health environs. Information exchange between health care facilities, governments, and other constituencies is becoming more prevalent, and the need for individuals within an organization to share and use the same information is becoming much more common. In this new climate, new professions are emerging—those of health information managers. These individuals will become active in planning, designing, implementing, managing, developing, and deploying information systems to meet the needs of rapidly changing health care systems. These information systems will vary in complexity from simple central registers to hospital and/or community data abstracting and on to complex interinstitutional networked decision support systems. In health care settings information is needed to support decisions that relate to:

1. Promoting wellness, preventing illness, and curing or ameliorating disease
2. Monitoring, evaluating, controlling, and planning health care resources
3. Formulating health and social services policy
4. Advancing knowledge through research and disseminating knowledge through education

An information manager is any individual within an organization who has been given the responsibility to manage the organization's information. Given the information revolution that today's society is experiencing, it is not uncommon to assume that information infers the involvement of computers and communication technology. In organizational settings, one often further assumes that the major issue involved is the introduction of information technology—within the organization. What is sometimes overlooked is that the introduction of information technology is much more a social process than it is a technical one. If information managers are to coordinate the acquisition and provision of information effectively, they must understand how people process information both as individuals and as members of organized groups or units. They will need to have excellent interpersonal skips in order to teach, motivate, convince, and influence a variety of people. The real challenges in implementing successful information systems are those of managing people and their perceptions. Information managers will be change agents, a bridge between older systems and models, and newer technologies and techniques. No matter where they are positioned in the organization, they are usually expected to develop planning processes for aligning all information systems with the strategic direction, objectives, and structure of the organization. This entails coordinating all information systems within the organization including computing services, minicomputers, microcomputers, records rooms, office automation, management engineering, voice

communication, and other related areas. Determining the investment to be made in information systems and providing a rigorous and disciplined framework for evaluating information benefits versus information costs is also a part of the job. Specific standards and guidelines need to be established for the definition, measurement, use, and disposition of information so that all segments within the organization are operating within the same framework. It is often left to the information manager to explain information technology and the need for new systems to staff at all levels of the organization. This critical educational role is often carried out in conjunction with the development of policies and procedures that ensure the coordination and justification of a request for personal computers, terminals, office automation devices, and various software packages.

These responsibilities can often be onerous, and those who succeed possess excellent interpersonal written and verbal communication skills (that is, an ability to function effectively at the Board, senior, and middle management and operational levels of the health care facility). Effective information managers understand the organization's mission and the business that it is in. They also understand the complexity and dynamics of health care delivery, are able to function in multidisciplinary teams and environments, appreciate small "p" and big "P" politics, and are able to assess political situations. To be effective information managers, people have to be doers. They have to be able to demonstrate short-term success while making progress on the long-range information systems requirements. To do so they must be able to understand the present and future capabilities of information technology, be technologically credible to their peers and staff, and be able to plan the effective use of information technology in the organization.

Information systems are people and information systems create change. Information managers must be able to manage change, which includes a sincere appreciation of the effects of change on people. They must be willing and able to teach and educate a wide variety of individuals at all levels of the organization, none of which can be done without having a positive attitude towards users. Effective information managers demonstrate leadership through effective listening, team building, and consensus building. They are creative, innovative, and have a vision of the future. Most of all, they have an honest concern for the organization's most critical resource—its people.

The health of a nation depends to a certain extent on how well organizations use the resources available to them to promote wellness, prevent illness, and cure disease. The health of an organization depends to a large extent on the effectiveness of the decisions made by its staff; effective decisions require effective managers and information systems, which produce reliable and useful information. The health of an information system is a function of how well it has been defined, designed, implemented, operated, and maintained. Keeping the organization's information systems healthy is the role of the information manager regardless of his or her title.

F. Organizational Transformation

In 1991, Scott Morton and his colleagues at the Massachusetts Institute of Technology's Sloan School of Management Research published a text entitled, *The corporation of the '90s*. The work was as a 5-year, multimillion-dollar research program on how organizations can make better use of information technology. A consortium of MIT faculty and 12 corporate and public sector sponsors contributed financial resources, advice, and their workplaces as experimental sites. The group's focus was about how new technologies are changing the way people work and the way organizations will collaborate and compete. The major findings have had a significant impact on a multitude of organizations in both the private and public sectors around the world, including the United Kingdom's National Health Service. The MIT findings are summarized under the following six headings:

Information technology is enabling fundamental changes in the way work is done. The degree to which a person is affected is determined by how much their work is based on information; that is, information on what product to make or service to deliver and how to do it (production task), as well as when to do it and in conjunction with whom (coordination task). The impact on production work is apparent in:

1. Physical production—Affected by robotic, process control, intelligent sensors
2. Information production—Affected by data processing computers for clerical tasks such as billings
3. Knowledge production—Affected by CAD (computer assisted design)/CAM (computer assisted manufacturing)
4. Workstations for building qualitative products such as a loan

What is less well known is that the new information technology is permitting a change in the economics and functionality of the coordinating process as distance can be shrunk to zero as far as information flow

is concerned. Time can shrink to zero or shift to a more convenient point. Organizational memory, as exemplified by the common database, can be updated by anyone and made available to all authorized users. New "group work" and team concepts combine all three aspects of coordination: distance, time, and memory. The increasing availability of information technology can fundamentally change management work as relevant and timely information on changes in the external environment and the organization's view of the environment affects the direction dimension. Relevant and timely information on measuring the organization's performance against critical success factors affects the control dimensions. The second aspect of control is interpreting such measures against the corporate plan and determining what actions to take.

Information technology is enabling the integration of business functions within and between organizations. Public and private telecommunication networks are making the principle of "any information, at anytime, anywhere, and anyway you want to look at it" economically feasible. The boundaries of organizations are becoming more permeable; where works gets done, when, and with whom is changing. The electronic integration is showing up in the following forms:

1. Within the value chain—Land area networks permit "teams" to work together on a common product
2. End-to-end links of value chains between organizations—EDI (electronic data interchange) and JIT (just in time) systems are shifting the boundaries of an organization to include elements of other organizations thereby creating a virtual organization
3. Value chain substitution via subcontract or alliance—Permit an organization to take advantage (mutual) of economics of scale and unique skips of its partner organization

Electronic integration is removing unproductive buffers and leveraging expertise.

Information technology is causing shifts in the competitive climate of many industries. Information technology is introducing unprecedented degrees of simultaneous competition and collaboration between firms. It is becoming increasingly important to know when to support standards and when to try to preempt competitors by establishing a proprietary defacto standard. The benefits do not flow from the mere use of information technology but arise from the human, organizational, and system innovations that are added on to the original business benefit. Information technology is merely an enabler that offers an organization the opportunity to invest vigorously in added innovations if it wishes to stay ahead of its competitors.

Information technology presents new strategic opportunities for organizations that reassess their mission and objectives. Organizations are going to go through three distinctive stages as they attempt to respond to their changing environments:

1. Automate—Reduce the cost of production, usually by reducing the number of workers. As an example, scanners, bar code, and universal product codes are being introduced for more than identifying goods.
2. lnformate—What happens when automated processes yield information as a byproduct. This necessitates that knowledge workers develop new skills to work with new information tools; it often entails new ways of thinking.
3. Transform—A stage characterized by leadership, vision, and a sustained process of organization empowerment. It includes the broad view of quality but goes beyond this to address the unique opportunities presented by the environment and enabled by information technology.

Production workers will become analyzers, a role offering a different level of conceptual skill from what was needed before as a doer or machine minder; it will require an ability to see patterns and understand the overall process rather than just looking at controlling information on a screen.

Successful application of information technology will require changes in management and organizational structure. Information technology is enabling a break up or disintegration of traditional organizational forms and multiple skips can be brought together at an arbitrary point in time and location. The ability of information technology to affect coordination by shrinking time and distance permits an organization to respond more quickly and accurately to the marketplace. This not only reduces assets the organization has tied up but improves quality as seen by the customer. The "metabolic" rate of the organization, that is, the rate at which information flows and decisions are made, is speeding up and will get faster in the next millennium. The measurements, the rewards, the incentives, and the required skills all require rethinking in the new information-technology-impacted world.

A major challenge for management in the next millennium will be to lead their organizations through the transformation necessary to prosper in the globally competitive

environment. Management must ensure that the forces influencing change move through time to accomplish the organization's objectives. Evidence to date is that at the aggregate level, information technology has not improved profitability or productivity. Some of the reasons are:

- Benefits are there but simply not visible
- Improvement is in lower prices or better quality
- Investment in information technology is necessary to stay in business
- The external world is demanding more
- Use of information technology in low pay off areas
- Information technology is laid on top of existing services
- No cost reduction, just cost replacement

To go successfully through the transformation process, organizations must have a clear business purpose and a vision of what the organization is to become; a large amount of time and effort must be invested to enable the organization to understand where it is going and why. The organization must have a robust information technology infrastructure in place, including electronic networks and understood standards; the organization must invest heavily and early enough in human resources—all employees must have a sense of empowerment. Last, but by no means least, understanding one's organizational culture and knowing what it means to have an innovative culture is the first key step in the move toward an adaptive organization.

1. Case Study: The National Health Service Information Management and Technology (NHS IM&T) Strategy

One public service organization which has adopted the MIT findings as a cornerstone to its corporate strategy is the United Kingdom's National Health Service (NHS). The goal of the NHS Management Executive is to create a better health service for the nation in three ways:

1. Ensuring services are of the highest quality and responsive to the needs and wishes of patients
2. Ensuring that health services are effectively targeted so as to improve the health of local populations
3. Improving the efficiency of the services so that as great a volume of well-targeted effective services as possible is provided from the available resources

The July 1992 White paper, the Health of the Nation, identified five priority areas and establishes key targets such as:

1. Reducing deaths from coronary heart disease in the under-65 age group by at least 40% by the year 2000
2. Reducing cervical cancer by at least 20% by the year 2000
3. Reducing suicides by at least 15% by the year 2000
4. Reducing gonorrhea by at least 20% by 1995
5. Reducing deaths from accidents among children under 15 by at least 33% by 2005

A strengthened information and research capability at central and regional levels is an essential component of the business plan. Expanded or new health surveys and epidemiological overviews to improve baseline statistics on the health of the population will be undertaken. A Central Health Outcomes Unit will lead on developing and coordinating work on assessment of health outcomes. Information systems, which enable adequate monitoring and review, will be developed including a public Health Information Strategy. Such was the case in 1992 under a Conservative government. In May 1997, a new Labor government was elected and the policies changed. One of their first efforts was the White paper, "Savings Lives: Our Healthier Nation" (http://www.doh.gov.uk/ohn/execsum.htm). In it they reject the previous Government's scattergun targets. Instead they set tougher (their term) but attainable targets in priority areas by the year 2010:

- Cancer—To reduce the death rate in people under 75 by at least a fifth
- Coronary heart disease and Stroke—To reduce the death rate in people under 75 by at least two fifths
- Accidents—To reduce the death rate by at least a fifth and serious injury by at least a tenth
- Mental illness—To reduce the death rates from suicide and undetermined injury by at least a fifth

IV. THE INFORMATION MANAGEMENT AND TECHNOLOGY (IM&T) STRATEGY

In 1992, any public health information strategy was to be a part of the NHS IM&T Strategy, which was to respond to the business needs of the NHS to see best benefit and value for money from IM&T investment.

It was to set the direction for computerization and information sharing across the NHS into the next century. The strategy was intended to ensure that the implementation of information systems in the NHS was coordinated and managed to achieve maximum potential benefits for patients, clinical staff, management, and administrative staff.

The strategy was intended to support better care and communication through the appropriate use of IM&T. It was to provide a framework for the connection and exchange of data. Whether or not it achieved these goals is outside the purview of this paper. In the opinion of some it failed while others hold the view that critical infrastructure elements were indeed put into place.

In September 1998, the Labor government released it's own Information Strategy entitled "Information for Health: An Information Strategy for the Modern NHS 1998–2005." The purpose of this information strategy is to ensure information is used to help patients receive the best possible care. The strategy will enable NHS professionals to have the information they need both to provide that care and to play their part in improving the public's health. The strategy also aims to ensure that patients, caregivers, and the public have the information necessary to make decisions about their own treatment and care and to influence the shape of health services generally.

The Government has set out the following specific objectives to be delivered through the implementation of this strategy over the period 1998–2005:

- Ensure that patients can be confident that the NHS professionals caring for them have reliable and rapid access, 24 hours a day, to the relevant personal information necessary to support their care
- Eliminate unnecessary travel and delay for patients by providing remote online access to services, specialists, and care, wherever practicable
- Provide access for NHS patients to accredited, independent, multimedia background information and advice about their condition
- Provide every NHS professional with online access to the latest local guidance and national evidence on treatment, and the information they need to evaluate the effectiveness of their work and to support their professional development
- Ensure the availability of accurate information for managers and planners to support local Health Improvement Programs and the National Framework for Assessing Performance
- Provide fast, convenient access for the public to accredited multimedia advice on lifestyle and health, and information to support public involvement in, and understanding of, local and National Health Service policy development

A. IM&T Principles

The five key principles of the 1998 Strategy are exactly the same as those of 1992, namely:

1. Information will be person based. Priority will be given to person-based systems where data are connected as part of the process of care. Such systems will hold a health care record for each individual, which can be uniquely referenced to that person's new English NHS number and thereby shared with other systems that use the same identifying key.
2. Systems are to be integrated. Wherever possible information should be entered into a computer only once; seamless care needs seamless information. After that it should be available to authorized NHS employees, with steps taken to protect confidential information from unauthorized access.
3. Information will be derived from operational systems. Whenever possible, information is to be captured at the point of delivery of care, from systems used by health care professionals in their day-to-day work. There should be little need for different systems to record management information. Information for management purposes (administrative, financial, research, etc.) should be derived from operational point-of-care systems. Data not connected in a way that helps clinical professionals do their jobs better will not be clinically acceptable and will not be usable for other purposes.
4. Information must be secure and confidential. While recognizing the need for sharing and accessibility of information across organizations, all systems must recognize and respect the principles of privacy, security, and confidentiality. Great care is being taken to ensure that all the data held in a computer will be available only to those who need to know it and are authorized to know it.
5. Information will be shared across the NFIS. Common standards and NHS-wide networking will allow computers to communicate so that information can be shared between health care professionals and organizations, again subject to security and the safeguard of confidentiality.

The specific targets of the new 1998 Strategy are:

- Reaching agreement with the professions on the security of electronic systems and networks carrying patient-identifiable clinical information
- Developing and implementing a first generation of person-based Electronic Health Records, providing the basis of lifelong core clinical information with electronic transfer of patient records between GPs
- Implementing comprehensive integrated clinical systems to support the joint needs of GPs and the extended primary care team, either in GP practices or in wider consortia (e.g., Primary Care Groups/Primary Care Trusts)
- Ensuring that all acute hospitals have the ability to undertake patient administration, including booking for planned admissions, with an integrated patient index linked to departmental systems, and capable of supporting clinical orders, results reporting, prescribing, and multiprofessional care pathways
- Connecting all computerized GP practices to NHSnet
- Providing 24-hour emergency care access to relevant information from patient records
- Using NHSnet for appointment booking, referrals, discharge information, radiology, and laboratory requests and results in all parts of the country
- Developing and implementing a clear policy on standards in areas such as information management, data structures and contents, and telecommunications, with the backing and participation of all key stakeholders
- Community prescribing with electronic links to GPs and the Prescription Pricing Authority
- Routinely considering telemedicine and telecare options in all Health Improvement Programs
- Offering NHS Direct services to the whole population
- Establishing local Health Informatics Services and producing costed local implementation strategies
- Completing essential national infrastructure projects including the networking infrastructure, national applications, etc.
- Opening a National Electronic Library for Health with accredited clinical reference material on NHSnet accessible by all NHS organisations
- Planning and delivering education and training in informatics for clinicians and managers

As a result of yet another round of reforms in the United Kingdom, there is a fundamental change in emphasis of Health Authority information responsibilities (from contracting to public health and service effectiveness) and a need to establish a two-way flow of information between the NHS and the communities it serves. This suggests the development of Health Authorities' information capability may need specific attention in the implementation program for the new NHS, the Public Health White Paper (in due course), and the implementation of this strategy. The effective use of the informatics skills of current public health practitioners will be particularly important.

Health Authorities and their Directors of Public Health already have access to a variety of nationally produced public health, epidemiological, and mortality data. The data presented in the Public Health Common Data Set are of particular value. The new National Framework will supplement this for Assessing Performance, which is currently being road-tested. However, the range of the data available needs to be extended if the vision of Our Healthier Nation and the consequent increased responsibilities are to be met. The information that may be needed to assess resistance to antibiotics is an example of the need to keep information requirements under review.

V. CONCLUSION—QUESTIONS TO BE ANSWERED

If informatics is to be compared to the human body, then the heart of informatics is information theory—the basis for which messages (information) are transmitted or communicated—without a need to live there is no life. The skeletal system on which informatics is built is information science—the foundation on which information is collected, classified, and stored. The nervous system of informatics is information technology, including communications technology—the ways and means in which data and information are moved about. Finally, the soul of informatics is information management—the inner drive that makes information relevant and brings it all together in a meaningful way. It is also the soul that raises questions about the meaning of information (the message) and what form of technology is the most appropriate medium.

Every individual has their own view, their own perception of the world around them. This view is a result of their individual backgrounds, cultures, education, and values. Everyone does not perceive that technology will affect them in the same way. In 1994, a survey of nursing students found that over 95% of them felt that they would never speak to a computer or use an expert system, Yet even at that time, voice

recognition technology had moved out of the research laboratory and expert systems were routinely being used in a growing number of sectors—including health care.

We are witnessing not only the automation of clerical activities, but also the automation of thoughtful technical and clinical work. What are the consequences? Will the responsibility for the production of reliable information rest more with rules incorporated in equipment and on established procedures than on a health professional's judgment? The acute care sector of health care is undergoing dramatic and radical changes in delivery and management, many of which are the result of new technologies and the realities of modern day fiscal constraints. The acute care sector is under increasing pressure to account for its actions, justify its decision making, and its use of resources. Health care organizations the world over are in the process of reengineering and changing the way people do their work.

Is the same degree of change and accountability occurring in the public health sector? Will the public health information systems, which currently support financial accounting and program delivery, be expected to allow costs to be matched to services provided and monitor productivity? Will the public health practitioner of the future be a multiskilled individual whose method of working is dramatically different than today? If not, why not?

SEE ALSO THE FOLLOWING ARTICLES

Health Care, Information Systems and • Medicine, Artificial Intelligence in

BIBLIOGRAPHY

Anonymous. (1998). Epidemiology at the Web café. *Technology Review,* Vol. 101, No. 6, 54.

Bradford, A. (1994). Palmtop practitioners. *British Journal of Healthcare Computing,* Vol. 10, 12–13.

Churchman, C. (1971). *The design of inquiring systems* New York: Basic Books.

Davidow, W.H., and Malone, M.S. (1992). *The virtual corporation: Lessons from the world's most advanced companies.* New York: Harper Business.

Davis, G. (1983). Evolution of information systems as an academic discipline, in *Administrative Sciences Association of Canada Conference Proceedings,* pp. 185–189.

Denning, P. (1999). *Talking back to the machine.* New York: Copernicus Books.

Friede, A., *et al.* (1994). CDC WONDER: A co-operative processing architecture for public health. *Journal of American Medical Informatics Association,* Vol. 1, No. 4, 303–312.

Hannah, K. (1985). Current trends in health informatics: Implications for curriculum planning, in *Computers in Nursing.* Amsterdam: North-Holland.

Hicks, J. (1993). *Management information systems: A user perspective.* New York: West Publishing Co.

Kaiser, L. (1999). Quantum leaps in healing. *Health Forum Journal,* Vol. 42, No. 4, 50.

Keen, J. (Ed.). (1994). *Information management in health services.* Buckingham: Open Univ. Press.

Levy, A. H. (1977). Is informatics a basic medical science? in *Medinfo 1977, Proceedings* (D. Shires and H. Wolfe, Eds.), pp. 979–981. Amsterdam: North-Holland.

Linthicum, D. (1994). Tommy, can you see me? *Open Computing,* September, 67–68.

McNurlin, B. C., and Sprague, R. H. (1989). *Information systems in management practice.* Englewood Cliffs, NJ: Prentice Hall.

Meadow, C. T. (1979). Information science and scientists in 2001. *Journal of Information Science,* Vol. 1, 217–221.

Moehr, J. R., *et al.* (1979). Specialized curriculum for medical informatics—Review after 6 years of experience, in *Proceedings of the International Conference in Medical Computing, Berlin.*

Mullin, S. (1994). Start talking to your computer—Three physicians rate IBM's speech recognition system. *Canadian Medical Informatics,* Vol. 1, No. 3, 16–17.

Ohlson, K. (1998). Non-US net users to dominate by 2002. *Computer World,* July 16.

Plucauskas, M. (1994). Internet and medicine part 11: Hooking up and using the Internet. *Canadian Medical Informatics,* Vol. 1, No. 3, 28–30.

Protti, D. J. (Ed.). (1982). A new under-graduate program in health informatics, in *AMIA Congress 1982 Proceedings,* pp. 241–245. San Francisco: Masson.

Radford, K. J. (1978). *Information for strategic decisions.* New York: Reston Publishers.

Ranade, W. (1994). *A future for the NHS: Health care in the 1990s.* London: Longman.

Relchertz, P. (1973). Protokoll der Klausurtangung Ausbildungsziele, inhalte und. *Methoden in der Medizinischen Informatik,* Vol. 2, 18–21.

Scott Morton, W. (Ed.). (1992). *The corporation of the '90s.* Boston: Harvard Univ. Press.

Shannon, C. E., and Weaver, W. (1960). *The mathematical theory of communication.* Urbana, IL: Univ. of Illinois Press.

Shires, D., and Ball, M. (1975). Update on educational activities in medical informatics, in *Proceedings of the 5th Annual Conference of the Society for Computer Medicine,* pp. 52–54.

Smith, R., and Gibbs, M. (1994). *Navigating the Internet.* Indianapolis: Sams.

Strauss, P. (1994). Beyond talking heads: Videoconferencing makes money. *Datamation.* Vol. I, 38–41.

Tapsell, S. (1998). Telling it like it is with teleconferencing. *Management,* Vol. 45, No. 3, 65.

Taubes, G. (1998). Virus hunting on the web. *Technology Review,* Vol. 101, No. 6, 50.

van Bemmel, J. H. (1984). The structure of medical informatics. *Medical Informatics,* Vol. 9, 175–180.

Wiener, N. (1948). *Cybernetics.* Newark, NJ: Prentice Hall.

Quality Information Systems

Fatemeh Zahedi
University of Wisconsin, Milwaukee

GLOSSARY

customers We refer to those who benefit from the services of an information system as its "customers" rather than "users." This broadens the category of those who benefit from the system's services to include its internal as well as external customers.

information systems We define an information system to include software, hardware, processes, human services, inputs, and outputs. In other words, all factors that lead to the production of information services are considered to be parts of the system.

quality There is no universal agreement on the definition of quality. However, a number of leaders in the quality movement have provided definitions that have a common focus on the customer's needs, requirements, and expectations. Feigenbaum (1991) defines quality as "[t]he total composite of product and service characteristics of marketing, engineering, manufacturing, and maintenance through which the product and service in use will meet the expectations of the customer." Other definitions of quality include: "conformance to requirements" and "fitness for use."

quality information systems (QIS) Involve the application of TQM to the development and operation of information systems.

requirement versus utility There is a difference between these two concepts. Customer requirements are what the customer needs; utility is the extent of the customer's satisfaction when the need is fulfilled. There is also a difference between customer requirements and the internal requirements of a company. It is normally the customer's requirements and wishes that lead to the determination of a company's internal requirement, particularly in systems development.

total quality management (TQM) A collection of principles, concepts, tools, and processes, all designed to promote quality within an organization and its functions as well as in interactions with its customers, suppliers, and environment. The word *total* emphasizes the all-encompassing nature of TQM (all processes, functions, and people). The word *management* adds organizational and behavioral components to the technical scope of quality, emphasizing its business focus.

I. ORIGIN AND LEADERS OF THE QUALITY MOVEMENT

The first systematic attempt toward increasing quality was the scientific management theory, pioneered by Fredrick Winslow Taylor (1947, originally 1911). In this theory, production was considered a closed system, which was to be improved by scientific and technical methods. The human-behavior component of quality has its origin in a series of studies (mostly performed at Western Electric's Hawthorne plant), which showed that workers increased their productivity when supervisors paid them special attention as reported by

Roethlisberger and Dickson in 1939. This was called the *Hawthorne effect* and introduced a behavioral approach to management. The attention to quality of industrial production coincided with the development of modern statistical theory. W. Edwards Deming, a physicist and later a statistician, began advocating the use of statistics in the service of industry and humanity. As a graduate student, Deming worked at the Hawthorne plant (10 years earlier than the Hawthorne experiments) and saw the impact of the industrial division of labor. Later, he worked with Shewhart to aid the war effort during World War II. After the war, Deming provided extensive advice to Japanese industry. Although Deming started from a statistician's point of view, he broadened his perspective and developed a 14-point principle in total-quality management.

Juran is another early contributor to the total quality management (TQM) movement. He emphasized the management and technical aspects of quality control and developed a 10-step approach to quality management. He advocated developing processes for quality control, quality improvement, and quality breakthroughs.

Feigenbaum, another TQM pioneer, was the first to coin the term total quality control (TQC), and in 1956 identified ten benchmarks for controlling quality. He introduced the cost of quality concept, which rejects the idea that higher quality requires higher costs, and advocated the development of measures and the collection data to assess the cost of quality.

In 1962, Kaoru Ishikawa, at Tokyo University, started quality circles in Japan. Quality circles consist of small, voluntary teams of workers who develop and monitor quality-control activities in their units, as a part of a company-wide quality program. Quality circles were the forerunners of TQM's quality team concept. Ishikawa advocated TQC, collecting quality data, and the use of cause-effect diagrams for identifying quality issues. Taguchi, another quality pioneer, has focused on design quality with an emphasis on robustness and minimum variation in product design.

Crosby has a strong focus on the behavioral and cultural aspects of TQM. He is the proponent of the zero-defect approach and "do it right the first time." His quality management philosophy is reflected in his 14-step program. Crosby's approach has been influential in the development of the Capability Maturity Model by the Software Engineering Institute (SEI), which has incorporated various elements of quality paradigm and has brought into focus the importance of quality concepts in software engineering practice as reported by Humphrey in 1991 and Fox and Frakes in 1997.

II. QUALITY MANAGEMENT FACTORS

The leaders of TQM created an extensive body of literature offering practical advice on how to implement TQM, based on their personal observations and experiences. They have identified factors important in implementing TQM. These factors have been alternatively called "points," "principles," "infrastructures," "critical success factors," "levers," and more recently "strategies" and "concepts."

Total quality management was mostly implemented by the adoption of strategies promoted by one of the previously mentioned quality leaders. For example, Deming proposed his 14 principles in the form of prescriptive commands:

1. Create constancy of purpose
2. Adopt the quality and customer-orientation philosophy
3. Cease dependence on inspection to achieve quality
4. End the practice of rewarding business on the basis of the price tag
5. Improve constantly and forever
6. Institute training
7. Institute leadership
8. Drive out fear
9. Break down barriers between departments
10. Eliminate slogans, exhortations, and arbitrary numerical goals
11. Eliminate numerical quotas
12. Remove barriers that rob employees of their pride of workmanship
13. Institute a vigorous program of education and self-improvement
14. Take action to accomplish the transformation

In 1992, Juran advocated managing quality in three parts: planning, control, and improvement. He recommended seven breakthrough sequences:

1. Breakthrough attitude
2. Identify the few vital projects
3. Organize for breakthrough in knowledge
4. Conduct analysis
5. Determine how to overcome resistance to change
6. Institute change
7. Institute controls

The dependency on an individual leader's school of thought has gradually been decreased by two developments. Guidelines for quality awards and standards,

particularly those of the Baldrige Award in the United States and ISO 9000 international standards, have synthesized, refined, and organized quality strategies into well-developed and detailed recommendations for practical implementations. (Quality awards and standards will be discussed in a later section.) The second trend has been the publication of systematic empirical research on quality factors and their impacts on performance outcomes, as discussed below.

In their research, Adam, Hershauer, and Ruch identified "levers" of quality improvement as: management's strategic support, collecting and analyzing quality data, and training. They developed a model for quality performance based on the organizational theory. Mondon, in his 1982 research into Toyota's quality management, also listed employee and supplier participation, process design, and strong top management support as important factors in TQM.

Garvin was among the first to use systematic observations for identifying the important factors in TQM. He observed and analyzed data from 16 air-conditioner manufacturers in the United States and Japan, and found that companies leading in quality had:

1. Management support for quality
2. Extensive goal setting
3. Cross-functional employee participation
4. Good information systems for communicating about quality
5. Quality-based product design
6. Focus on quality in production scheduling
7. Emphasis on defect-free products
8. Quality training and quality circles for workers
9. Strong vendor relationships

Saraph, Benson, and Schroeder in 1989 were the first to undertake research on developing constructs for TQM factors. (A construct is a variable that is not directly observable and must be observed through its indicators. For example, a person's IQ is a construct that is not directly observable and could be measured only through the answer to a number of questions as indicators of IQ.) They generated 120 organizational prescriptions for effective quality management from an extensive review of quality literature, and grouped them into eight categories, which also embody the prescriptions of the quality leaders. Their categories are:

1. The role of the top management leadership
2. The role of the quality department
3. Training
4. Product/service design
5. Supplier-quality management
6. Process management
7. Quality data and reporting
8. Employee relations

They developed the survey instrument for measuring the TQM strategy constructs and validated them by collecting data from 162 managers in 22 randomly selected firms.

Anderson, Rungtusanatham, and Schroeder in 1994 developed a theory of quality management based on Deming's writings. They used a Delphi approach to identify the underlying concepts in Deming's 14 points. (The Delphi method was developed by the RAND Corporation in the 1950s for organizing opinions of complex matters by iterative input from a panel of experts until a consensus is reached.) Anderson and colleagues used a panel of 7 experts to identify 37 concepts underlying Deming's 14 points, and then grouped them into seven categories:

1. Visionary leadership
2. Internal and external cooperation
3. Learning
4. Process management
5. Continuous improvement
6. Employee fulfillment
7. Customer satisfaction

Flynn, Schroeder, and Sakakibara in 1995 collected data from 706 plant managers from 42 randomly selected manufacturing companies in order to identify the strategies for quality management as:

1. Top management support
2. Product design process
3. Process-flow management
4. Work-force management
5. Work attitude
6. Statistical control and feedback
7. Supplier relationship
8. Customer relationship

Ahire, Golhar, and Waller in 1996 identified eleven factors in TQM as:

1. Top management commitment
2. Employee training
3. Design-quality management
4. Using statistical process control
5. Internal quality information use
6. Benchmarking

7. Employee empowerment
8. Employee involvement
9. Supplier performance
10. Supplier quality management
11. Customer focus

As one can see, both TQM leaders and researchers have identified similar factors or strategies for TQM. In what follows, we discuss TQM factors and their importance in quality information systems: quality leadership, human resource strategies, quality control and tools, quality design, process improvement strategies, customer focus, satisfaction, and involvement strategy, and benchmarking strategy.

A. Quality Leadership

Leadership has been advocated as a critical part of quality management by all quality pioneers as reported by Saraph and colleagues in 1989. From their theory-building analysis of Deming's views, Anderson and colleagues in their 1994 review define the underlying concept of visionary leadership as

> [t]he ability of management to establish, and lead a long-term vision for the organization, driven by customers' requirements, as opposed to an internal management role. This is exemplified by the clarity of vision, long-term orientation, coaching management style, participative change, employee empowerment, and planning and implementing organizational change.

Top management leadership in quality requires the creation of a shared vision to promote the constancy of purpose and the uniformity of direction in the organization's activities. A *vision* is defined as the desired or ideal state of the unit. The leadership of the organization develops the vision through a participatory process, which includes the organization's stakeholders, consisting of employees, stockholders, managers, and in some cases the representatives of internal and external customers. Developing vision is not limited to the organization as a whole, the critical functional areas should also develop their visions based on the overall vision of the firm. This is particularly important for information systems (IS). An example of information technology (IT) vision is: "an integrated and reliable information technology that meets its customers' requirements for timely, accurate, and on-demand information."

A number of related concepts make the vision tangible and operational: mission, values and principles, goals, strategies, plans, and tactics. While vision describes *what* the desired state is, a mission describes *how* the organization should move toward its vision. Values and principles are the core beliefs under which the organization operates. Goals are often defined in terms of measures (often numerical) that lead the organization toward its vision. Strategies, plans, and tactics constitute road maps of actions leading to the vision.

The Baldrige Award identifies indicators of quality leadership as the ways senior leaders in the organization (1) set, communicate, and deploy organizational values, (2) set and communicate performance expectations, (3) focus on customers and other stakeholders, (4) create and reinforce an environment of empowerment and innovation, (5) encourage and support organizational and employee learning, and (6) address the organization's responsibilities to the public and its key communities (see reports by Baldrige National Quality Program). Part of quality leadership involves the clear choice of quality as a major strategy and setting quality goals and policies. One example of quality policy is to evaluate the performance of managers based on their quality efforts and allocate adequate resources for quality improvement (see work of Ahire and colleagues).

Empirical studies have identified quality leadership and top management commitment as a critical factor in a number of diverse organizations. Ahire and colleagues report on diverse organizations, such as Sahi Breweries in Japan, Xerox in the United States, Dunlop in Malaysia, and Dow-Corning in Australia, in which the role of top management commitment has proven to have a critical role in quality management. In a study involving a survey of IS executives of Fortune 1000 companies, Ravichandran and Rai have identified top management leadership as one of the important factors in a quality-oriented organizational structure for software development. They have found that top management leadership in quality is significantly related to management infrastructure that facilitates quality deployment in IS organizations. Thus, empirical findings support and reinforce the views by the TQM pioneers and researchers regarding the importance of quality leadership and provide evidence for its relevance in creating quality information systems.

B. Human Resource Strategies

Anderson and colleagues in their 1994 review define employee fulfillment as

> [t]he degree to which employees of an organization feel that the organization continually satisfies their need. This is exemplified by job satisfaction, job commitment, and price of workmanship.

Human resource strategies are even more critical to quality information systems, which are staffed with knowledge workers who are in short supply and expensive, and on whose experience and skills a company depends for competing in the market. Three human-resource factors—employee involvement, empowerment, and training—are viewed as critical to achieving employee fulfillment.

Employee involvement—The participation and involvement of employees have been shown to increase their commitment to quality (see work of Oliver). Quality circles and cross-functional quality-improvement teams have been the popular vehicles for employee involvement. Employee involvement and participation should accompany the appropriate information collection measures, evaluations, and rewards. A number of organizations use profit-sharing programs (and more recently stock options) as a reward mechanism to enhance employees' sense of ownership and participation in quality improvement (see work of Ahire and colleagues). Another mechanism for employee involvement is to collect employee suggestions, take action on useful suggestions, and reward employees whose ideas have been used to improve quality. Recognition, acknowledgement, and other nonmonetary rewards have been shown to motivate employees' involvement and participation.

Employee empowerment—One of the important incentives for employee involvement is empowerment, as pioneered by Toyota. The focus of quality management is to move the decision to the source, or to employees who are directly involved with the outcome. Ahire and colleagues identify the indicators of employee empowerment as the ability to inspect what they do, encouragement and access to resources for identifying problems and fixing them, and an environment with a problem-solving orientation. Focusing on information and software systems, Ravichandran and Rai measured programmer/analyst empowerment as: (1) participation of team members in project planning, (2) participation of team members in resource allocation, and (3) participation of team members in project scheduling.

In their study of Japanese quality management, Ebrahimpour and Ebrahimpour and Withers report on the empowerment of production workers to stop production based on their assessment of the process. In a case study of Corning Incorporated, Shrednick, Shutt, and Weiss discuss the information-service division's successful approach to employee empowerment through self-managing teams. Employee empowerment is therefore a key factor in stimulating greater participation of employees and is critical to the human-resources strategy.

Employee training—Quality leaders and gurus have been unanimous in identifying training as a major quality strategy (see work of Saraph and colleagues). Anderson and colleagues in their 1994 review define the underlying concept of learning as

> [t]he organizational capability to recognize and nurture the development of its skills, abilities, and knowledge base. This is exemplified by company-wide training, fundamental knowledge, process knowledge, educational development, continuous self-improvement, and managerial learning.

It is argued that if employees do not have adequate knowledge of quality management, their involvement and empowerment would not produce positive results. With increasing recognition of workers' skills and knowledge as the most important organizational asset, training is considered an investment, rather than a cost. A national focus on systematic and frequent training of Japanese workers has been recognized as one of the major contributors to Japan's success in quality initiatives as reported by Ebrahimpour in 1985 and Juran in 1981. Shrednick and colleagues report that empowerment implementation involves greater opportunities for employees "to learn and use new skills and knowledge," with substantial annual savings to the company. It is through quality training that employees understand the value of their participation and execution of their empowerment to the organization.

Ahire and colleagues identify indicators of employee training as the availability and frequency of training, including quality training, levels of employees in the same training course, and employee satisfaction with their training experiences. Ravichandran and Rai have included "quality orientation of rewards" and "commitment to skill development" as components of a sophisticated management structure for supporting quality in software development. They have found that quality requires a management infrastructure and quality environment that promotes quality in developing information systems. In other words, employee involvement, empowerment, and training create a common purpose among employees and internalize quality values that propel workers to want to do their jobs well. This is even more critical in IS environments in which superior outcomes depend heavily on knowledge workers' creativity, knowledge, and self-initiatives.

C. Quality Control and Tools

In his writing, Juran focuses on the importance of planning and designing quality into products and services.

Taguchi has proposed methods for designing quality in products based on concepts of experimental design, minimum variation, and loss function. Shingo promote error-proofing techniques as reported by Robinson and Schroeder in 1990. Quality deployment function (QFD) has been proposed as an effective method for translating customers' needs to the design of products and systems. Reviews by Zahedi and Zultner give examples. Visual tools such as storyboarding for visualizing various quality tools have also been developed for use problem identification and problem solving.

Monitoring quality on a continuous basis is an important aspect of quality management, and keeps the quality focus within the organization. One of the important approaches in continuous monitoring of quality is statistical quality control, which has been used for some time to detect variations in manufacturing quality. (Statistical quality control is discussed in a later section.) Brassard and Zahedi have reported on a number of other tools for quality control, including scatter diagrams, Pareto charts, cause-effect diagrams, and control charts. These charts make it possible to visualize the quality behavior of a system and detect its deterioration. The increased reliance of organizations on information systems for their internal and external activities makes controlling and monitoring the quality of information systems a critical factor in organizations' success in dealing with their customers and suppliers, as well as increasing the efficiency and effectiveness of their operations.

D. Quality Design

Taguchi has been one of the pioneers in focusing on designing quality into a system or product. Quality design is highly critical in IS. While the end-user involvement has long been an important ingredient in creating effective information systems, the TQM approach to information systems expands the level of involvement to include the major stakeholders: end users, systems developers, programmers, indirect internal users, and indirect external users in all stages of systems development. One of the tools that could be used in quality information systems (QIS) is the QFD and its corresponding mapping scheme "house of quality." Zahedi reported in 1995 that this approach could be used successfully in mapping customers' requirements to technical specifications and systems designs. Combined with the relative ratings derived from using the analytic hierarchy process (AHP), QFD could incorporate the ratings of users for various parts of the system and compute the relative importance of each technical component within the system. Such a rating allows the managers to allocate higher levels of resources to higher rated components in the system.

Designing quality in an information system involves design for maintainability, ease of assembly, ease of maintenance, and reliability. One approach that makes it possible to incorporate these objectives is called concurrent engineering. Concurrent engineering has its origin in the early development of car design, and has been revived by Lockheed and the Defense Advanced Research Projects Agency (DARPA) initiative in concurrent engineering. This approach involves concurrent and multiple designs of various parts of the system through the participation of all involved in the systems-development process, and the selection of final design based on the design goals. The possible availability of reusable software components makes it possible to reduce the cost of concurrent engineering by selecting competing software components for systems development. Ravichandran and Rai measure the formalization of analysis and design in IS by (1) the use of formal methods for eliciting requirements, (2) use of idea-generating techniques, (3) use of formal quality techniques for design, such as QFD, and (4) the use of standard representation schemes such as entity-relation (ER) diagrams and QFD. Ravichandran and Rai have found that formalizing reusability in systems design, as well as formalizing design methods, are important ingredients of effective process management that increase performance quality in information systems.

E. Process Improvement Strategies

There are a number of strategies for process improvement, including continuous improvement, reengineering, process management, zero defect, and software capability maturity model, as discussed below. In many cases, these strategies are complementary.

Continuous improvement—Anderson and colleagues in their 1994 review define the underlying concept of continuous improvement as "[t]he propensity of the organization to pursue incremental and innovative improvements of its processes, products, and services. This is exemplified by continuous improvement." Deming in 1986 advocated continuous improvement in one of his imperatives: "[i]mprove constantly and forever the system of production and service." In 1986, Imai reported that continuous improvement has its origin in the Japanese concept of *kaizen,* which means "ongoing improvement involving everyone—top management, managers, and workers." In the United States, the idea of the continuous improvement cycle was developed by Shewhart and popularized by Deming. The cycle has four components: plan, do, check, and act. This cycle is continu-

ously implemented in order to improve quality in the processes and outcomes. Ravichandran and Rai observe that the additional benefit of this cycle is the generation of process knowledge that could be used across teams and projects to enhance performance. Innovation is an important part of the concept of continuous improvement. However, in this case, innovation is less focused less on paradigm shifts, and more on incremental changes through observations, testing, and reinvention. (See 1994 review by Anderson and colleagues). Continuous improvement is the cornerstone of every quality undertaking in any organization.

Reengineering—Continuous improvement emphasizes relatively small changes, while business-process reengineering involves sweeping changes at the corporation or functional level. Hammer and Champy define reengineering as "[t]he fundamental rethinking and radical design of business processes to achieve dramatic improvements in critical, contemporary measures of performance, such as cost, quality, service, and speed." Based on their observations of reengineering efforts in major corporations, they have found that reengineering has the following characteristics: it is focused on process, is sweeping and extensive, breaks the rules, and relies on the innovative use of information systems. The common features in business-process reengineering are combining jobs, empowering employees as decision makers, creating multitrack processes, accomplishing the work in a natural and common-sense way, reducing costly controls and checks, reducing the number of external-contact points, creating one contact point through a case manager, and using a combination of centralized and decentralized structures.

Process management—Anderson and colleagues in their 1994 review define the underlying concept of process management as

> [t]he set of methodological and behavioral practices emphasizing the management of process, or means of actions, rather than results. This is exemplified by management of processes, prevention orientation, reduction of mass inspection, design quality, statistical process control, understanding of variation, elimination of numerical quotas, elimination of management by objectives, elimination of merit-rating reward systems, understanding motivation, total cost accounting, and stable employment.

Ravichandran and Rai provide measures for process control, process efficiency, and process maturity and have found that process management has a direct effect on quality outcomes in information systems.

Zero defects—Motorola, with its 1987 six-sigma initiative, has focused its quality effort on reducing defects to zero. This could be accomplished with a relentless pursuit of designing processes, and finding methods and approaches that move the quality metrics close to zero defects. Quality metrics are used for identifying variability in quality and reducing it through the continuous improvement process. Variability is normally measured by variance or standard deviation from the mean or expected value of the metric. Six sigma or six standard deviations (three standard deviations on each side of the expected value of the metric) practically covers the entire region of values for the metric in most cases. As this region is tightened through setting goals, the system metric moves closer to zero variability, or zero defects if the metric is used to measure defects.

Software capability maturity model—The software capability maturity model (SW-CMM) was developed at the Software Engineering Institute (SEI) at Carnegie Mellon University. It is a normative model for helping organizations gradually progress from an initial and ad hoc process of software development to a mature, optimized, and disciplined stage. Herbsleb and colleagues discuss five levels in SW-CMM: initial, repeatable, defined, managed, and optimizing as follows. At the *initial* level, the software process is in an ad hoc and at times chaotic stage, and successful outcome depends on individuals and their heroic efforts. At the *repeatable* level, the focus is on project-management processes. At this level, costs, schedule, and functionality are tracked within a basic project-management process with the goal of repeating earlier successes with similar software projects. The *defined* level emphasizes engineering processes and organizational support for such processes. At this level a standard software process is developed to include the management and engineering activities for software development. Projects use a version of the standard software process that is tailored and approved for developing and maintaining software projects. At the *managed* level, the focus is on product and process quality. At this level, detailed quality metrics are developed for measuring and controlling the quality of software processes and products. At the final stage, *optimizing level,* the focus is on continuous improvement, and the continuous-improvement process is in place to facilitate quantitative feedback from the process and for piloting and implementing innovative ideas and technologies. Ravichandran and Rai have developed a scale for measuring the five stages of process maturity.

Several case studies have shown dramatic improvements in performance and in cycle time using SW-CMM (see work by Dion, Hollenbach and colleagues, and Humphrey and colleagues). Hollenbach, Pflugard, and Smith report on some of the lessons learned in implementing SW-CMM. Herbsleb and colleagues discuss

a number of case studies regarding the performance results of implementing SW-CMM. They report 9–67% productivity gains per year, 15–23% reductions in time to market per year, 10–94% reductions in post-release defects per year, and a business value ratio from 4:1 to 8.8:1 as the result of implementing SW-CMM.

The SW-CMM addresses the quality of software engineering (development) processes, hence has the perspective of developers or producers of software systems. More recently, the SEI has developed a comparable CMM for software acquisition (SA-CMM), which addresses the quality of processes for the acquisition of software, and provides buyers perspective (see work by Fergusson and colleagues).

The major criticism of CMM-based improvements has been the increase in rigidity and bureaucracy in the organizations using the CMM. In their analysis of the impacts of implementing quality processes on software quality, Krishnan and colleagues have found that adherence to development practices in the CMM framework is associated with higher software quality. There is a common belief that high software quality could only be achieved by lengthier development-cycle time and higher costs. In an empirical study, Harter, Krishnan, and Slaughter have found that using the CMM is associated with higher product quality as well as a higher development effort. However, their findings show that gains in quality and reduction in cycle time outweigh the additional efforts needed to achieve them, leading to a positive net effect due to using the CMM approach.

F. Customer Focus, Satisfaction, and Involvement Strategy

Customer satisfaction is the main motivator of the total quality approach, and the measurements of customer satisfaction have been mostly based on the customers' perception of the quality of products and services (see 1994 review by Anderson and colleagues). Deming and other quality leaders encouraged companies to focus on customers' needs, wishes, and delights, and to consider customers as the most important part of the production line as reported by Scherkenbach in 1986. In his writings, Deming argued that customer satisfaction leads to customer loyalty, which is a great contributor to organizational well being. Loyal customers are now considered a major business asset because the costs of acquiring new customers are much higher than keeping the existing ones, and customer-relations management has recently emerged as a critical business function.

While traditional marketing and economic literature have ascribed positive costs to quality, in that higher quality is more costly, Deming and a number of other quality pioneers have argued that higher quality products are not necessarily more costly. The statement that "quality is free" implies that process improvements inherent in TQM lead to reduced cost with higher quality. Therefore, attaining a higher level of customer satisfaction does not necessarily imply higher expenditures. However, the organization should be willing to make the initial investment in quality in order to reap its benefits at a later stage.

Ahire and colleagues measure the customer focus of an organization using a four-item scale, which includes customer-satisfaction survey feedback to managers, availability of customer complaints to managers, extent to which customer feedback is used for product improvement, and customer focus on quality management. Ravichandran and Rai propose a three-item scale for measuring user participation in IS development as the active participation of users in (1) determining system requirements, (2) identifying input/output needs, and (3) developing test plans.

Before quality and customers became the major industry focus in the global market, the field of IS paid special attention to user satisfaction as an important measure (see work of Ives and Olson and Roby and Farrow). Cyert and March emphasize the importance of users' needs in IS. Ives, Olson, and Baroudi have observed that "satisfaction of users with their information system is a potentially measurable, and generally acceptable, surrogate for utility in decision making." There is a large body of literature in defining and measuring the satisfaction of IS users. However, few studies formally incorporate satisfaction measures in developing decision models for IS managers. Sahin and Zahedi have developed decision models for fixing, maintaining, and upgrading systems based on the customer-satisfaction metric. They formulate an optimization model in which defects and shortcomings in the system are discovered randomly through time, and the decisions to change the system by fixing its defects, improving it, or upgrading it depend on the impact of the decision on the customer-satisfaction metric. This model includes the impact of technological change on customer satisfaction as well.

Using customer satisfaction metrics in making critical decisions regarding information systems creates a customer-focus in the decision-making process and makes customers a participant in major decisions, as recommended strongly in TQM. This is supported by empirical findings in Ravichandran and Rai's study that stakeholder participation plays a significant role in quality performance of IS organizations.

G. Supplier Relationship and Performance Evaluation Strategy

Quality management emphasizes internal and external cooperation. Internal cooperation manifests in close collaborations among various levels, units, and individuals in non-competitive and mutually beneficial forms, such as quality teams, and supervisors as facilitators. External cooperation manifests mainly through cooperation between a firm and its suppliers. Recent approaches to purchasing, such as just-in-time (JIT) and zero-inventory, are based on this type of collaboration. Supplier relationship and supplier performance evaluation are critical in developing a supplier strategy.

1. Supplier Relationship

The suppliers' quality focus, as well as their involvement, has been considered an important aspect of quality management. The quality of incoming parts or services determine the level of inspection by the organization, has great impact on the quality of the final product or service, and influences the organization's ability to meet customers needs and demands. Xerox and Ford have been among the companies that have developed long-term relationships with their quality-oriented suppliers. Such relationships are developed based on extensive and frequent supplier-evaluation systems. Some companies provide assistance in quality management to their suppliers for ensuring better inputs (see work of Denton). The importance of supplier management has increased with the movement by many companies to outsource some of their activities (such as IS) or to spin off a function of the company as an independent supplier (such as manufacturing). The effective negotiation with such suppliers and the relationship management with them have become increasingly more central in today's business environment.

Ahire and colleagues propose six indicators for measuring the extent of supplier quality-management efforts. They include the relative importance that a company places on the quality of purchased products, on the supplier's technical capability, on the supplier's financial capability, on the supplier's delivery performance, on providing technical assistance to suppliers, and on long-term relationships with suppliers. Ravichandran and Rai propose a two-item scale for measuring vendor participation in information systems as: (1) establishing long-term relationships with vendors and consultants and (2) making vendors and consultants an integral part of the system process.

2. Supplier Performance Evaluation

Poor quality of incoming parts and services contribute to low quality of the final product (see work by Flynn and colleagues). Ahire and colleagues advocate measuring the supplier's performance as a construct related to quality management. They propose a six-item set of indicators for measuring supplier performance including: the performance, conformance, reliability, and durability of supplied parts as well as the cooperation of suppliers in resolving quality problems and suppliers' willingness to improve quality.

Vendor relationships in IS have been viewed as an important factor in quality information systems. Ravichandran and Rai include vendors in their definition of stakeholders, and have reported that stakeholder participation (including vendors) plays a key role in product quality and process efficiency. This relationship becomes even more critical as organizations outsource parts of their IS functions. In this case, a close tie with the supplier becomes a critical aspect of quality programs in the organization.

H. Benchmarking Strategy

Another avenue for external cooperation is benchmarking. The underlying philosophy of benchmarking is a learning organization—the firm learns from the best practices of its competitors and noncompetitors. Anderson and colleagues write that a learning organization engages in "knowledge-seeking activities at the individual, group, and organizational levels." While learning at the individual and group levels takes place through training and team activities, organizational learning from the environment could take place through benchmarking activities.

Benchmarking has a long history in China and Japan. The Japanese word *dantosu* (the best of the best) embodies the idea of benchmarking. In the United States, Juran posed the question: "What is it that organizations do that gets results much better than ours?" The first known comprehensive benchmarking in the United States was carried out by Xerox in 1979 (see review by Camp); Motorola adopted the strategy in the early 1980s.

Benchmarking is a part of the continuous-improvement process. It requires: self-evaluation, identification of weak spots, definition of metrics, and identification of processes, policies, and structures of interest. These steps provide the focus and purpose for benchmarking activities. Benchmarking requires the identification of benchmarking partner(s), planning,

information collection, analysis, goal setting, and implementation. These steps were discussed in detail by Zahedi in 1995.

To select the appropriate benchmarking partner, one needs to select the appropriate type of benchmarking: internal, competitive, functional, or generic. In locating the benchmarking partner, one can use a number of resources such as: internal experts, internal functional areas, professional associations, journals, external experts and consultants.

There are a number of issues that should be addressed, such as obtaining the benchmarking partner's consent, forming benchmarking teams, selecting methods of benchmarking (such as observations, interviews, questionnaires, site visits, telephone surveys, and mail surveys), analyzing the results, communicating the recommendations, and implementing them. In successful cases of benchmarking, companies could continue the benchmarking partnership and/or make benchmarking an integral part of the continuous-improvement process.

Benchmarking is particularly valuable for information systems. The speed of technological change and the relative paucity of information regarding the innovative use of IT make benchmarking a requirement for creating quality IS. Since business-process reengineering often requires the integration of information systems in the new process, benchmarking the use of information technology in business processes is extremely useful.

Ahire and colleagues propose five indicators for measuring the extent of benchmarking in organizations: the emphasis on benchmarking competitors, benchmarking noncompetitors, effectiveness of benchmarking in quality improvement, effectiveness of benchmarking in product cost reduction, and the willingness of organizations to benchmark in the future.

Benchmarking is even more critical for IS organizations, since the technology changes rapidly and companies do not have adequate time or resources to internally evaluate the impacts of new technologies or approaches. Therefore, they need to heavily rely on the experiences of early adopters of the technology. However, benchmarking for QIS is not limited to new technologies—cross-industry or cross-functional benchmarking for developing new approaches in IS development or deploying could yield great benefits in quality and costs.

III. SOFTWARE QUALITY ASSURANCE

The quality approach to system development without using TQM has evolved in four areas: software reliability and quality assurance, design methodologies, software-process management, and participative design (see work of Ravichardran and Rai). Adapting the 1991 IEEE-STD-60 standard definition of quality assurance to software, one can define quality assurance as a planned and systematic pattern of all actions necessary to provide adequate confidence that the software process and product conform to established technical requirements. Software quality assurance generally focuses on both process and product quality.

Software assurance has a number of dimensions, including reliability, portability, efficiency, human engineering, and maintainability (see review by Rai and colleagues). Of these dimensions, software reliability has long been the focus of researchers and practitioners. An extensive body of literature has emerged in the last three decades in software reliability and consequently there are more than twenty models in software reliability. These models have been reviewed, evaluated, and categorized extensively (see, for example, work by Goel, Musa, and Zahedi and Ashrafi). The focus of most of these models is on evaluating the reliability of the software after its development.

There is little research on the reliability of IS. Although millions of dollars are spent in developing and deploying IS, little attention has been paid to formal metrics of IS performance. An exception is the work by the SIM Working Group that provides guidelines for implementing quality assessment and planning tools in information systems. Zahedi has followed the SIM Group's recommendation in synthesizing customer-satisfaction measures and system-performance data into a single reliability metric for information systems.

Determining the level of reliability at the design time for various components of IS for the purpose of resource allocation is a relatively recent concept. Zahedi and Ashrafi have developed a reliability-allocation model based on the customer's utility for various aspects of information or software systems. Their approach is more useful in developing complex systems or in performing cost/benefit analysis for purchasing components of information systems.

IV. DATA AND INFORMATION QUALITY

One of the important aspects of using information technology in TQM, as well as creating and operating QIS, is the quality of data and information used in these systems. Strong, Lee, and Wang report that 50–80% of criminal records in the United States contain poor quality data; and that low data quality has a

social and economic cost in the billions of dollars. Redman discusses far-reaching impacts of low data quality on operational, tactical, and strategic decisions within an organization.

One approach for improving data quality is to view the creation, storage, and use of data as a manufacturing process (see work by Wang). The application of TQM principles to the data- and information-manufacturing process is called total data quality management (TDQM). Wang defines the TDQM's continuous-improvement cycle as: define, measure, analyze, and improve, similar to the Shewhart-Deming cycle. Orr suggests a systems-theory approach to data quality, in which customers and users of data are a significant part of the continuous-improvement feedback loop. In this approach, data quality is defined on the basis of the use of data, and is improved by users' feedback.

Although TDQM is relatively new, a number of researchers are working on developing theories and methodologies for dealing with data and information quality within the context of TDQM.

V. QUALITY METRICS

In order to evaluate quality objectively, one needs to define metrics that formalize and quantify quality. The handbook on metrics published in 1991 by the United States Air Force defines a metric as a combination of measures designed for depicting an attribute of a system or entity. The handbook lists the characteristics of a good quality metric:

1. Meaningful to customers
2. Containing organizational goals
3. Simple, understandable, logical, and repeatable
4. Unambiguously defined
5. Capable of showing trends
6. Economical in data collection
7. Driving appropriate action
8. Timely

In developing metrics, it is important to apply the following principles for its success: (1) make collecting accurate data beneficial to the data collectors, (2) use metrics that are flexible and responsive to customers' needs, (3) define the process prior to the development of metrics, (4) verify the measurement through establishing feedback, and (5) develop rewards, recognition, and advice procedures for those whose work is being measured (see work of Schulmeyer). It is also important to define how, by whom, and when the metrics are used before data collection begins.

The standards for software life cycle metrics have been articulated in the 1993 IEEE document for software quality metrics. These standards specify steps for developing the software metrics as: specifiy software-quality requirements, identify software-quality metrics, implement the metrics, analyze the results, and validate the software-quality metrics. The constructs for software quality are portability, reliability, efficiency, human engineering, testability, understandability, and modifiability (see also work of Curtis and Schulmeyer). The last three factors could be considered the maintainability of the software.

Metrics could be categorized in two dimensions: organization-customers versus processes-results, thus generating four types of metrics: (1) organization-process focus, (2) customer-process focus, (3) organization-result focus, and (4) customer-result focus. Zahedi discusses in detail metrics related to these four categories for design, reliability, implementation, and operations of information systems as well as for the quality of team management.

1. Analysis of Quality Metrics

Statistical quality control (SQC) and control charts are the main methods for analyzing quality metrics. In SQC, the observed values of quality metrics, like most business data, have random elements. Therefore, the deviation of quality metrics from their target values could have two sources: (a) random or chance elements that have the same probability distribution for all observed values of the quality metric (common cause), and (b) a shift in performance, which causes the metric to deviate from its target value (special cause). Statistical analysis offers the capability of distinguishing between the two causes and to identify the early signals for a change of conditions. Two types of statistics are commonly computed for summarizing data: measures of central tendency (mean, median, or mode) and measures of dispersion or variation (standard deviation, range, or mean-absolute deviation). For a given sample, the common test in SQC is to test the hypothesis that the metric's mean (or any other statistic) has deviated from its desired level.

Control charts graphically show the SQC test. Control charts could be categorized into Shewhart type and non-Shewhart type. A Shewhart-type control chart has a central line that indicates the target value of the metric, with two warning lines (one upper and one lower with the central line in the middle), and two lines farther out (one upper and one lower) showing control limits. There are a number of Shewhart-type control charts, which depend on the way the sample

statistics of the metric are computed. Examples of Shewhart control charts are *x* chart, also called *c* chart or *np* chart, (these charts track the cumulative number of failures in a sample of *n* binary-valued metric), *p* chart (tracks the ratio of failures), *u* chart (tracks number of events in an interval), *x-bar* chart (tracks the sample mean), *m* chart (tracks the sample median), and *r* and *s* chart (track sample variability). Non-Shewhart charts require more complex methods of establishing control limits. Examples of this type are MOSUM (moving sum of sample statistics), EWMA (exponentially weighted moving average), CUSUM (cumulative sum), and modified Shewhart charts (see Zahedi for a brief description of each).

The purpose of control charts is to discover variations caused by special circumstances and to take appropriate actions. The data on quality metrics should be collected regularly and the sample statistics (mostly the mean) are plotted on the control charts. Any regular pattern or movement outside the warning lines and control limits are signals for activating the Shewhart-Deming cycle of continuous-improvement actions.

VI. ASSESSMENT OF IMPLEMENTING TQM IN IS AND SOFTWARE SYSTEMS

Although the idea of using TQM in information systems is relatively recent, there are already published case studies, anecdotal evidence, and empirical research to support the efficacy of using the quality approach in information systems. In a case study of U.S. West Technologies implementing TQM in its software and information systems, Arthur discusses improvements and lessons learned. He reports significant improvement in terms of drastic reductions in outages as well as in the billing systems. He summarizes the results of using TQM in terms of significant gains in speed, quality, and costs. He also reports on lessons learned in this implementation of QIS as:

1. Attention to the entire information system—people, processes, hardware, applications software, operating system, and environment
2. Focusing on the few contributing factors responsible for the bulk of the problem
3. Focusing on high-level goals and results
4. Strong sponsorship, guidance, and support from senior leaders
5. Involvement of skilled improvement leaders, beginning with consultants and developing internal leaders
6. Rapid, outcome-oriented approach to problem solving with maximum results and minimum resources
7. Stabilizing, sustaining, and replicating the successful improvements

Flynn and colleagues show how various dimensions of quality management impact quality performance in terms of the competitive advantage of the firm. Ravichandran and Rai have found that an integrative, comprehensive strategy to quality management plays a pivotal role in enhancing the quality performance of information systems.

VII. QUALITY AWARDS AND STANDARDS

Organizations fuel and gauge their quality programs by competing for quality awards or implementing quality standards. Both approaches allow the firms to receive an objective evaluation of their quality initiatives and identify the ways they can improve their strategic and operational decision process. Furthermore, when successful in an award competition or in acquiring the standard certification, firms use their success for enhancing their image and expanding their market.

1. Quality Awards

After the Japanese success with TQM, various agencies have taken initiatives to encourage and promote quality. Among them are the Deming Prize, the European Quality Award, and the Baldrige Award.

The Deming Prize is a Japanese award, instituted in 1951 by the Japanese Union of Scientists and Engineers (JUSE) for companies dedicated to quality. This prize has an extensive application process. The European Quality Award was created in 1988 by 14 European companies, which formed the European Foundation for Quality Management. This award is the European equivalent of the Baldrige Award in the United States.

The Baldrige Award was created in the United States in 1987, and named after Malcolm Baldrige, the Reagan Administration's Commerce Secretary. The purpose of this annual award is to raise American industry's awareness of the significance of quality in the global market. Every year, the criteria for the award are updated and published. The criteria document for year 2000 by the Baldrige National Quality Program discusses the set of core values and concepts used for developing the quality criteria. This set includes visionary leadership, customer focus, organiza-

tional and personal learning, valuing employees and partners, agility, managing for innovation, management by fact, public responsibility and citizenship, focus on results and creating values, and having a system perspective (managing the organization as a whole as well as its components through synthesis and alignment). Based on these core values and concepts, the award-program committee has developed seven categories of quality criteria, which embody most of the TQM strategies discussed in this entry. The criteria categories are leadership, strategic planning, customer and market focus, information and analysis, human-resource focus, process management, and business results. Recognizing the unique features of educational and health organizations, the program committee has also developed criteria for education and health care organizations.

2. Quality Standards

Quality standards were developed before the popularity of TQM. The U.S. Airforce in World War II recognized the need for quality standards, and initiated quality assurance programs that led to the military quality-assurance standard MIL-Q-9858, later revised as MIL-Q-9858A in 1963. This standard was adopted by NATO in 1968 (AQAP-1), its European version (DEF STAN 05-21), created in 1970, was the basis of Britain's standards (BS 5750).

Europe's standard-setting body (The European Committee for Standardization) commissioned a private organization (International Organization Standards; ISO) to develop quality-assurance standards. The ISO used the previously developed United States and NATO standards to develop the ISO 9000 series, which was completed in 1987, and was revised in 1994 and 2000.

ISO 9000:1994 consisted of a series of quality-assurance guidelines for companies engaged in different phases and types of production of goods and services. ISO 9000 is a general guideline. ISO 9001 provides guidelines for companies engaged in design, development, production, installation, and servicing functions. ISO 9002 has guidelines for companies engaged in production and installation functions only. ISO 9003 is developed for companies engaged in final inspection and testing functions. ISO 9004 describes the elements of a quality-management system. ISO 9000-2 is a guideline for selected service industries, and ISO 9000-3 provides guidelines for applying ISO 9001 to the software industry.

In the late 1990s, a major revision effort was undertaken to update ISO 9000, resulting in ISO 9000:2000 series (see reports by West and colleagues and Cianfrani). The goals for the new revisions were to be usable by all sizes of organizations and all types of industries and sectors, to focus on business processes, and to be simple and clear. The ISO 9000:2000 series has moved away from the manufacturing focus, has removed the ISO 9002 and ISO 9003 series, and has revised ISO 9001 and ISO 9004 extensively. The standards have clearly stated management principles: customer focus, leadership, involvement of people, process approach, system approach to management, continuous improvement, factual approach to decision making, and mutually beneficial supplier relationships. The ISO 9001:2000 standards specify quality systems for internal use, certification, and contractual purposes, and ISO 9004:2000 provides more extensive guidance for quality improvement and is not intended for certification or contractual purposes.

Various countries have adopted ISO 9000 standards and given their own code names. For example, the British adoption of ISO 9000 is BS 5760; in the United States it is called ANSI/ASQC Q90; the European Community knows it as EN 29000, Australia as AS 3900, and Japan as JIS Z 9900. Brazil, Denmark, France, Germany, Portugal, and Spain also have their own code names for the ISO 9000 quality-management standards. In 1993, ISO established a committee to develop the ISO 14000 series for environmental management systems. The ISO 9000:2000 series has been made more compatible with the environment aspects of the ISO 14000 series. Furthermore, the TL 9000 series is being developed for the telecom products. This indicates an increased emergence of industry-focused standards or guideline documents.

In 1994, the United States auto manufacturers (Chrysler, Ford, and General Motors) combined their company-wide quality management standards under the Quality Systems Requirements QS 9000 standard, which is based on the ISO 9000 series, especially on ISO 9001 guidelines. The difference between ISO 9000 and the United States quality-management guidelines (such as Q90 or QS 9000) is that the ISO 9000 series is used for the certification process, which is required for dealing with European Common Market countries, while the United States guidelines are voluntary standards of quality assurance. With the continual revisions of standards and participation of the United States delegation in revising the ISO 9000 series, one may expect the differences between standards to disappear over time.

There are quality standards unique to information systems. The ISO/IEC 12207 standard for IT is specific

to IT and provides a common framework for developing and managing such systems. The Institute of Electrical and Electronics Engineers (IEEE) and the Electronic Industries Alliance (EIA) jointly have developed guidelines to foster better understanding of the standard as IEEE/EIA 12207.0-1996 for software and information systems life cycle processes. The document discusses various processes involved in the life cycle of systems. It includes the following quality management processes: quality assurance, verification, validation, joint review, audit, and problem resolution (IEEE/EAI Standard 1998). This standard is helpful for developing a formalized structure for QIS.

VIII. SUMMARY AND CONCLUSION

This entry presented QIS within the larger context of TQM. The definition of QIS is based on the definition of TQM and quality. The origin of QIS is closely tied to the origin of quality movement and contributions of its leaders and gurus. These leaders have created an extensive body of literature for offering practical advice on how to implement TQM, which in turn has become a foundation for systematic empirical studies of factors and strategies critical in the success of TQM and QIS. These factors include quality leadership, human resource strategies, quality controls and tools, quality design, process improvement strategies, customer focus, satisfaction and involvement strategy, supplier relationships and performance evaluation strategy, and benchmarking strategy. Furthermore, issues related to software quality assurance and data and information quality have long been of great concern in information systems, and should be integrated in QIS design. In monitoring and controlling the quality in information systems, metrics and assessment plan play a critical role. Moreover, quality awards and standards have increased the importance of QIS and provided help in creating QIS. The discussion of QIS topics in this article highlights the emergence of a body of systematic knowledge in creating QIS and the use of such systems in achieving quality goals within organizations.

SEE ALSO THE FOLLOWING ARTICLES

Benchmarking • Computer Assisted Systems Engineering • Control and Auditing • Corporate Planning • Human Resource Information Systems • Prototyping • Success Measures of Information Systems • Systems Implementation • Total Quality Management and Quality Control

BIBLIOGRAPHY

Adam, E., Hershauer, J., and Ruch, W. A. (1981). *Productivity and quality*. Englewood Cliffs, NJ: Prentice Hall.

Ahire, S. L., Golhar, D. Y., and Waller, M. A. (1996). Development and validation of TQM implementation constructs. *Decision Sciences,* Vol. 27, No. 1, 23–56.

Anderson, J. C., Rungtusanatham, M., and Schroeder, R. G. (1994). A theory of quality management underlying the Deming management method. *The Academy of Management Review,* Vol. 19, No. 3, 472–509.

Arthur, L. J. (1997). Quantum improvement in software quality. *Communications of the ACM,* Vol. 40, No. 6, 47–52.

Baldrige National Quality Program. (2000a). Criteria for performance excellence. Milwaukee, WI: American Society for Quality.

Baldrige National Quality Program. (2000b). Education criteria for performance excellence. Milwaukee, WI: American Society for Quality.

Baldrige National Quality Program. (2000c). Health care criteria for performance excellence. Milwaukee, WI: American Society for Quality.

Black, S. A., and Porter, L. J. (1996). Identification of the critical factors of TQM. *Decision Sciences,* Vol. 27, No. 1, 1–21.

Brassard, Michael. (1984). *The memory jogger*. Methuen, MA: GOAL/QPC.

Brassard, Michael. (1994). *The memory jogger II*. Methuen, MA: GOAL/QPC.

Camp, R. C. (1989). *Benchmarking: The search for industry best practices that leads to superior performance*. Milwaukee, WI: Quality Press.

Cianfrani, C. A., Tsiakals, J. J., and West, J. (2000). *ISO 9001:2000 explained*. Milwaukee, WI: ASQ Quality Press.

Crosby, P. B. (1979). *Quality is free*. New York: New American Library.

Crosby, P. B. (1984). *Quality without tears: The art of hassle-free management*. New York: McGraw-Hill.

Curtis, B. (1981). The measurement of software quality and complexity. *Software Metrics*. A. J. Perlis, F. G. Sayward, and M. Shaw (eds.). Cambrige MA: MIT Press.

Cyert, R. M., and March, J. G. (1963). *A behavioral theory of the firm*. Englewood Cliffs, NJ: Prentice Hall.

Deming, W. E. (1986). *Out of the crisis*. Cambridge, MA: MIT Press.

Deming, W. E. (1994). *The new economics: for industry, government, education*. 2nd Edition. Cambridge, MA: MIT Press.

Denton, D. K. (1991). Lesson on competitiveness: Motorola's approach. *Production and Inventory Management Journal*, Vol. 32, No. 3, 22–35.

Dion, R. (1993). Process improvement and corporate balance sheet. *IEEE Software,* Vol. 10, No. 4, 28–35.

Ebrahimpour, M. (1985). An examination of quality management in Japan: Implications for management in the United States. *Journal of Operations Management,* Vol. 5, No. 4, 419–431.

Ebrahimpour, M., and Withers, B. E. (1992). Employee involvement in quality improvement: A comparison of American and Japanese manufacturing firms operating in the U. S. *IEEE Transactions on Engineering Management,* Vol. 39, No. 2, 142–148.

Feigenbaum, A. V. (1956). Total quality control. *Harvard Business Review,* Vol. 34, No. 6, 93–101.

Feigenbaum, A. V. (1991). *Total quality control,* Third Edition. (First Edition in 1951). New York: McGraw-Hill.

Ferguson, J., Cooper, J., Falat, M., Fisher, M., Guido, A., Marciniak, J., Matejceck, J., and Webster, R. (1996). Software acquisition capability maturity model (SA-CMM) Version 1.01. Technical Report CMU/SEI-96-TR-020 ESC-TR-96-020, http://www.sei.cmu.edu/pub/.

Flynn, B. B., Schroeder, R. G., and Sakakibara, S. (1995). The impact of quality management practices on performance and competitive advantage. *Decision Sciences,* Vol. 26, No. 5, 659–691.

Forsha, H. I. (1995). *Show me: The complete guide to storyboarding and problem solving.* Milwaukee, WI: ASQC Quality Press.

Fox, C., and Frakes, W. (1997). The quality approach: Is it delivering? *Communications of the ACM,* Vol. 40, No. 6, 25–29.

Garvin, D. A. (1983). Quality on the line. *Harvard Business Review,* Vol. 61, No. 5, 65–75.

Garvin, D. A. (1984). Japanese quality management. *Columbia Journal of World Business,* Vol. 19, No. 3, 3–12.

Goel, A. A. (1985). Software reliability models: Assumptions, limitations, and applicability. *IEEE Transactions on Software Engineering,* Vol. SE-11, No. 12. 1411–1423.

Hammer, M., and Champy, J. (1993). *Reengineering the corporation,* New York: Harper Business.

Harter, D. E., Krishnan, M. S., and Slaughter, S. (2000). Effect of process maturity on quality, cycle time, and effort in software product development. *Management Science,* Vol. 46, No. 4, 451–466.

Helmer, O., and Rescher, N. (1950). On the epistemology of the inexact science. *Management Science,* Vol. 6, 25–52.

Herbsleb, J., Zubrow, D., Goldenson, D., Hayes, W., and Paulk, M. (1997). Software quality and the capability maturity model. *Communications of the ACM,* Vol. 40, No. 6, 31–40.

Hollenbach, R. Y., Pflugard, A., and Smith, D. (1997). Combining quality and software improvement. *Communications of the ACM,* Vol. 40, No. 6, 41–45.

Humphrey, W. S., Snyder, T. R., and Willis, R. R. (1991). Software process improvement at Hughes Aircraft. *IEEE Software,* Vol. 8, No. 4, 11–23.

IEEE. (1993). *Standards for a software quality metrics methdology P-1061.* New York: Institute of Electrical and Electronics Engineers.

IEE/EAI Standard. (1998). *IEE/EAI 12207.0-1996: Industry implementation of international standard ISO/IEC 12207: 1995 standard for information technology: software life cycle processes.* New York: The Institute of Electrical and Electronics Engineers.

IEEE Computer Society: Standard Coordinating Committee. (1991). *IEEE standard glossary of software engineering terminology, IEEE-STD-610.12-1990.* New York: The Institute of Electrical and Electronics Engineers. New York, NY

Imai, M. (1986). *Kaizen: The key to Japan's competitive success.* New York: McGraw-Hill.

Ishikawa, K. (1976). *Guide to quality control.* Asian Productivity Organization, Tokyo.

Ives, B., and Olson, B. (1984). User involvement and MIS success: A review of research. *Management Science,* Vol. 30, No. 5, 586–603.

Ives, B., Olson, M., and Baroudi, J. (1983). The measurement of user information satisfaction. *Communications of ACM,* Vol. 26, No. 10, 785–793.

Juran, J. M. (1978). Japanese and Western quality: A contrast in methods and results. *Management Review,* Vol. 67, No. 11, 27–45.

Juran, J. M. (1981a). Product quality: A prescription for the West, Part I. *Management Review,* Vol. 70, No. 6, 8–14.

Juran, J. M. (1981b). Product quality: A prescription for the West, Part II. *Management Review,* Vol. 70, No. 7, 57–61.

Juran, J. M. (1986). *Quality control handbook.* New York: McGraw-Hill.

Juran. J. M. (1992). *Juran on quality by design.* New York: The Free Press.

Juran, J. M., and Gryna, F. M. (1980). *Quality planning and analysis.* New York: McGraw-Hill.

Kettinger, W. J., Teng, J. T. C., and Guha, S. (1997). Business process change: A study of methodologies, techniques, and tools. *MIS Quarterly,* Vol. 21, No. 1, 55–80.

Krishnan, M. S., Kriebel, C. H., Kekre, S., and Mukhopadyay, T. (2000). An empirical analysis of productivity and quality in software products. *Management Science,* Vol. 46, No. 6, 746–759.

Mondon, Y. (1982). *Toyota production system.* New York: American Institute of Industrial Engineers.

Musa, J. D. (1971). A theory of software reliability and its applications. *IEEE Transactions on Software Engineering,* Vol. SE-1, pp. 312–327.

Oliver, N. (1988). Employee commitment and total quality control. *International Journal of Quality and Reliability Management,* Vol. 7, No. 1, 21–29.

Orr, K. (1998). Data quality and systems theory. *Communications of ACM,* Vol. 41, No. 2, 66–71.

Rai, A., Song, H., and Troutt, M. (1998). Software quality assurance: An analytical survey and research prioritization. *The Journal of Systems and Software,* Vol. 40, No. 1, 67–84.

Ravichandran, T., and Rai, A. (2000a). Quality management in systems development: An organizational system perspective. *MIS Quarterly,* Vol. 24, No. 3, 381–415.

Ravichandran, T., and Rai, A. (2000b). Total quality management in information systems development; Key constructs and relationships. *Journal of Management Information Systems,* Vol. 16, No. 3, 119–155.

Redman, T. G. (1998). The impact of poor data quality on the typical enterprise. *Communications of ACM,* Vol. 41, No. 2, 79–82.

Robey, D., and Farrow, D. (1982). User involvement in information systems development: A conflict model and empirical test. *Management Science,* Vol. 28, No. 1, 73–85.

Robinson, A. G., and Schroeder, D. M. (1990). The limited role of statistical quality control in a zero-defect environment. *Production and Inventory Management Journal,* Vol. 31, No. 3, 60–65.

Roethlisberger, F. J., and Dickson, W. J. (1939). *Management and the Worker.* Cambridge, MA: Harvard University Press.

Sahin, I., and Zahedi, F. M. (2000a). Control limit policies for warranty, maintenance, and upgrade of software systems. *IIE Transactions,* Vol. 31, No. 2, 146–158.

Sahin, I., and Zahedi, F. M. (2000b). Optimal policies under risk for changing software systems based on customer satisfaction. *European Journal of Operational Research.* Vol. 123, 175–194.

Saraph, J. V., Benson, G., and Schroeder, R. G. (1989). An instrument for measuring the critical success factors of quality management. *Decision Sciences,* Vol. 20, No. 4, 810–829.

Scherkenbach, W. W. (1986). *The Deming route to quality and productivity: Road maps and road blocks.* Washington, DC: Mercury Press.

Schmauch, C. H. (1994). *ISO 9000 for software developers.* Milwaukee, WI: ASQC Quality Press.

Shrednick, H. R., Shutt, R. J., and Weiss, M. (1992). Empowerment: Key to IS world-class quality. *MIS Quarterly,* 491–505.

Schulmeyer, G.G. (1999). Software quality assurance metrics. *Handbook of software quality assurance,* 3rd Edition G. G. Schulmeyer, and J. I. McManus, (eds.). Upper Saddle River, NJ: Prentice Hall.

SIM Working Group. (1992). Focus on quality: Quality assessment and planning tools for IS. Chicago, IL: Society for Information Management.

Strong, D. M., Lee, Yang W., and Wang, R. Y. (1997). Data quality in context. *Communications of ACM,* Vol. 40, No. 5, 103–110.

Taguchi, G. (1989). *Taguchi methods.* American Supplier Institute, Tokyo, Japan.

Taguchi, G., and Clausing, D. (1990). Robust quality. *Harvard Business Review,* Vol. 68, No. 1, 65–75.

Taylor, F. W. (1947). *Scientific Management,* New York: Harper and Brothers.

U.S. Air Force, HQ AFSC/FMC. (1991). *The metrics handbook.* Andrews AFB, Washington D. C.

Wang, R. Y. (1998). A product perspective on total data quality management. *Communications of ACM,* Vol. 41, No. 2, 58–65.

West, J., Cianfrani, C. A., and Tsiakals, J. J. (1999). ISO 9000:2000: A Shift in focus. *Quality Progress,* 100–101.

West, J. Cianfrani, C. A., and Tsiakals, J. J. (2000). Quality management principles: Foundation of ISO 900:2000 family. *Quality Progress,* 113–115.

Woodall, J., Rebuck, D. K., and Voehl, F. (1997). *Total quality in information systems and technology,* Delray Beach, FL: St. Lucie Press.

Zahedi, F. M. (1995). *Quality information systems.* Danvers, MA: Boyd & Fraser.

Zahedi, F. M. (1997). Reliability metric for information systems based on customer requirements. *International Journal of Quality and Reliability,* Vol. 14, No. 8, 791–813.

Zahedi, F. M. (1999). Quality control. *Encyclopedia of Electrical and Electronic Engineering.* New York: John Wiley and Sons.

Zahedi, F. M., and Ashrafi, N. (1991). Software reliability allocation based on structure, utility, price, and cost. *IEEE Transactions on Software Engineering,* Vol. 17, No. 4, 345–356.

Zahedi, F. M., and Ashrafi, N. (1995). A decision framework for selecting software reliability models. *Journal of Information Technology Management,* Vol. VI, No. 4, 25–43.

Zultner, R. E. (1990). Software quality deployment: Applying QFD to software. *Second Symposium on QFD Transactions.* ASQC and GOAL/QPC, Novi, MI, 132–149.

Zultner, R. E. (1993). TQM for technical teams. *Communications of the ACM,* Vol. 36, No. 10, 79–91.

Reengineering for Business Processes

Vikram Sethi and Kevin Duffy
University of Texas, Arlington

GLOSSARY

business process A set of interrelated work activities characterized by specific inputs and value-added tasks that produce specific outputs.
business process reengineering A reorganization of business activities that results from questioning the status quo. It seeks to fulfill specific objectives and can lead to breakthrough improvement. It is often associated with significant cultural changes.
core competence An understanding of the products or services that the organization has obligated itself to provide for its customers and of those aspects to which the customer attributes values. It is the core product or strategy or the basis on which an organization competes.
strategic intent A goal toward which organizational energy will be focused, representing a target for all organizational units.

This article addresses the **BUSINESS PROCESS REENGINEERING** phenomenon prevalent in the business world during the 1980s and 1990s. This article examines the underlying premises of BPR, classifies BPR projects into types, provides theoretical support for BPR, and finally looks at BPR criticisms and shortcomings.

I. INTRODUCTION

> In Nature we encounter quite often the interruption of continuity.
>
> V. Finkel in *The Portrait of a Crack*

In this article we address the organizational desire to review whether assumptions that have guided work remain valid. From operational details to strategic direction, every element of organizational reality is being questioned. The ultimate objective is to achieve radical improvement in business performance that otherwise would be unattainable.

The sheer magnitude of the changes under consideration has prompted the use of terms such as "reengineering," "reinvention," "rebuilding," and "remaking" when referring to this process. These terms represent a language of change and indicate a desire by the corporate world to increase productivity and global competitiveness. Even though changes in culture, work habits, and incentive systems resulting from restructuring have quite often been chaotic, there is increasing confidence that reexamining organizations is the long term solution to profitability. There are many examples of altering current conditions to achieve breakthrough results. GE, IBM, Blue Cross, and Xerox are among the group of companies that have chosen to examine each activity in their organizations and ask: "If we were a new company, how would we operate?" By tracing information and work flow vertically and horizontally through the organization, these companies have found simple, elegant solutions to increase productivity. Older companies such as General Electric, AT&T, and ALCOA have moved away from rigid, bureaucratic, conventional operating models to new, innovative, and more responsive organizations. This cultural transformation has been part of a move in organizations to examine how

current operating conditions can be changed, not just improved.

This article seeks to examine the nature of this change, termed business process reengineering (BPR). The following working definition for BPR has been adopted here:

> Business process reengineering (BPR) is a reorganization of business activities that results from questioning the status quo. It seeks to fulfill specific objectives and can lead to breakthrough improvement. It is often associated with significant cultural changes.

There are five elements of this working definition that need to be stressed: (1) a focus on business processes, (2) a questioning of the status quo, (3) specific objectives, (4) breakthrough achievement, and (5) significant cultural change. Each of these topics will be addressed in this article.

A. A Focus on Business Processes

Formally, a *business process* can be defined as a set of interrelated work activities characterized by specific inputs and value-added tasks that produce specific outputs. Business processes consist of horizontal work flows that cut across several departments or functions. Production processes come into physical contact with the hardware or software that will be delivered to an external customer. Business processes are all service processes that support the production process.

Organizations are replete with business processes. It is estimated that 80% of these are repetitive. Processes are so prevalent in organizations that we rarely give any thought to them. A cursory examination will reveal the following common processes within organizations: order processing, billing, purchasing, shipping, receiving, inventory management, auditing, business planning, budget planning, and client acquisition.

The classical result of the proliferation of business processes is that it invariably leads to complexity, which in organizations manifests itself in increasingly intricate bureaucracies, product line proliferation, and what Kriegal in 1992 termed an *M&M* culture—memos and meetings—that shuns decision making. As organizational complexity (external environment and internal decision making) increases, operational groups within an organization tend toward isolation.

B. A Questioning of the Status Quo

It has been written, "People will do tomorrow what they did today because that is what they did yesterday." This is natural. So is the fact that today's practices become tomorrow's unquestioned truths and remain in effect forever, almost. That is how redundant work activities develop and add exponentially to a firm's overhead. Individual tasks or entire functions may be affected. Consider the following examples which, in 1992, DeBono called "the continuity of time sequence."

1. A large manufacturer required managers to sign six duplicate forms when authorizing a subordinate to spend money.
2. An insurance company required employees to fill out requisitions for all office supplies and get them approved by Purchasing.
3. The Ford Motor Company operated as many as 18 monstrous warehouses, some taking up 1 million square feet, to move parts from its 5000 suppliers to its 9000 repair shops.
4. A chemical company has an accounts payable system in which 550 million invoices are paid, 8 million computers pages are generated, and 2 million pages are microfilmed annually.

Procedures have a habit of arranging themselves into structures, institutions, and concepts and becoming immortal. Breaking away from historical precedent requires conscious effort and the will to ask questions about what exists and why it exists. Such an analysis determines if a lock-in effect is present. (A lock-in occurs when an action is fixed to such an extent that it continues despite its shortcomings.) In 1989, Arthur provided a classical example of a lock-in, that of the QWERTY typewriter keyboard. In 1873, an engineer named Christopher Scholes designed the QWERTY typewriter keyboard, specifically to slow typists down because contemporary typewriting machines tended to jam if the typist went too fast. Today, the QWERTY keyboard is the standard for almost all keyboards even though the problem of jamming no longer exists. The design is locked in forever despite the availability of better alternatives.

C. Specific Objectives

Reengineering is typically a reverse-design process in that it begins with a desired outcome and works backwards to create the value-chain elements required to meet that outcome. Consider the case of Heinz's Pet Products. Faced with a drop in market share from 23 to 15%, the Pet Products Division decided to reengineer its business. Starting with a price that consumers wanted to pay—between 25 and 30 cents per 5 1/2 oz

can—the company reconfigured its process to achieve that target. This process—called backward chaining—begins by specifying the outputs of an ideal process and concludes by identifying the inputs needed to achieve these objectives.

The specific triggers for undertaking business process reengineering vary with each organization and its environment. In a 1992 survey of 200 public and privately held companies in the Midwest, Price Waterhouse found that customer demands and the need to create competitive processes were the overwhelming change drivers for business process reengineering. The most frequent operational objectives were to improve the quality and speed of customer service and to enhance customer and supplier relationships. Specific objective-driven projects under each strategic goal perform two main tasks. First, they ensure that the BPR effort remains focused, and second, they provide measurement benchmarks for the process.

D. Breakthrough Results

BPR targets, and can achieve, significant results. The characteristics described above—a focus on processes, a questioning of the status quo, and setting specific objectives—allow for this outcome. Organizational systems have an inherent slack built into them. Sometimes this slack can be quite significant and can occur as excessive inventory, production capability, manpower, or working capital. A part of the slack also finances the nonvalue adding activities that characterize business processes. When the actual work time in producing an output is 90 minutes in a process that takes 30 days, the balance is financed by organizational slack. Besides driving up the cost of capital, slack sets an upper limit to organizational maneuverability and growth.

In 1993, Allen cited GTE as a case in point. GTE Telops is the largest unit of GTE with $16 billion in revenues, providing local calls and short range toll calls and access to long distance carriers. It serves 10 million households and 1 million businesses into 40 states. Due to deregulation, top management realized that they needed to transform a protected bureaucracy to an information communication company capable of competing with the country's best data and communications services providers. The company opted to undertake radical change and examine every aspect of its business functions. Based on 1000 interviews and 10,000 on-the-job observations, the company discovered that administrative burdens put on management by its bureaucracy were reducing productivity by as much as 50%. In order to remedy this situation, GTE Telops has started on a program to identify best practices, create conceptual platforms for new processes, and integrate functions into a value-added path to ensure a complete focus on the customer.

E. Significant Cultural Change

Radical business change has a radical impact on people. BPR is as much social engineering as it is work engineering. BPR typically involves eliminating, consolidating, and otherwise altering work activities. Quite often, BPR is associated with layoffs and early retirement programs. Several reengineering experts have tried to distance the BPR concept from staff reductions; however, the reality is quite different. In the current competitive economy, the corporate world has taken the idea of downsizing to heart. Whether due to reengineering or otherwise, staff reductions abound. In the United States in the period 1990–1994, layoffs steadily increased. Several of the companies associated with BPR programs appeared on a 1994 listing of the 25 U.S. companies with the largest announced staff reductions. It is difficult to conclude whether the layoffs are occurring as a result of BPR or in conjunction with BPR as a result of business conditions. In either case, the magnitude of these cutbacks is unsettling.

The extent of these cutbacks creates an environment of extreme uncertainty, especially for employees whose company is undertaking reengineering. The spurt in layoffs, whether called reengineering, downsizing, or rightsizing, created an unprecedented challenge for human resource departments. The rapidity and magnitude of change or even the fear of change was undermining the morale and effectiveness of employees. The implication is that the BPR effort is (in reality or in perception) an insecure experience characterized by fear and uncertainty that can leave a lasting impact on employees.

Work rationalization is also closely associated with a very new and very different employer–employee contract. This contract now involves (1) no job security,(2) an employee's responsibility to learn new skills, (3) increasing expectations of innovative work, and (4) simply doing more with less. The employment-for-life approach is a philosophy of the past. At the height of reengineering, the average tenure of employees continued to decrease, and job satisfaction declined. What is the cost of employee resentment and sense of betrayal? Can the benefits of BPR outweigh these costs in the long run? These questions prompt a greater emphasis on the human and cultural aspects of BPR.

II. A CLASSIFICATION OF BPR PROJECTS

BPR projects do not necessarily involve the entire organization. They may be limited to specific processes or departments. Which processes have been the targets of BPR? A Price Waterhouse study developed a list of the four top processes being reengineered in manufacturing and service firms. For manufacturing firms, the list includes Customer Order Management, Customer Service, Financial Management, and Production Management. Alternately, the list of processes within service firms is comprised of Production Management, Financial Management, Information Services, and Human Resources.

BPR efforts may take several forms. They may affect a single function or an entire value chain. In addition, changes introduced may be limited or extensive. These characteristics of BPR projects are referred to as their *scope* (narrow vs broad) and their *scale* (limited vs extensive). A categorization of BPR projects based on these characteristics results in four types of BPR projects: functional refinement, functional integration, process redesign, and business redefinition.

Functional refinement refers to work improvements limited to a single department or function where the changes involved are not major. However, there is some activity reorganization to provide desired benefits. Consider a Midwest manufacturer of large water pumps that was looking to add capacity to its manufacturing operations. In examining work flow on the shop floor, it found that, due to the existing machine configuration, each pump traveled 30 miles on the floor before being completed. This limited plant capacity to 45 pumps/day. By a reorganization of work centers, the capacity was increased to 65 pumps/day.

BPR efforts in this category also include the use of information technology to leverage current operations. The imaging system installed by the United Services Automobile Association (USAA) in 1987 permits customer service representatives to answer a query while the customer is still on the phone, because it gives the 2000 employees access to 2 million files (nearly 2 billion pages).

Functional integration describes efforts of organizations that have undertaken an integration of work activities across departments to form "service cells." One example of this scenario is provided by Mutual Benefit Life, which decided to reengineer its application process to attain the strategic target of 60% productivity improvement set by its president. The traditional step-by-step process of handling applications—and the job definitions and departments that supported it—was tossed out. In its place, a new position called a *case manager* was created. This position was made responsible for an application from initiation to policy issuance. Using a PC linked to the company's main computer, the case manager was given the tools to perform all tasks needed to process the application from start to finish. The company's capacity to handle applications doubled. It was able to eliminate about 100 positions and reduce the average turnaround time for applications from 5 to 25 days to 2 to 5 days.

Process redesign projects examine the entire value chain to see where improvement can occur.

Northern Telecom (NT), for example, took a systemic view of its business to analyze whether its current set of processes is capable of taking the organization to its "vision" state. To ensure that this objective is reached, the company started with a set of customer-stated objectives and implemented a new order flow process to decrease the time from customer request to final bill by 71%. This project, "Project Chrysalis," envisioned a horizontal organization and involved an examination of all disconnects in the order fulfillment process. By eliminating these disconnects, the process will ensure that customer demands are built into the very process that produces the outputs. Future plans propose extending the benefits generated to the global marketing, production, and engineering processes.

Business redefinition builds on process redesign but with a fundamental difference. Before examining system processes, BPR reviews whether the organization and its links with customers, suppliers, and other alliance partners are sufficient to provide it with the cooperative advantage it needs to succeed. Are we doing everything that is expected of us and to the best possible standards? These questions form the basis of redefining the business and its processes.

A. The BPR Process

One framework for reengineering divides BPR into four components: business drivers of reengineering, the reengineering process, reengineering facilitators, and the reengineering end product. A discussion of each component follows.

1. Business Drivers of Reengineering

Three key factors often lead a firm to consider business reengineering: a competitive environment, recent internal changes, and a strong IT infrastructure. The fact that the general business environment is becoming more competitive is not new. Increasing glob-

alization of markets is forcing firms to be more innovative and efficient at the same time. In terms of business cycles, this implies speeding new products to the market and quickening response to customers in manufacturing and delivery. U.S. automakers take about 5 years to design and produce a new vehicle while Japanese manufacturers are closer to 3 years. This means the Japanese manufacturers touch the market one more time in a typical product cycle. This can lead to significant market advantages and, when coupled with efficient delivery and service, long-lasting competitive advantage. As the general environment becomes more competitive, there is greater pressure on organizations to improve internal processes.

A second business driver is the recent changes in a firm's internal environment. Downsizing, layoffs, and plant closings have been very visible in the 1980s. Most such steps are permanent. The issue for the organization is then how to best conduct business with 20 to 30% fewer employees and reduced budgets. These internal stresses cause organizations to reevaluate business functions. Finally, a strong IT infrastructure provides an added thrust to a firm's reengineering efforts. Although reengineering is not a technology endeavor but a business problem, IT occupies a central role as the enabler of cross-functional business processes. Most reengineered processes are enabled by IT. Executives will need to think creatively about how technology can remove steps, people, time, money, and inefficient organizational structures. This, however, requires the presence of a strong IT culture in an environment and top management which is comfortable with the use of IT.

B. The Business Reengineering Process

The reengineering effort begins by articulating the strategic intent of the firm and defining its core competencies. Strategic intent is defined in the management literature as the firm's desired leadership position in its environment. The concept also encompasses an active management program that includes focusing the organization's attention on the essence of winning, motivating people by communicating the objective, leaving room for individual and team contributions, sustaining enthusiasm by providing new operational definitions in changing circumstances, and using the intent consistently to guide resource allocation.

Traditional strategy literature has often defined a future objective for the firm: "Where do we want to be?" *Strategic intent* or *strategic ambition* is a resolution on the part of the organization to achieve this end goal. Strategic intent represents a goal toward which organizational energy will be focused and represents a target for all organizational units.

A related concept is that of the *core competence* of the firm. Core competency entails an understanding of the products or services that the organization has obligated itself to provide for its customers and of those aspects to which the customer attributes values. It is the core product, strategy, or basis on which an organization competes. For example, Canon specializes in producing desktop laser printer "engines" and holds an 84% share of the world market. Defining the strategic intent and core competencies establishes a target for the firm and identifies the capability of the firm that will be utilized to achieve that target. The next step is describing how the core products of an organization will be delivered to the market and the functional processes within the firm that will support this transfer. For example, if a hospital believed its strength lay in fast and efficient care, its primary process would be "patient delivery." Process definition is followed by activity modeling. *Activity modeling* helps to identify the key activities in a business process, regardless of where they are performed. Further, activity modeling also identifies the true cost of, and the value added by, each activity and the process as a whole. Northern Telecom, when reengineering its core processes, analyzed order flow at the organizational level: How do orders flow into Northern Telecom, between functions within the organization, and back to the customers? Similarly, in the case of the hypothetical example of a hospital targeting the patient delivery process, all external points of contact with the patient would be documented. For example, how many different ways do patients come into the hospital (walk-ins, by appointment, etc.)? How are patients delivered through the hospital functions?, etc. The flow charting of activities helps in identifying "disconnects" or missing, redundant, unconnected, or non-value-added interactions. Another type of analysis that is useful at this stage is what numerous researchers have named the "fishbone analysis" or "root cause analysis." This examination searches for the most basic causal factors, which, if corrected or removed, will prevent the recurrence of a problem. For example, in the case of the hospital's patient delivery process, problems may have accrued in that patients had to wait too long for treatment or examination. The immediate cause may be the unavailability of the physician or the support staff, but a root cause analysis may reveal a "load-balancing" problem and the need for a better matching of patients with medical

resources or simply a bottleneck in a particular function. The end result of activity analysis and root cause analysis should be a refined process—an improved method of performance. Continuing with the example developed above, the refined process might envision that the patient go through 5 care centers and be contacted by 10 employees instead of 10 centers and 60 employees. The refined process is therefore simpler and more efficient.

The final step in process reengineering is that of activity benchmarking. *Benchmarking* sets a target improvement standard for a process. The standard is determined by identifying industry best practices and using them as improvement targets. The search for best practices helps in setting standards for activities.

C. Reengineering Facilitators

Several facilitators of reengineering have been identified in the literature. The most important is the effective management of change, particularly with respect to people. As mentioned above, reengineering often involves learning new skills and responsibilities and establishing appropriate measurement systems and control mechanisms. Education and training of employees are thus important aspects of the reengineering process. Authors such as Ryan proposed, in 1992, a sequence of four steps to gain employee acceptance: (1) awareness—helping the employee define the nature of change; (2) education—describing details of how the nature of the work will change; (3) testing—helping employees assess for themselves how the changed work will differ from the work currently done; and (4) understanding—acceptance by the employees of change.

D. The Reengineering End Product

The final process must be judged using two criteria:

1. Has it met the goals of the reengineering effort?
2. Is the process flexible enough to accommodate future changes?

An assessment of the redesigned process using the first criterion should ensure that the process now reflects a firm's strategic intent and core competencies and meets the standards set during the benchmarking exercise. However, as is discussed in the literature, a critical goal of the reengineering effort is the design of an organizational infrastructure that is open to improvement. Reengineering is not an everyday exercise. Once a new system is in place, it should have the capability of responding quickly to changes in the business environment.

III. THE ROLE OF IT IN REENGINEERING

The role of IT in the business reengineering exercise lies in its ability to enable the redesigned process. For example, in the case of the hospital which wanted a patient to receive care from only 10 employees instead of 60, IT could provide a bedside terminal to help give caregivers access to the ordering of and the end results of tests. Davenport described in 1993 different types of impacts that IT can have on process redesign. These are:

1. Automational—Eliminating human labor from a process
2. Informational—Capturing process information for purposes of understanding
3. Sequential—Changing process sequence or enabling parallelism
4. Tracking—Closely monitoring process status and objects
5. Analytical—Improving analysis of information and decision making
6. Geographical—Coordinating processes across distances
7. Integrative—Coordination between tasks and processes
8. Intellectual—Capturing and distributing intellectual assets
9. Disintermediating—Eliminating intermediaries from a process

All reengineering cases reviewed for this study reveal a significant contribution of IT to the new process. The examples indicate that IT has affected two sets of business functions. It has changed the nature of the front end value chain functions (consumer-oriented business processes) through "case management" techniques and affected the back end value chain functions (production-oriented business processes) through "cycle management" techniques.

A. Consumer-Oriented Processes and "Case Management" Techniques

An examination of successful reengineering efforts reveals that the most common impact of IT has been

integrative: coordination of tasks and processes. For example, the creation of case management positions in organizations demonstrates the move away from sequential activities being performed by different individuals to a single point person who performs the entire process. Davenport and Nohria, in 1994, described several examples of the integration of labor through the use of case management techniques. At Pacific Bell, providing a business customer with Centrex telephone service once took 11 employees and more than 5 business days. Service representatives had to update at least 9 computer systems with the correct information, making frequent errors and consulting customers several times. Now, with a redesigned process for provisioning, Centrex service coordinators handle all contact with customers. Using a computer workstation that interfaces with all 9 systems, they provide the customer's service in no more than 2.3 days; and if the order is for a type of service predefined in network software (as in 80% of orders), is usually done the day of the customer request.

Several researchers agree that these types of techniques are especially prominent in service organizations in which the emphasis of IT is on improving service quality, increasing service satisfaction, and improving employee coordination to increase the speed of delivery.

B. Production-Oriented Processes and "Cycle Management" Techniques

IT also impacts back end value chain functions by focusing on improving cycle times in organizations. A comprehensive example of cycle time management through the use of IT is provided by a 1990 *Industry Week* article which details the reorganization undertaken by General Motors at its Pontiac East plant. General Motors started with a redesigned product. It completely reworked its pickup trucks to make them easier to build, then set out to use IT to support the manufacture of the redesigned truck. The result is an automated system—from customer order to production to delivery. The plant covers 2.4 million square feet, contains 25 miles of conveyor, uses 146 robots, and rolls out 960 trucks in 2 shifts each day. About 2200 parts are used in each truck; 22 hours are required to go from metal to rolling truck.

After a customer orders a truck, GM's dealer computer network routes the order to the Pontiac East plant. A computer reviews the model mix and places the order in a build sequence. The order is then routed to a plant production support system, which controls the process that actually builds the truck. The system, for example, cues the computer controlling the body shop to begin building the truck body. At the same time, it passes the order information to the paint shop computer, which organizes its own work. Other systems are linked to the production system controlling tasks such as testing electronic equipment, checking component calibration, and confirming the order data. Finally, all systems are linked to GM's accounting systems which keep track of cost, expenses, and dealer billings.

The focus of IT in cases such as this is on improving three types of cycle time: system, product/service, and project. *System cycle time* refers to the length of time it takes to push something through the system—for example, a new product design, a customer's order, an internal report, an engineering change, or an approval of an investment. *Product/service cycle time* is the time involved in the commercialization cycle, from conceptualization to implementation. Finally, *project cycle time* is the time taken to complete any activity—for example, a multiyear effort to develop new technology, construct a new factory, or introduce a new product.

IV. THE THEORETICAL BASIS OF BUSINESS PROCESS REENGINEERING

BPR has been a practiced field, its recent knowledge base having developed in consulting. Its conceptual roots, however, may be traced to several academic theories which have for long influenced the study of organizational transformation. Theoretical support for business process reengineering (BPR) stems from management and strategic management literature. Within this literature, research has examined the occurrence of change and turbulence within an organization or an industry, and the impact of change and turbulence upon the organization, its products, and its industry as a whole. It is this literature and, in particular, the theory of punctuated equilibrium, that this article now examines.

The *punctuated equilibrium model of organizational transformation* provides a theoretical framework for characterizing and investigating propositions in BPR. The framework describes organizations as evolving through relatively long periods of stability in their basic patterns of activity that are punctuated by relatively short bursts of fundamental change. Various studies in the field have focused on issues of technology management and executive leadership. Other studies have integrated the punctuated change

proposition with the clock-setting process, in which organizations enjoy increased freedom immediately following a revolution. Most changes occur in an incremental fashion, which is to say that one small change after another occurs. Throughout the management literature, change is seen as continuous. As an example, Meyer, Brooks, and Goes in 1990 discussed change as comparable to the difference between movement and acceleration. Movement is a constant. While in motion, a cyclist, for example, needs to constantly make minute steering adjustments to remain on track. These small adjustments serve to aid the cyclists balance and to keep the bicycle on course. The cyclist continues in a period of motion until environmental conditions necessitate the application of acceleration (the cyclist, for example, encounters hilly terrain which requires acceleration in order to keep the cycle from falling as well as to restrain its course). Thus, it is the magnitude of change which can catastrophically impact organizations.

During equilibrium periods, organizations and industries are relatively stable. No major changes occur, and one change builds upon another change as the capabilities of a product or an industry are extended in a steady manner. Equilibrium characterizes a period of steady improvement. Changes to existing products or management structures happen slowly and build upon one another.

During equilibrium, the status quo is maintained, and small change is continuous, with one change following another. In this fashion, product changes are seen as evolutionary, as products evolve from one form to another. Researchers have termed such periods as being periods of first-order change, where evolution is taking place, but the changes are small enough that they can be easily absorbed into the organization's adaptive capacities. Such changes may extend product features, or may eliminate features which have become obsolete, but the changes themselves are built upon, and are easily incorporated into, the dominant design for the product and/or management structure. In other words, the dominant design serves as the foundation for the product throughout periods of incremental change, rather than yielding to an alternative design or an alternative management structure.

Alternately, the period of equilibrium is suddenly interrupted by a large scale, massive change. This change can be almost catastrophic in comparison to the smaller incremental changes taking place during equilibrium. The impact of the change is to radically alter existing products or structures. At times, the change is so large as to render entire industries momentarily dormant while they watch, waiting for a dominant model to establish itself. Hence, the theory explains that the stability experienced during the period of equilibrium is punctuated, or broken, by the sudden catastrophic change. In contrast to the evolutionary change continuously taking place during periods of equilibrium, large, discontinuous changes are seen as comprising second order, or discontinuous change. In other words, the changes are so radical that they exhaust managers' abilities to incorporate the impacts into the organization's existing adaptive capacity. As such, they mark the advent of new products, new orders, and/or new structures.

The change is characterized as being momentous and huge. Structural orders may be overturned, and products suddenly veer off into new, unexpected directions as a result of these changes. The length of the period of change is unknown. Gradually, the change comes to an end as a dominant design for a product emerges, or a industry settles upon its management structures. Importantly, though, there is a period during which actors are uncertain as to which design shall emerge as dominant. During this time, there are rival designs competing against one another. At some point, though, one of the designs will triumph over the others, becoming the dominant design. The change can introduce either competence-enhancing discontinuities, or competence-destroying discontinuities. Competence enhancing discontinuities radically alter the existing structure, or the existing design of a product, yet ultimately, the new design is relatively easier for established organizations to assimilate into their production routines. This assimilation occurs because some companies naturally hold a leading position within their industry and have, over time, built a technological foundation able to withstand and adapt to the change. On the other hand, competence destroying discontinuities totally vanquish the existing order. A competence destroying discontinuity might be introduced by a new player to an industry. For example, the transition, within the airline industry, from propeller powered planes to jet aircraft was a competence destroying change. As the jet was distanced from the existing plane in function and design, much of the knowledge base within the industry no longer pertained to the jet as it did to the plane. While established organizations, in order to preserve their knowledge bases, may struggle to keep the existing design alive (a new player would be apt to escape the resulting switching costs), eventually a new technology will emerge as the dominant design.

The lesson for organizations, then, is to adopt, adapt, and assimilate, or face certain failure. However, whether competence enhancing or competence destroying, the change is abrupt and radical enough that it is viewed as revolutionary change, in stark contrast to the evolutionary change usually seen as products evolve from one design to another, incorporating with them small, steady, continuous changes along the way.

Once the change results in the selection of a dominant design, the period of change is seen as ending, giving way to another period of equilibrium. This cycle repeats, although on an uncertain basis. One cannot predict the length of an equilibrium period. Likewise, one could not ascertain in advance how long the period of change will last, or how long it will take for a technology to finally emerge as the dominant design. Thus, a model depicting the cycle of "equilibrium to change to equilibrium" can be seen as a line with a series of spikes, where the spikes denote the catastrophic changes punctuating, or breaking, the relatively stable periods of equilibrium. In applying the theory to organizations, researchers have noted that large changes are most apt to be introduced during periods of relative decline. He notes that during periods of prosperity, organizations view themselves as meeting customer demands. In other words, an assumption would be that the organization is doing things correctly, and in order to continue prosperously, it wishes to continue doing these things in a fashion which has resulted in successful performance. During periods of decline, however, the organization may feel more greatly the need to introduce technological or other types of change to once again return to a profitable state.

The theory has been empirically tested within the health care industry. Researchers undertook a longitudinal study, collecting data from the 1960s, through the 1970s, and into the 1980s. During the 1960s and early 1970s, the health care industry was seen as stable. Incremental changes were introduced as technology expanded to allow new or modified methods of responding to patients. By and large, though, the industry remained stable and grew during this period. Relationships with individual physicians remained on the tried and true pattern which had been established over time.

A decade later, however, the situation was remarkably different. Health care costs were skyrocketing, and while medical technology advanced, so, too, did the cost of investigating alternative medical procedures. Hospitals merged or were acquired by other hospitals. Importantly, the relationships between physicians and hospitals changed throughout the health care industry. Once these changes had been assimilated into the industry, however, the health care industry once again returned to a period of relative calm and stability. Note that the result of the catastrophic change was not a return to the old manner of conducting business, but rather the incorporation of the changes into a new organizational regime.

Punctuated equilibrium, in summary, attempts to describe the situation in which organizations or industries, faced with differing magnitudes of change, must respond to changes. Changes which build upon existing knowledge bases are incorporated into product or organizational designs. Alternately, large-scale, massive change disrupts the existing order, necessitating a response of similar magnitude from the organization itself or the industry as a whole. In such a situation, the organization may need to greatly expand or possibly redefine its knowledge base. Change is continuous. Organizational response represents a means of adapting to change, whether through incorporation of change into an evolutionary model of conduct, or through overthrow and redefinition of the organization and its routines.

A simulation model for the theory was developed by Sastry in 1997. The model is built on organizational attributes of the level of inertia, strategic orientation, appropriateness of an organization's strategy, and competence in the execution of strategy.

When an organization is first formed, the *level of inertia* is low and strategic orientation set by the organization's founders provides the necessary direction. Inertia sets into the model due to social processes and the development of company culture over time. Once inertia begins to build, it becomes easier to increase, since more developed internal and external relationships provide a basis for their own further extension.

As the organization develops, its ability to change decreases. When inertia is high, organizational managers are less able to recognize and respond to the need for a change. Signals of poor performance go unnoticed and organizational members are slow to perceive discrepant signals. This is a reinforcing cycle where failure to respond allows inertia to build, further reducing the ability to change.

The remaining loops relate performance and organizational change through the key processes of convergence and reorientation. Early in the organization's life, performance may be low. The initial level of low inertia is associated with low competence, as the organization's members have had little opportunity to learn

by doing, and the structural and social dimensions of the organization are not yet developed. Depending on the organization, appropriateness, or the fit between the organization and its environment, may be high or low. If it is low—the organization comes into being with a strategic orientation that is not well-suited to its environment—the organization will perform poorly. Pressure to change may be generated as a result, in turn leading to organizational change. If this change is in the wrong direction or dimension, the organization's performance will remain low.

If, by contrast, at its founding the organization's strategic orientation is well matched to environmental requirements, then overall performance, a function of both appropriateness and competence, would reach a moderate level. If the level of performance is high enough to avoid pressure to change, there is no impetus for change, and, without a reorientation, strategic orientation remains at its initially chosen value. In the absence of change, the reinforcing loop operates to generate convergence: as the organization continues doing what it has been doing with increased competence, organizational members learn to perform their tasks more efficiently, and improvements in technology and work processes follow from experience. Higher levels of competence result in increased performance, which reduces pressure to change. As a result, the organization does not change its strategic orientation, and inertia-building processes continue, boosting competence and validating members beliefs that they are on the right course. Performance increases, as a result of higher competence, and pressure to change is further reduced.

As a result of shifts in the external environment that are exogenous to the mode, the required strategic orientation—defined as the orientation best suited to the environment—may change over time. Once the environment shifts, ever-increasing competence in an inappropriate strategic orientation no longer benefits the organization. Higher levels of inertia cause delay and increase the difficulty of change processes. When pressure for change has built up to a level high enough to overcome the effects of inertia, management relieves the pressure by changing the organization's strategic orientation. The negative loop thus ensures that a reduction in appropriateness, which causes a drop in performance, is eventually addressed by a change in strategic direction. This change may or may not improve performance, depending on the appropriateness of the new strategic orientation. When the shift is deleterious, change in strategic orientation no longer positively affects appropriateness, and the relationship switches to a negative one. The balancing loop turns into a reinforcing loop, as a result.

V. CAUTIONS IN THE PRACTICE OF BPR

There are two unmistakable conclusions about business process reengineering. One, it is an increasingly popular concept; and two, it has not produced the dramatic results it promised. Disenchantment therefore appears inevitable. There are two main reasons why BPR can fail to deliver on its promise. First, staff layoffs can become a centerpiece of the BPR strategy, leading to a lack of employee commitment for the BPR process. Second, by consistently arguing for the radical "overthrow" of current practices, BPR proponents may ignore organizational realities. The principles of BPR are fascinating and can provide a strong basis for organizational development. However, if BPR cannot unlink itself from layoffs and if it cannot incorporate incremental change in its theses, BPR is destined to failure.

There is no mistaking the spread of BPR. The popularity of BPR is understandable. Reengineering promises spectacular improvement in costs, quality, and customer service—targets no manager can ignore. Yet the early results from BPR are less than spectacular. According to one estimate, 85% of reengineering projects fail. In a 1994 report, the CSC Index Group found that fewer than half of the firms aiming at increases in market share through BPR were successful.

Watching these shortfalls in BPR targets, experts have argued that the principles of reengineering are flawed. An analysis of the trends in BPR and BPR implementations suggests that the problem may lie not in the BPR principle but in its practice. While the fundamental idea of reengineering is sound—improving current and future business processes—the manner of its implementation leaves a lot to be desired.

One major problem with reengineering is that it has developed a kinship with staff layoffs. The CSC Index group reported in its 1994 survey that each reengineering initiative led to 360 job losses in North America and 760 in Europe. Other estimates show larger numbers. Another study has shown that 18 of the 25 companies having the biggest downsizings since 1991 have been actively involved with BPR. These companies include IBM, AT&T, General Motors, USPS, Sears, and GTE. The size of cutbacks per firm ranges from 9000 to 85,000 since 1991. In total, in 1993, large U.S. firms announced nearly 600,000 layoffs—25% more than were announced in 1992 and nearly 10% above the levels of 1991, which was technically the bottom

of the recession in the U.S. Large layoffs are not surprising any more. The 1980s were a prime example of across-the-board reductions in personnel. However, the layoffs of the 1990s were qualitatively very different from those that occurred in the 1980s when mega mergers, subsequent restructuring, and defense cuts leading to plant closings accounted for the major share of job losses. Even so, some companies tried to avoid layoffs and found alternate ways to counter the downturn in business cycles. IBM cut overtime and restricted hiring; Delta reassigned pilots and flight attendants; and Hallmark and American Television & Communications relied on attrition and retraining. Mass layoffs were typically a last resort measure used by companies trying to avoid Chapter 11 bankruptcy.

This was not the situation in the 1990s. Layoffs are now considered as "the" first step to be taken in response to any business downturn—real or perceived. In fact, layoffs are now being presented by some as a deliberate strategy designed to allow a company to compete. There is no longer a stigma attached with mass layoffs. In fact, companies are using BPR to legitimize the use of layoffs as a business strategy. After all, BPR promises radical improvement and what better way exists to realize quick cost savings than to lay off employees? This is totally consistent with a need for instantaneous gratification.

There are, however, several problems with this strategy.

BPR will fail unless all employees are involved, committed, and excited about the new processes. Its success is a function of the extent to which all employees accept and incorporate the new processes in their work. When this does not happen, its failure is attributed to employee resistance—that universal characteristic of all human beings to resist change, especially radical change. Undoubtedly, there exist animosity and resistance on the part of employees toward BPR. However, it is critical to acknowledge the target of this resistance. It is not change that is being resisted but being laid off. In fact, the term "resistance to change" has become an overused metaphor to justify failure. Consider that, on average, middle managers find themselves out of a job two or more times in their careers. Most employees have participated in the cyclical nature of their industries and recognize that the need to adapt is vital to personal and organizational survival. In an environment of intense competition, declining profit margins, and declining employment opportunities, the term "employee resistance" is a self-serving excuse. Motivating half a firm's employees while releasing the other half is impossible to achieve. Also, while downsizing may allow a firm to reduce costs in the very short term, the real business problems that necessitated downsizing still remain. These problems (which could range from quality control, customer needs, target markets, skills, etc.) have little chance of being resolved in the environment of uncertainty and fear caused by layoffs. In fact, the short term gains due to downsizing may even lull an organization into believing that its "real" problems have been resolved. This is a dangerous conclusion.

Layoffs also risk creating a cyclical employment policy. It is difficult for an organization to define the right size that it needs to achieve. Invariably, layoffs are overdone and can seriously hurt the organization.

It may be argued that BPR, by clarifying business processes, allows a more stable employment policy because it defines the right organizational size. However, this argument focuses on current business processes only; and if current business processes are used as guidelines to downsize, then inevitably the resultant organization will have barely sufficient employees to handle current business functions. This can be termed "managing for today." Yet the focus of BPR is supposed to be "managing for the future," and downsizing clearly limits an organization's capability for future growth. Downsizing implies that a good portion of organizational slack—especially personnel—is overhead and should be removed. This streamlines the asset base and increases its efficiency. This improvement may come at the expense of increasing work hours and reduced job satisfaction. In addition, the reduced and overworked asset base can no longer provide a strong and innovative basis for future growth, and thus undermine the very thesis of BPR. A second problem with BPR is its inherent assumption that anything short of fundamental change is unacceptable. There are several problems with this assumption.

All organizations are heavily invested in the "present" and the manner in which they perceive that that "present" will evolve. Skills, technology, information systems, and capital investments provide the organizational infrastructure that represents the "present." While this base may not be optimally efficient, it is nevertheless the most realistic. It provides the necessary stability for any organization and supports its internal politics, history, and the set of normative agreements that make the organization work. Organizational learning and experience are embodied (e.g., about markets, products, and competition) and reflected in the present infrastructure that has been shaped by the cumulative outlook of the management team. To argue that all organizational heritage is worthless is equivalent to concluding that all organizational skills are

outdated, products/services unwanted, and management and worker skills wanting. (Recall that in 1992 when the then-Japanese Prime Minister Kiichi Miyazawa and Yoshio Sakarauchi, the Speaker of Japan's House of Representatives, made much milder statements about American industry, they created a furor in this country.) While some companies are in situations where the past is a burden, other organizations cannot and should not disclaim their legacies and destroy all accumulated learning.

While companies should not be prisoners of their past, there needs to be, however, complete responsibility and accountability for it. Social systems must not be treated as scientific experiments in that if one fails, the ingredients may be thrown out and another started. Finally, to examine change by either "revolution" or "reform" in a broader context, consider the following statement by Sir Karl Popper in response to the idea that society, as a whole, can only be reformed by radical transformation: "Our western democratic societies are very imperfect and in need of reform, but they are the best ever. Further reforms are imperative. But of all political ideas the wish to make man perfect and happy is perhaps the most dangerous. The attempt to realize heaven on earth has invariably produced hell."

BPR is a powerful tool that has provided a framework to incorporate diverse themes from the disciplines of TQM, systems engineering, industrial engineering, and work-flow analysis. It has also related these ideas to business strategy, industry structure, and competitive changes. This will always be the fundamental contribution of BPR. However, for these contributions to be realized, BPR must recognize that incremental change can be just as powerful as radical change. In addition, BPR must maintain its focus on process design and improvement as opposed to becoming one more excuse for mass layoffs.

VI. CONCLUSIONS

Reengineering practice represents a move toward organizational simplicity. The rationale for reengineering is that organizations can best be structured along service delivery lines. This allows an organization to deliver a complete, well-defined service to a specific market—a process which encourages common integrated systems and the more efficient management of these systems.

Examining the genealogy of reengineering, it becomes clear that the development of the reengineering concept represents a logical succession to the "quality movement." Quality management concepts have long been rooted in manufacturing operations even though in the last several years these principles have been extended to other environments as well. Quality improvement programs have been very effective in improving product quality and the manufacturing productivity of organizations. Reengineering is now having the same type of impact on the rest of the organization. It is useful to examine the history of the total quality movement and the impact it has had on the firm in order to understand how reengineering is changing business functions.

In the early 1900s, beginning with Henry Ford, Fredrick Taylor, and John Radford, quality was defined as conformance to "specifications" and relied on inspections and activities such as counting, grading, and rework. Walter Shewhart's 1931 book *Economic Control of Quality of Manufacturing Products* proposed the Statistical Quality Control system. This system was a product-based, manufacturing view of quality, focusing on manufacturing and engineering practices and relying on statistical analysis.

The Deming philosophy was based on Walter Shewhart's ideas and emphasized management's responsibility for continuous improvement of the systems of production. Kaoru Ishikawa also advocated this approach and developed the "Ishikawa Diagram" to examine causes of manufacturing problems. Joseph Juran was the next main influence on the quality movement and proposed the idea of the "cost of quality." In 1956, Armand Fergenbaum extended this principle and argued for a more comprehensive plan for the total quality movement. He argued that even though statistical principles and management controls were necessary, other activities such as new product design and vendor selection must be emphasized to improve product quality. Other quality ideas have also targeted the manufacturing environment—JIT, Kaizan or continuous improvement, Taguchi methods, Poka-yoke, and the Shingo system. As a result of the advances in these methods, two observations can be made: (1) most organizations have adopted some form of a quality program, and (2) this has resulted in a major improvement in product quality.

The results from such programs are promising. A study by *Business Week* in 1994 found that the quality of U.S. manufactured products is up significantly since 1990. Productivity in the manufacturing sector has also been affected. Being a quality leader may have a financial impact as well. Consider a 1993 *Business Week* article which reported that if $1000 were invested in each publicly traded company that has won the U.S. Commerce Department's Malcolm Baldrige National

Quality Award at the time the award was announced, a cumulative gain of 89.2% would result.

The above discussion has focused on improving quality in the United States. One effect of this is that consumer expectations regarding products and services are increasing. As better products become the norm, companies are forced to meet a minimum level of quality before they can compete. As a result, organizations have begun examining other areas where efficiencies can be achieved. This has led to an analysis of the business processes that bring a quality product to the market.

SEE ALSO THE FOLLOWING ARTICLES

Benchmarking • Cost/Benefit Analysis • Enterprise Resource Planning • Executive Information Systems • Operations Management • Project Management Techniques • Quality Information Systems • Resistance to Change, Managing • Staffing the Information Systems Department • Total Quality Management and Quality Control

BIBLIOGRAPHY

Hammer, M. (1990). Reengineering work: Don't automate, obliterate. *Harvard Business Review,* July/August.

Hammer, M., and Champy, J. (1993). *Reengineering the corporation: A manifesto for business revolution.* New York: Harper Collins.

Industry Week. (1990). Creating the computer-integrated enterprise. *Industry Week,* June, 42–64.

Sastry, M. A. (1997). Problems and paradoxes in a model of punctuated change. *Administrative Science Quarterly,* Vol. 42, 237–275.

Sethi, V., and King, W. R. (1998). *Organizational transformation through business process reengineering: Applying the lessons learned.* Upper Saddle River, NJ: Prentice Hall.

Relational Database Systems

Catherine M. Ricardo
Iona College

GLOSSARY

attribute A characteristic of an entity, represented in the relational model as a column of a table.
candidate key An attribute or minimal combination of attributes that could be used to uniquely identify tuples in a relation.
domain A set of possible values for an attribute.
entity A person, place, event, concept, or other "thing" which is to be represented by data in the database.
foreign key An attribute or set of attributes within a relation that matches the primary key of some, usually different, relation.
normalization A process for designing relational schemas with minimal redundancy and data anomalies.
primary key An attribute or combination of attributes that is used to uniquely identify tuples in a relation.
relation A set of *n*-tuples, each of which represents facts about an entity or relationship.
relationship An association or logical connection between entities.
schema The structure of a relational database, specified by writing the name of each relation and a parenthesized list of its attributes, with primary keys and foreign keys indicated. The schema also includes domains, views, character sets, constraints, stored procedures, authorizations, and other related information.
superkey Any attribute or set of attributes that uniquely identifies an entity.
table A physical representation of a relation, consisting of rows that correspond to entities and columns that correspond to attributes.
tuple A row of a table, corresponding to a single record.

I. INTRODUCTION

The relational model is currently the most widely used model for database management systems. The model itself is powerful but conceptually simple, and it is based on the mathematical notion of a relation. In the theoretical model, data are represented by relations, which are sets of tuples of attributes and their values. The relations are physically implemented as tables. The columns of the tables stand for data attributes. Each row of the table represents a tuple or record, which is a collection of related data values for the attributes. When discussing the theoretical model, we use the terms relation, attribute, and tuple. However, in the implementation of the model, these constructs are physically represented as tables, columns, and rows, respectively. Relationships between tables are represented by the use of foreign keys. The structure of a relational database is described in its schema, often presented as a list of its tables, each having a list of its attributes, with the primary keys and foreign keys indicated. The schema also includes constraints, domains, views, character sets, authorizations, stored procedures, and other related information. A design

process called normalization is used to help produce a good relational schema. A number of data languages can be used with the relational model, but SQL, Structured Query Language, is the standard relational language. There are many relational database management systems available; among them the best-selling database management products are Access, Oracle, DB2, SQL Server, and Sybase.

II. HISTORY OF THE RELATIONAL MODEL

The relational model was described by E.F. Codd in his 1970 paper, "A Relational Model of Data for Large Shared Data Banks." Existing database management systems at that time were hierarchical and network model systems. These systems were outgrowths of earlier file management systems and were complex and difficult for users. Codd's seminal paper proposed a revolutionary change in the structure of databases, and it sparked a flurry of research projects to test his theories. An early relational model project, System R, was developed at the IBM research laboratory in San Jose, California, in the 1970s. A more theoretical early project was the Peterlee Relational Test Vehicle, developed at the IBM UK Scientific Laboratory. Another early implementation of the model was INGRES, developed at the University of California at Berkeley. All of these projects resulted in many research papers and projects dealing with issues in the implementation of the relational model, including data representation, data languages, transaction management, user interfaces, and others. In the early 1980s microcomputer-based implementations of database management systems, overwhelmingly based on the relational model, became available. Since that time, the relational model has gained wide acceptance and has continued to be the subject of research studies to extend its capabilities. Extensions include efforts to add object-oriented concepts that can support more complex data, to capture more semantic information about the data, and to add intelligence to database systems.

III. THEORETICAL FOUNDATIONS OF THE RELATIONAL MODEL

The relational model is based on some fundamental ideas from mathematics that Codd extended to the realm of database systems.

A. Sets and Domains

A domain is a finite or infinite set from which values can be chosen. For example, the set of all integers is an infinite domain. The set of names, {Ann, Bob}, is a finite domain. The set of all integers between 5 and 7 inclusive, {5,6,7}, is another finite domain.

B. Cartesian Product and Tuples

For two domains D1 and D2, the Cartesian product, denoted D1 × D2, is defined as the set of all ordered pairs such that the first element is a member of D1 and the second element is a member of D2. That is,

$$D1 \times D2 = \{(d1,d2) | d1 \in D1, d2 \in D2\}.$$

If we let D1 be the set {Ann, Bob} and D2 be the set {5,6,7} then the Cartesian product is the set of ordered pairs that can be formed by choosing the first from D1 and the second from D2,

$$D1 \times D2 = \{(Ann,5), (Ann,6), (Ann,7), (Bob,5), (Bob,6), (Bob, 7)\}.$$

If we have three domains, D1, D2, and D3, we define the Cartesian product as the set of ordered triples with the first element from D1, the second from D2, and the third from D3. For example, if we let D3 = {red,blue} and let D1 and D2 be the sets defined earlier, then

$$D1 \times D2 \times D3 = \{(Ann,5,red), (Ann,6,red), (Ann,7,red), (Bob,5,red), (Bob,6,red), (Bob,7,red), (Ann,5,blue), (Ann,6,blue), (Ann,7,blue), (Bob,5,blue), (Bob,6,blue), (Bob,7,blue)\}.$$

If we have n domains, D1, D2, . . . , Dn, the Cartesian product is the set of ordered n-tuples, defined as

$$D1 \times D2 \times \ldots \times Dn = \{(d1, d2, \ldots, dn) | d1 \in D1, d2 \in D2, \ldots, dn \in Dn\}.$$

In the case of n domains we called the members of the Cartesian product n-tuples or just tuples, rather than ordered pairs or ordered triples.

C. Relations

If we have two domains D1 and D2, a relation can be defined as any subset of the Cartesian product of the domains. In our earlier example D1 × D2 had 6 ordered pairs. We can choose any number of these to form a relation, R. For example, we might choose

$$R = \{(Ann,5), (Bob,5)\}.$$

Sometimes it is possible to define the relation by giving a rule for the selection of members. For example, the following is an equivalent definition of R:

$$R = \{(d1,d2) \mid d1 \in D1, d2 \in D2, \text{ and } d2 = 5\}.$$

A second relation, S, could be defined simply by listing the members, as

$$S = \{(Ann,6), (Bob,6), (Ann,7), (Bob,5)\}.$$

For $D1 \times D2 \times D3$, shown earlier, an example of a relation might be

$$T = \{(Ann,5,red), (Bob,6,blue)\}.$$

For n domains, a relation would be any number of the n-tuples that we choose to be members of the relation.

Codd made a connection between relations in mathematics and stored facts in a database. For example, suppose D1 = {Ann, Bob} represents the set of tenants in an apartment building. Suppose D2 = {5,6,7} represents the apartment numbers. Suppose {red, blue} represents the set of favorite colors that people have. Then we could choose the tuples of the relation to correspond to facts about the occupants of the apartments and their favorite colors. For example, the relation

$$T = \{(Ann,5,red), (Bob,6,blue)\}$$

would mean that Ann occupies apartment 5 and her favorite color is red, while Bob occupies apartment 6 and his favorite color is blue. Obviously, when we create relations in a database we are not choosing tuples from the Cartesian product arbitrarily. Instead, we choose the tuples that correspond to values from the domains that represent facts. The name "relational" refers to relations of this type. They are the sets of tuples that represent facts that we wish to store in the database.

IV. STRUCTURE OF THE MODEL

A. Attributes

In the relational model, the word attribute has essentially the same meaning as it does in other models, such as the entity-relationship model. Attributes are data items of interest. Examples of attributes are customer names, balances of accounts, dates of transactions, and so on.

B. Tables

Relations are physically represented by means of two-dimensional tables in which the columns represent the attributes and the rows are the tuples of the relation. Each table represents the data for an entity set or for a relationship set. For example, Figure 1 shows a database with three tables. The Customer table displays the facts about customers. Each row lists the values of the attributes for a single customer. The column names are chosen to describe the attributes they represent. For example, the first row tells us that the customer with ID of C101 is named Martinez, lives in New York, and has a credit rating of 20. Similarly, the Item table gives facts about items in inventory. The Order table describes orders, including the customers who placed them and the items ordered. We assume each order is for only one item.

1. Domains

Domains are the sets from which values of attributes are chosen. However, in the relational model a domain must be an elementary data type, consisting of atomic, indivisible units. Complex types that can be broken down into simpler components are not permitted. Domains can be either finite or infinite. For example, the domain for creditRating might be specified as the (finite) set of positive integers between 1 and 20, while the domain of custName would be infinite, consisting of any string that could be interpreted as a name. A fundamental assumption is that all relations obey domain constraints, which means that the values in a column are taken from the domain of that attribute.

2. Null Values

For some attributes, the value for a particular entity may be unknown, missing, or not applicable. In that case the table is permitted to have a null value for that attribute. For example, the Item table in Figure 1 has no value for qtyOnHand for item I1004. This may mean that the item has never been ordered, or that we do not know how many we have, or that someone has neglected to enter the value. In the Customer table, if we had a new customer for whom we have not yet calculated a credit rating, we might enter the Customer record without a value for that column.

3. Keys

The notion of key is fundamental to relations. Just as in mathematical sets we must be able to distinguish

Customer	custID	custName	custCity	creditRating
	C101	Martinez	New York	20
	C105	Jones	London	20
	C110	LeBlanc	Paris	15
	C118	Wright	New York	10
	C125	LeBlanc	Montreal	18

Item	itemNo	itemName	price	supplier	qtyOnHand
	I1001	widget	2.99	Ace	200
	I1004	manifold	5.50	Acme	
	I1010	widget	3.75	Wright	150
	I1015	brace	6.80	Ace	16

Order	orderNo	custID	itemNo	qtyOrdered
	O10101	C101	I1004	50
	O10102	C105	I1010	30
	O10103	C118	I1015	5
	O10104	C101	I1001	30
	O10105	C125	I1015	10

Figure 1 The OrderSystem Database.

one element from another, with no repeats, it must always be possible to tell tuples apart by examining their values. Traditionally, a key consists of an attribute that has unique values. In a given relation there may be several such attributes, called candidate keys, and we can choose the one we want to use for the primary key. For example, if we are storing data about employees, and the company assigns a unique empID for each one, but we also store some national identifier such as Social Security Number, both of these attributes are candidate keys. We choose one of them as the primary key. If there is no single attribute with unique values, then a composite key, consisting of two or more attributes, may be used. Again, there may be several candidate composite keys. When we choose a composite key, it is essential that it be irreducible, i.e., that all the attributes be needed for uniqueness.

We sometimes use the primary key of one relation as an attribute of another relation, in order to show relationships between them. For example, in Figure 1, custID, which is the primary key of the Customer table, appears as an attribute of the Order table as well. The purpose of this repetition is to connect an order record to the customer who placed it. The custID is called a foreign key in the Order table. In general, a foreign key is an attribute that is the primary key of another relation, which we can call its home relation. In the Order table, itemNo is also a foreign key that refers to the Item table.

C. Schema

A schema for a relation consists of the name of the relation and its attributes. The relation schema is represented by a simple listing, such as

Customer(custID, custName, custCity, creditRating).

A database schema includes the schemas for all its relations. It is customary to underline the primary key for each relation. Often foreign keys are indicated in some way, such as by italics. We could give the database schema for the OrderSystem database shown in Figure 1 as

Customer(<u>custID</u>, custName, custCity, creditRating)

Item(<u>itemNo</u>, itemName, price, supplier, qtyOnHand)

Order(<u>orderNo</u>, *custID, itemNo,* qtyOrdered).

The schema is a relatively unchanging aspect of a database. It is the underlying structure, modified only rarely, when new data requirements arise. The actual data in the database, the records of actual customers, items, and orders, containing values such as those

shown in the bodies of the tables in Figure 1, are called an instance of the database. The database instance changes constantly, as new orders are placed, new customers or new items are added, old records are deleted, or data item values are changed. Some authors use the word "intension" to refer to the schema or stable structure, and "extension" to refer to the instance. Strictly speaking, the schema also includes domains, views, character sets, constraints, stored procedures, authorizations, and other related information.

D. Characteristics of Relations

Relations have certain properties because of the way they are defined to correspond to mathematical relations. Since the underlying domains are scalar data types, each attribute must have a single irreducible value for each tuple. When we see the table in spreadsheet-type view, each cell of the table must have just one value. For example, in the Item table in Figure 1, we can list only one supplier for each item. The order of tuples is immaterial. Although it may be convenient to list tuples in order by primary key, the relation remains the same if we display the tuples in a different order, because the elements of a set are unordered. The order of columns is also immaterial. Note that this assumption is a departure from the usual mathematical definition of relation, in which the individual elements within the tuples in the Cartesian product must be in order. In the relational data model, it is the name of the attribute, rather than its position, that indicates its meaning. Although it is usual to display a table's columns in the order shown in its schema, the relation remains the same if we switch the positions of columns in the schema. Of course when we display the data, every row in the table will have the attribute values placed in a consistent fashion to correspond to the position of the column names. There are no duplicate tuples, just as there are no repeating elements in a set. This property guarantees that it is always possible to distinguish between tuples, which means there will always be a key, although it may be a composite one.

E. Integrity Rules

Two fundamental rules defined by Codd are entity integrity and referential integrity. Entity integrity means that all components of the primary key must always have an actual value for each tuple of the table. No null values are permitted for any part of the primary key. This rule guarantees that we can always tell tuples apart. The second rule is referential integrity, which refers to foreign keys. Any nonnull value of a foreign key must match a value in the home relation. For example, a custID value in the Order table of Figure 1 must be one of the values of custID in the Customer table. An order record cannot refer to a customer who does not exist. In some cases, the foreign key value is permitted to be null.

F. Views

The actual tables in a relational database are called base tables. A database administrator can grant users permission to access and/or modify the contents of those tables. In some cases, the administrator may wish to allow a user to access only a subset of a base table, or a virtual table created by combining parts of base tables, or by performing summary or other group functions on them. The database administrator can create a view, which can be thought of as either a window into a base table, or as a virtual table. Although the definition of the user's view is stored permanently, the view itself is created dynamically when the user requests access to it. The views that act as windows into a base table can change dynamically as the underlying table is updated. Others are "snapshots," and their contents remain static during the user's session, although they are updated for each new session. Users can perform queries on views. Depending on the nature of the view, it may be possible for the user to make updates that affect the underlying base tables. This might be the case when the user's view contains the primary key and all other required fields of a table, and the view is a window into a base table. A view is both a security mechanism, protecting hidden attributes, and a facility for the user, simplifying his model of the database. It also provides a way to insulate the user from changes to the database structure, a characteristic called logical data independence. For example, if the Customer table is restructured to include the customer's telephone number, a view may be defined that does not include the new attribute, allowing existing applications to continue functioning by using the view.

G. System Catalog

A relational database is self-describing. When the database designer executes the commands that create or

modify the structures of tables, views, and indexes, the system itself records a description of the database in its system catalog. The catalog is itself a small relational database that stores information about each relation, its attributes, its indexes, and any integrity constraints such as primary and foreign keys. One of the tables in the system catalog keeps track of the database's tables. Each row lists all the information about one of the tables in the database, including its name, number of attributes, the name of its creator, the date it was created, and so on. Another table describes all the attributes in the entire database. Each row lists an attribute, the table it belongs to, its data type, and so on. Similar tables are kept for views, indexes, and constraints. The catalog also stores statistics about the size of tables, the size and number of values of indexes, the range of values for some attributes, and other data that are used by the query subsystem to plan efficient execution of queries. Access control information and usage information about users are also stored in the catalog. Since the catalog is a relational database, it can be queried using SQL or other query language.

V. RELATIONAL DATA LANGUAGES

A number of languages can be used to create, manipulate, and administer a relational database. These include SQL, Query by Example, and two theoretical languages, relational algebra and relational calculus. Some of the query languages are procedural, which means they specify the sequence of operations to be performed to retrieve the desired data. Others are declarative (nonprocedural) and specify what data are desired, but now how to retrieve it.

A. Relational Algebra

Relational algebra is a formal, theoretical language that provides a basic set of operators that can be applied to tables and that return another table as a result. Codd originally proposed eight basic relational algebra operators, but the set has been extended. Relational algebra is a procedural language. Although it is not implemented in its native form in commercial database management systems, it is useful because it enables us to understand the basic retrieval operations that are required for any query language. It also serves as a benchmark for other relational languages, in that any relational language that can perform the same operations as relational algebra is said to be relationally complete. The original language proposal included four set operations, Union, Intersection, Difference, and Cartesian Product. These operators are defined to be similar to the corresponding set-theoretic operations in mathematics. However, the elements of the sets in this case are tuples, the rows of the tables, rather than individual values.

1. Union

The relational algebra union operation on tables R and S, denoted R ∪ S, is a binary operator that is applied to two tables that are union-compatible, which means that they have the same attributes, defined on the same domains, and in the same order, on both tables. For example, consider the union-compatible tables shown in Figure 2:

AutoInsCust(custID, lname, fname, agentID)

LifeInsCust(custID, lname, fname, agentID).

We can infer that the tuples in the first table show information about customers of an insurance agency who own automobile insurance policies and the tuples of the second about customers who own life insurance policies. We assume a customer can own either or both types of policies, but a customer always has the same agent. The union of the two tables consists of the tuples that are in either AutoInsCust or LifeInsCust or in both. The result is shown in Figure 3. It lists customers who own either type of policy, or both. Observe that duplicates have been eliminated, so that those who own both types are listed only once in the result. If the two original tables have essentially the same attributes, with the same domains, but different names, it is possible to rename the attributes so that they match and then the union can be performed.

2. Intersection

The intersection operation, written R ∩ S, is another binary operator that is applied to union-compatible

AutoInsCust	custID	lname	fname	agentID
	C123	Smith	John	A444
	C456	Jones	Mary	A555
	C789	Adams	Sue	A444
LifeInsCust	**custID**	**iname**	**fname**	**agentID**
	C123	Smith	John	A444
	C789	Adams	Sue	A444
	C246	Barclay	Tom	A777

Figure 2 Union-Compatible Tables.

custID	lname	fname	agentID
C123	Smith	John	A444
C456	Jones	Mary	A555
C789	Adams	Sue	A444
C246	Barclay	Tom	A777

Figure 3 AutoInsCust ∪ LifeInsCust.

tables. It consists of the tuples that are in both of the tables simultaneously. For the tables in Figure 2, the intersection is shown in Figure 4. It consists of customers who own both automobile insurance and life insurance policies with the agency.

3. Difference

The difference between two union-compatible tables, R − S, is the set of tuples that belong to the first table, R, but not to the second, S. The result of AutoIns Cust − LifeInsCust, shown in Figure 5, is the set of tuples of AutoInsCust that are not also in LifeInsCust. These are the customers who own automobile insurance policies, but not life insurance policies. Note that, unlike Union and Intersection, Difference is not commutative, meaning you do not get the same results if you interchange the two tables.

4. Cartesian Product

As described previously, the Cartesian product of two sets, A and B, consists of all the ordered pairs that can be constructed with the first element coming from the first set, A, and the second element coming from the second set, B. We can extend that notion to tables, R and S, which have distinct attributes. The product of the tables consists of all tuples that can be formed by concatenating every tuple of R with every tuple of S. For example, consider the Customer table and the Item table in Figure 1. The Cartesian product, Customer × Item, is shown in Figure 6. The number of columns in the result is the sum of the number of columns in each of the two original tables. For the example shown, there are 4 columns in Customer and 5 in Item, so there are 9 columns in the result. The number of rows is the product of the number of rows in the original tables. Since there are 5 rows in Customer and 4 in Item, the result table shows 20 rows. Note that in the case where the two original tables have one or more attribute names that are identical, we can use the name of the original table as a prefix before the attribute name in the result, so that each of the columns in the result can have a unique name. For example, if custName and itemName were both simply called name in their respective tables, we could call the first Customer.name and the second Item.name in the result.

custID	lname	fname	agentID
C123	Smith	John	A444
C789	Adams	Sue	A444

Figure 4 AutoInsCust ∩ LifeInsCust.

custID	lname	fname	agentID
C456	Jones	Mary	A555

Figure 5 AutoInsCust − LifeInsCust.

Codd also described several relational operators created for relational databases, including Select, Project, Equijoin, and Divide.

5. Select

The selection operator is unary, which means it is applied to one table at a time. The result is a new table that has the same structure as the original. The operation takes rows from the original table that satisfy a specified condition, called the selection condition, producing a horizontal subset of the table. Symbolically, we write

$$\sigma_{\text{<selection condition>}}\ (\text{table-name}).$$

The Greek letter σ (sigma) is the symbol for Select. The subscript condition, sometimes written as the Greek letter θ (theta) and called the θ-condition, is a predicate involving some condition on the table, normally using a comparison operator and one or more attributes. For example, if we want to choose the rows of the Customer table in Figure 1 where the creditRating is greater than 15, we would write

$$\sigma_{\text{creditRating} > 15}\ (\text{Customer}).$$

The result of this operation is shown in Figure 7. The selection condition may use any of the standard comparison operators {=, <, <=, >, >=, ≠} to compare an attribute with a constant value or with another attribute. The logical connectives AND, OR, and NOT can also be used to make a compound selection condition. For example, if we wrote

$$\sigma_{\text{creditRating <=10 AND custCity} \neq \text{'New York'}}\ (\text{Customer})$$

custID	custName	custCity	creditRating	itemNo	itemName	price	supplier	qtyOnHand
C101	Martinez	New York	20	I1001	widget	2.99	Ace	200
C101	Martinez	New York	20	I1004	manifold	5.50	Acme	
C101	Martinez	New York	20	I1010	widget	3.75	Wright	150
C101	Martinez	New York	20	I1015	brace	6.80	Ace	16
C105	Jones	London	20	I1001	widget	2.99	Ace	200
C105	Jones	London	20	I1004	manifold	5.50	Acme	
C105	Jones	London	20	I1010	widget	3.75	Wright	150
C105	Jones	London	20	I1015	brace	6.80	Ace	16
C110	LeBlanc	Paris	15	I1001	widget	2.99	Ace	200
C110	LeBlanc	Paris	15	I1004	manifold	5.50	Acme	
C110	LeBlanc	Paris	15	I1010	widget	3.75	Wright	150
C110	LeBlanc	Paris	15	I1015	brace	6.80	Ace	16
C118	Wright	New York	10	I1001	widget	2.99	Ace	200
C118	Wright	New York	10	I1004	manifold	5.50	Acme	
C118	Wright	New York	10	I1010	widget	3.75	Wright	150
C118	Wright	New York	10	I1015	brace	6.80	Ace	16
C125	LeBlanc	Montreal	18	I1001	widget	2.99	Ace	200
C125	LeBlanc	Montreal	18	I1004	manifold	5.50	Acme	
C125	LeBlanc	Montreal	18	I1010	widget	3.75	Wright	150
C125	LeBlanc	Montreal	18	I1015	brace	6.80	Ace	16

Figure 6 The Cartesian Product: Customer × Item.

the result would be an empty table, since no row of Customer satisfies this condition.

6. Project

Project is a unary operator that produces a vertical subset of a table. The result is a new table with only the columns specified in a list of attributes, called the projection list, of the original table. For those columns, it eliminates duplicate rows so that only the unique combinations of values are returned. The general form is

$$\Pi_{\text{projection list}} \text{ (table-name)}.$$

The projection list can be a single column, as in

$$\Pi_{\text{supplier}} \text{ (Item)}.$$

Using the Item table shown in Figure 1, the result of this projection is shown in Figure 8. Note that the duplicate value for Ace was eliminated. If we project over more than one attribute, the unique combinations of values appear. For example,

$$\Pi_{\text{custName, custCity}} \text{ (Customer)}$$

produces the table shown in Figure 9. Even though some names are repeated and some cities are repeated, no given combination of values is repeated.

custID	custName	custCity	creditRating
C101	Martinez	New York	20
C105	Jones	London	20
C125	LeBlanc	Montreal	18

Figure 7 Example of Selection: $\sigma_{\text{creditRating} > 15}$ (Customer).

supplier
Ace
Acme
Wright

Figure 8 Example of Projection: Π_{supplier} (Item).

custName	custCity
Martinez	New York
Jones	London
LeBlanc	Paris
Wright	New York
LeBlanc	Montreal

Figure 9 Projection over Multiple Attributes: $\Pi_{\text{custName, custCity}}$ (Customer).

7. Equijoin

The equijoin is a binary operator on two tables that have a common attribute. It is formed by concatenating rows from the first table with rows from the second table that have the same value for the common attribute. If A is the attribute that appears in both tables, R and S, the equijoin could be expressed as

$$R \mid\times\mid_{R.A = S.A} S.$$

Equivalently, the equijoin could be expressed as the result of performing the Cartesian product, and then selecting the rows that have the same value for the common attribute. For R and S, the equijoin is the same as

$$\sigma_{R.A = S.A} (R \times S).$$

As an example, we can form the equijoin of Customer and Order, since both tables have custID, giving the result shown in Figure 10. We could form the entire Cartesian product of the two tables, and then choose only those rows where Customer.custID = Order.custID. Note that if the two tables have attributes with a common domain, even if the attributes have different names, the equijoin can still be performed.

8. Division

Division is a relatively complex operator that is defined on two tables whose schemas are related so that one (the divisor) is a subset of the other (the dividend). The schema for the result (the quotient) is the set of attributes that appear on the dividend table but not on the divisor table. The tuples are the values of these attributes that appear on the dividend table with all values that appears on the divisor table. The general notation is

$$R \div S.$$

For example, assume we have the two tables shown in Figure 11. We note that all of the attributes of Skill (skillName) are contained in the HasSkill table (skillName, empName). Here, Skill is the divisor and HasSkill is the dividend. The division operation HasSkill ÷ Skill produces a quotient, a table showing the names of employees who have all the skills listed on the Skill table, as shown in Figure 12. Several other join operators have been added to relational algebra since Codd's original work. They include the theta-join, the natural join, and the outerjoin.

9. Theta Join

A theta join is a binary operation defined as the result of taking the Cartesian product of the two tables and then applying selection, using a θ-condition, to the result. The general form of a theta join is

$$R \mid\times\mid_{\theta} S.$$

From the definition, this is equivalent to $\sigma_{\theta} (R \times S)$. For example, if we wanted to take a theta join of the Customer and Item tables from Figure 1, with θ being creditRating > qtyOnHand, we would have the result shown in Figure 13. This result is obtained by applying the selection operator θ to the Cartesian product of the two tables, which we found in Figure 6. An equijoin could be considered a specialized form of the theta-join, in which the condition is equality.

10. Natural Join

When an equijoin is performed, the resulting table will always have two columns with identical values. If

Customer.custID	custName	custCity	creditRating	orderNo	Order.custID	itemNo	qtyOrdered
C101	Martinez	New York	20	O10101	C101	I1004	50
C101	Martinez	New York	20	O10104	C101	I1001	30
C105	Jones	London	20	O10102	C105	I1010	30
C118	Wright	New York	10	O10103	C118	I1015	5
C125	LeBlanc	Montreal	18	O10105	C125	I1015	10

Figure 10 Equijoin: Customer $\mid\times\mid_{\text{Customer.custID=Order.custID}}$ Order.

HasSkill	skillName	empName
	programming	Smith
	programming	Jones
	programming	Adams
	systems analysis	Smith
	systems analysis	Adams
	database administration	Smith
	database administration	Jones
	database administration	Adams

Skill	skillName
	programming
	systems analysis
	database administration

Figure 11 Tables for Division.

we remove one of the duplicate columns by means of a projection, so that the common column appears only once in the result, we have a natural join. Because this is the most common type of join, we use the shorthand notation |×| for it, without a subscript, as in

$$R \mid\times\mid S.$$

For example, Figure 14 shows Order |×| Item.

11. Outerjoin

The outerjoin is an extension of the natural join. The natural join is formed and then those tuples from either of the original tables that are not represented in the result because they had no matching tuples are added, with null values for attributes from the other relation. For example, if we took the natural join Customer |×| Order we would find that there is no Order tuple with custID of C110, so that Customer tuple would not be included in the natural join. In the outerjoin, we include such tuples, inserting null values for the attributes from the second table. The result is shown in Figure 15. In this example, only the left-hand table, Customer, had an unmatched tuple. If the right-hand table, Order, had unmatched tuples, then we would have added the unmatched tuples from that table as well. It is possible to distinguish three different outerjoins: the left outerjoin, in which we add only unmatched tuples from the left-hand table, the right outerjoin, in which we add only unmatched tuples from the right-hand table, and the full outerjoin, in which we add unmatched tuples from both tables, if any exist.

empName
Smith
Adams

Figure 12 Division: HasSkill ÷ Skill.

12. Composition of Relational Algebra Operations and Renaming

As we have seen, each relational algebra operation results in another relation, a property called closure. The resulting relation could therefore have been used to perform a second relational algebra operation, giving a sequence of two operations. In fact, we can carry out a sequence of several operations, each of which is performed using the result of the previous one. For example, using Figure 1, suppose we wished to find the names of all Customers who have ordered item I1004. We can see from the Order table that only the first tuple involves I1004. Choosing rows that have a specified value is a SELECT operation, so we write

$$\sigma_{\text{itemNo}='\text{I1004}'}(\text{Order})$$

for that part of the query. Once we have found the tuple or tuples having the correct itemNo value, we do a natural join with the Customer table to find the corresponding customer records. We indicate that the join is to be performed with the previous result by writing

$$(\sigma_{\text{itemNo}='\text{I1004}'}(\text{Order})) \mid\times\mid \text{Customer}.$$

Finally, we find the custName by applying a projection to this result, so that the entire expression becomes

$$\Pi_{\text{custName}}((\sigma_{\text{itemNo}='\text{I1004}'}(\text{Order})) \mid\times\mid \text{Customer}).$$

It is permissible to assign names to the intermediate results, a process called renaming, by a simple statement such as

$$\text{Temp1} \leftarrow \sigma_{\text{itemNo}='\text{I1004}'}(\text{Order}).$$

custID	custName	custCity	creditRating	itemNo	itemName	price	supplier	qtyOnHand
C101	Martinez	New York	20	I1015	brace	6.80	Ace	16
C105	Jones	London	20	I1015	brace	6.80	Ace	16
C125	LeBlanc	Paris	18	I1015	brace	6.80	Ace	16

Figure 13 Theta Join: Customer $|\times|_{\text{creditRating} > \text{qtyOnHand}}$ Item.

Then we could write

$$\text{Temp2} \leftarrow \text{Temp1} \; |\times| \; \text{Customer}$$

and

$$\text{Result} \leftarrow \Pi_{\text{custName}} \; (\text{Temp2}).$$

We can rename tables, attributes, or both using a Rename operator, usually designated by the Greek letter rho, ρ. If R is a table having attributes A1, A2, . . . , An and we wish to rename the table to S and/or the attributes to B1, B2, . . . , Bn, we write

ρ_S (R) to rename the table to S and keep all the attribute names

or

$\rho_{(B1, B2, \ldots, Bn)}$ (R) to keep the table name R but change attribute names to Bi

or

$\rho_{S(B1, B2, \ldots, Bn)}$ (R) to change the table name to S and the attribute names to Bi.

For example, to change the name of the Customer table to Cust and the names of some of the attributes, we could write

$$\rho_{\text{Cust(custNumber, custName, custCity, custCredit)}} \; (\text{Customer}).$$

We can then refer to Cust and its attributes in subsequent relational algebra expressions, such as

$$\text{Cust} \; |\times|_{\text{custNumber=custID}} \; \text{Order}.$$

13. Other Extensions of Relational Algebra

Relational algebra has been extended in many ways by various researchers. Among the most important extensions are aggregate functions such as SUM, AVG, MAX, MIN, and COUNT. The standard operators have also been extended to include systematic treatment of null values.

14. Query Optimization

Although relational database management systems use a variety of query languages, when the queries are implemented the resulting operations can be expressed in relational algebra or some equivalent language. It is sometimes possible to express the same relational algebra query using different operations or different sequences of operations. Some of these equivalent expressions are more efficient than others. The DBMS typically has a query optimizer to handle the task of finding and evaluating the alternative ways a query could be executed, producing a query plan that dictates which operations will be executed and in what order. To do so, the optimizer uses some common algebraic laws to develop alternative relational algebra expressions. It can be proven, for example, that joins are commutative. If R and S are two tables, then

$$R \times S = S \times R, \quad \text{and} \quad R \; |\times| \; S = S \; |\times| \; R.$$

Joins are also associative. If R, S, and T are three tables, then

$$(R \times S) \times T = R \times (S \times T) \quad \text{and}$$
$$(R \; |\times| \; S) \; |\times| \; T = R \; |\times| \; (S \; |\times| \; T).$$

orderNo	custID	itemNo	qtyOrdered	itemName	price	supplier	qtyOnHand
O10101	C101	I1004	50	manifold	5.50	Acme	
O10102	C105	I1010	30	widget	3.75	Wright	150
O10103	C118	I1015	5	brace	6.80	Ace	16
O10104	C101	I1001	30	widget	2.99	Ace	200
O10105	C125	I1015	10	brace	6.80	Ace	16

Figure 14 Natural Join: Order |×| Item.

custID	custName	custCity	creditRating	orderNo	itemNo	qtyOnHand
C101	Martinez	New York	20	O10101	I1004	50
C101	Martinez	New York	20	O10104	I1001	30
C105	Jones	London	20	O10102	I1010	30
C110	LeBlanc	Paris	15	null	null	null
C118	Wright	New York	10	O10103	I1015	5
C125	LeBlanc	Montreal	18	O10105	I1015	10

Figure 15 Outerjoin: Customer OUTERJOIN Order.

Set union is also both commutative and associative, as is set intersection. There are also laws governing selection, projection, and other operations. The laws involving selection are particularly useful, because selection usually produces a result that is significantly smaller than the original table. Therefore, if a selection can be done early in the query plan, the rest of the query operations may be performed on tables that are much smaller than the initial ones, resulting in considerable improvement in efficiency. The most common heuristic, or "rule of thumb" used by optimizers is to do selection as early as possible. For example, if we wanted to find the item number of all items ordered by customer Martinez, we could start by doing a natural join of the Order and Customer tables over the common column, custID, then do a selection to find the rows where the custName is Martinez, then do a projection over the itemNo. The relational algebra translation, using renaming, would be

$$\text{Temp1} \leftarrow \text{Customer} \mid\times\mid \text{Order}$$

$$\text{Temp2} \leftarrow \sigma_{\text{custName}=\text{'Martinez'}} (\text{Temp1})$$

$$\text{Result} \leftarrow \pi_{\text{itemNo}} (\text{Temp2}).$$

For the data shown, Temp1 would contain 5 records, each having 7 attributes. Temp2 would result in 2 records, still having 7 attributes. The result would have 2 records with a single attribute each.

However, if the selection is done first, we would get

$$\text{Temp1} \leftarrow \sigma_{\text{custName}=\text{'Martinez'}} (\text{Customer})$$

$$\text{Temp 2} \leftarrow \text{Temp1} \mid\times\mid \text{Order}$$

$$\text{Result} \leftarrow \pi_{\text{itemNo}} (\text{Temp2}).$$

Using this scheme, Temp 1 would have only 1 record. Temp 2 would have 2 records, with 7 attributes each, and the result would be as before. By doing the selection early, we have reduced the size of the first intermediate table considerably. Relational database management systems use algebraic equivalences like these to transform queries into more efficient equivalent forms. They then choose among possible query plans based on physical factors such as the existence of indexes, the order of records in tables, the selectivity of conditions, the number of values for attributes, and the number of records for tables.

B. Relational Calculus

Relational calculus is another formal, theoretical language for expressing relational database queries. It is equivalent to relational algebra, which means that any query that can be expressed in one of the languages can also be written in the other. Therefore, relational calculus is relationally complete. It is a nonprocedural language, allowing us to express what data we want without specifying how the retrieval is to be done. There are two forms, tuple relational calculus and domain relational calculus. Both use first-order logic, a branch of mathematics.

1. Tuple Relational Calculus

Tuple relational calculus uses variables that range over the tuples of a relation. For example, the query, "Find the custID and custName of all customers with credit rating greater than 15" might be written in a much-simplified version of tuple relational calculus as

$$\{\text{C.custID, C.custName} \mid \text{Customer(C) and C.creditRating} > 15\}.$$

Here C is a tuple variable and we are specifying that we want the set of all custID and custName values for all the tuples (values of C) where C ranges over the Customer table and the creditRating value of C is greater than 15. As in relational algebra, predicates can include the comparison operators $<$, $<=$, $>$, $>=$, $\neq$ and the logical connectives AND, OR, and NOT. The expressions can also include the logical quantifiers

FORALL and THERE EXISTS. There are strict rules about the form of an expression in tuple relational calculus to guarantee that the expression will be safe, which means it will not lead to an infinite result.

2. Domain Relational Calculus

In domain relational calculus the variables range over the domains of attributes. The general form of a query is

$$\{<x1, x2, \ldots, xn> \mid P(x1, x2, \ldots, xn)\}.$$

This means the set of all domain variables x1, x2, . . . , xn for which the predicate P(x1, x2, . . . , xn) is true.

For example, to find complete Item records for which the qtyOnHand is greater than 150, we could write

$$\{<N, A, P, S, Q> \mid <N, A, P, S, Q> \in \text{Item and } Q > 150\}.$$

If we wanted only specific fields of these item records, we would have to add the condition that there exist values for the other attributes for which the predicate is true. For example, to find only the itemNo and itemName for items with qtyOnHand above 150, we would write

$$\{<N, A> \mid (\text{THERE EXISTS } P)\ (\text{THERE EXISTS } S)\ (\text{THERE EXISTS } Q)$$
$$<N, A, P, S, Q> \in \text{Item and } Q > 150\}.$$

C. SQL

SQL, Structured Query Language, is the standard data language for relational database management systems. It is based on tuple relational calculus, although it also has similarities to relational algebra. Most relational database management systems support some version of SQL. SQL has data definition, data manipulation, user authorization, and transaction control facilities. In the data manipulation language, the query statement, the SELECT, incorporates the same fundamental operations as relational algebra. Its syntax is quite similar to the relational algebra statements as well. For example, the query we considered earlier using relational algebra, Find the names of all customers who have ordered item I1004, can be written in SQL as

```
SELECT custName
FROM Customer, Item
WHERE itemNo = 'I1004' AND Customer.custID =
                    Item.custID;
```

D. Query by Example

Query by Example was originally developed by IBM to provide a user-friendly graphical interface for relational databases. It is based on domain relational calculus. Although IBM offers a commercial version that is very close to relational calculus, QBE-type interfaces are available with many other microcomputer-based database management systems, including Microsoft's Access. To design a query in Access, the user is presented with windows that can be modified. In one window the user can choose the tables to be included in the query. They will be displayed there along with their attributes and relationships. The second window is a form that the user can fill in by clicking on attribute names from the tables and adding criteria or conditions. There are options for sorting and other facilities. Once the query is designed, it can be executed immediately, and the results can be displayed or named and saved for use in another query or for reports. There is also an option to allow the user to view the SQL version of the query.

E. ODBC and JDBC

Standards exist for allowing application programs to access data in databases, using the SQL language. Microsoft's Open Database Connectivity (ODBC) technology provides standards for a common interface for accessing SQL-based databases. Using the interface, an application program can access different databases, having various database management systems and operating systems environments, using the same code. ODBC database drivers are provided by most database management system software. For applications written in Java, the de facto standard is JDBC.

VI. NORMALIZATION

Modeling is the process of designing a schema for a database. One method that is widely used in relational modeling is normalization, a technique for creating a database schema that minimizes redundancy and that is free of certain undesirable properties called anomalies. There are several normal forms, each with its own restrictions. The normalization process uses a series of tests in which each relation in a proposed schema is examined to determine whether it satisfies the requirements of each normal form. If it does not conform to the normal form, it may be possible to decompose the relation into two or more equivalent

relations that satisfy the requirements of the form. It is important to note that normalization is concerned with the schema, the intension of the database. An instance or extension may be used to provide examples to guide us to see problems with the schema. An instance may also provide a counterexample, showing that the relation is not in the form required, but an instance cannot prove that a relation is in the normal form. Such proof requires that we examine some fundamental aspects of the relations and attributes. These aspects include functional dependency, multivalued dependency, and properties of decompositions. Codd first proposed normalization and defined the first three normal forms, based on functional dependency. Along with Boyce, he refined the third normal form definition into Boyce–Codd normal form. Other researchers independently identified multivalued dependencies, which are involved in fourth normal form, and join dependencies, which are used in fifth normal form.

A. Functional Dependency

If A and B are attributes or sets of attributes of relation R, then B is said to be functionally dependent on A if each value of A in R has associated with it exactly one value of B. Equivalently, if two tuples of R have the same A value, they must have the same B value. We write this as A → B, read as A functionally determines B. The attribute or attribute set to the left of the arrow is called the determinant. In normalization, the first task is to determine all the functional dependencies, sometimes called FDs, among the data items. For example, consider the following set of data items for a company with many employees, several departments, and many projects:

{empID, SSN, empName, deptID, deptMgr, jobTitle, rating, projID, projName, projDir, budget, salary, hoursAssigned}.

We assume that SSN represents Social Security number or an equivalent universal identifier, and that empID is a unique identifier issued within the company. We also assume that each employee may work on many projects simultaneously, and that hoursAssigned records the number of hours an employee works on a project, and rating the worker's performance rating on that project. Each project has one director, one name, one budget, and many employees. Project names are unique. Names of persons are not unique. An instance of the Company table is shown in Figure 16.

Given those assumptions, we have the following FDs. Attributes written on the right of the arrow are individually determined by attributes or sets of attributes on the left.

empID → SSN, empName, deptID, deptMgr, jobTitle, salary

SSN → empID, empName, deptID, deptMgr, jobTitle, salary

projID → projName, projDir, budget

projName → projID, projDir, budget

deptID → deptMgr

{empID, projID} → rating, hoursAssigned, plus all attributes functionally dependent on either empID or projID individually

{SSN, projID} → rating, hoursAssigned, plus all attributes functionally dependent on either SSN or projID individually

{empID, projName} → rating, hoursAssigned, plus all attributes functionally dependent on either empID or projName individually

{SSN, projName} → rating, hoursAssigned, plus all attributes functionally dependent on either empID or projName individually.

We ignore "trivial" functional dependencies, in which the attribute(s) on the right is included in the determinants, such as

empID → empID or

{empName, salary} → empName.

Functional dependencies help us to identify candidate keys for a table. If we look at the list of FDs we can identify attributes or sets of attributes such that every attribute is functionally dependent on them. They therefore identify each tuple uniquely. Such a set is called a superkey. For our example, we have many superkeys, some of which are

{SSN, projID}

{empID, projID}

{SSN, projName}

{empID, projName}, but also

{empID, empName, projID}

{projID, SSN, deptID} and others.

Each of these superkeys uniquely identifies each row of the table. That is, if you were told the values of the

empID	SSN	empName	deptID	deptMgr	jobTitle	rating	projID	projName	projDir	budget	salary	hours Assigned
101	111223344	Adams	10	Munez	programmer	4	J10	Atlantic	Quinn	100000	45000	20
101	111223344	Adams	10	Munez	programmer	5	J25	Pacific	Ryan	200000	45000	10
102	234567890	Burns	12	Nolan	DBA	4	J10	Atlantic	Quinn	100000	60000	30
103	222233344	Cabot	11	Peyton	DBA	3	J30	Arctic	Smith	200000	60000	10
104	445566778	Adams	12	Nolan	programmer	3	J25	Pacific	Ryan	200000	50000	15

Figure 16 Example for Normalization: The Company Table.

attributes in any one of these superkeys, you would be able to say which row of the table they occurred in. Note that the last two contain "extra" attributes that are not needed for the functional dependency. Superkeys without such unneeded attributes are called candidate keys. That is, candidate keys are minimal superkeys, sets in which all the attributes are needed for functional dependency. The candidate keys for our example are {SSN, projID}, {empID,projID}, {SSN, projName}, and {empID, projName}. The database designer chooses one of the candidate keys to be the primary key of the relation. All the attributes of any candidate key are called prime attributes.

B. Anomalies

A primary reason to normalize tables is to avoid anomalies that can arise when we update, insert, or delete tuples in a relation. For example, consider the Company database shown in Figure 16 and assume that the primary key is chosen to be {empID, projID}. An update anomaly could occur if we changed information in one tuple, making it inconsistent with the same data in another tuple. For example, if director of the Atlantic project changes from Quinn to Smith, we might update it in the first record, for employee 101, and neglect to update it in other records having the same project. An insertion anomaly occurs when we are unable to insert data because we do not know the value of one of the attributes of the primary key. If there is a new employee who does not yet have a project assignment, we cannot insert the employee's record because the primary key includes projID, for which we have no value. The same problem would arise if we had a project to which no employee were assigned yet. A deletion anomaly occurs when the deletion of one or more records causes us to lose other information as a side effect. For example, if we delete the record for employee 103 we lose all information about the Arctic project, since that employee is the only one assigned to that project. Normalization addresses the issue of anomalies by separating data appropriately.

C. First Normal Form

A relation is in first normal form (abbreviated 1NF) if the domains of all its attributes are atomic, and the value of any attribute in a tuple is a single value from its domain. An equivalent definition is that each attribute is nondecomposable and is functionally dependent on the key. Values that are composites, or multiple values for an attribute, are disallowed. This requirement is actually part of the definition of a relation. A relation that violates this rule is said to be unnormalized. Unnormalized relations are permitted in object-oriented models and in object-relational models, but not in strictly relational ones. The Company table of Figure 16 demonstrates this requirement. Employee 101 is listed twice, once for each project he or she is assigned to. If the table were not in first normal form, we could have put both projects in the same record.

D. Second Normal Form

The definition of second normal form involves the notion of full functional dependency. An attribute or set of attributes B is fully functionally dependent on an attribute or set of attributes A if it is functionally dependent on A, but not on any (proper) subset of A. If the determinant A consists of several attributes, A1, A2, . . . , An and we remove any one of them from A, then B is no longer functionally dependent on this smaller set of attributes. That is, all of the attributes of A are needed to functionally determine B. A relation is in second normal form if it is in first normal form and every nonprime attribute is fully function-

ally dependent on the key. In actuality, the definition extends to full functional dependency on any candidate key as well.

To understand why second normal form is desirable, note that even though the Company relation is in first normal form, we saw that it has insertion, update, and deletion anomalies. Codd identified the source of such problems as partial functional dependencies. The key of the Company relation is {empID, projID}. However, some of the nonkey attributes are functionally dependent on only one of the attributes in the key. For example, empID → empName, without projID. This is an example of partial dependency, which second normal form does not allow. To remedy the problem, the relation should be decomposed by relational algebra projection into three relations, each of which is in second normal form. They are

Employee(empID, SSN, empName, deptID, deptMgr, jobTitle, salary)

Project(projID, projName, projDir, budget)

Assignment(empID, projID, hoursAssigned, rating)

as shown in Figure 17. In this schema, each attribute of Employee stores a single fact about one employee, each attribute of Project a single fact about one project, and each attribute of Assignment a single fact about one employee's work on one project. Note that the insertion, deletion, and update anomalies are resolved in this decomposition, and all the constraints and facts in the original table are preserved.

E. Third Normal Form

Third normal form is based on the notion of transitive dependency. If A, B, and C are attributes of relation R, such that A → B, and B → C, then C is transitively dependent on A, unless either B or C is a candidate key or part of a candidate key. Equivalently, a transitive dependency exists when a nonprime attribute determines another nonprime attribute. A relation is in third normal form if it is in second normal form and no nonprime attribute is transitively dependent on any candidate key. In the Employee table of the company database in Figure 17, we see that deptID → deptMgr, so we have one nonprime attribute determining another. The only candidate keys for the relation are empID and SSN. This means that deptMgr is transitively dependent on empID, the primary key, since neither deptID nor deptMgr is part of any candidate key for this relation. Because of the transitive dependency, we can still have anomalies. If we have a new department with a new manager, we cannot insert this information unless we have an employee, which is an insertion anomaly. If the manager of department 10 changes from Munez to Miller, we may update one employee's record and fail to update another's in the same department, leading to an update anomaly. If we delete the record of the only employee in a department, we lose the information about the department, including the manager's name.

To correct the design, we can decompose the Employee table into

Employee1(empID, SSN, empName, *deptID,* jobTitle, salary)

Department(deptID, deptMgr)

resulting in the schema shown in Figure 18. The Project and Assignment tables are already in third normal form. In the Project table, projID and projName are both candidate keys, and there are no transitive dependencies. In Assignment, only the composite {empID, projID} is a candidate key, and there are no other functional dependencies.

F. Boyce–Codd Normal Form

Boyce–Codd normal form is slightly stricter than 3NF. A relation is in Boyce–Codd normal form (BCNF) if and only if every determinant is a candidate key. (Recall that 3NF allows a determinant to be a part of a candidate key and does not require it to be an entire candidate key.) It is rare to find a relation that is 3NF without also being BCNF, but it is possible when a relation has composite candidate keys that have at least one attribute in common. Our revised table, Em-

Employee(empID, SSN, empName, deptID, deptMgr, jobTitle, salary)

Project(projID, projName, projDir, budget)

Assignment(*empID, projID,* hoursAssigned, rating)

Figure 17 The Company Database in 2NF.

Employee1(empID, SSN, empName, *deptID,* jobTitle, salary)

Department(deptID, deptMgr)

Project(projID, projName, projDir, budget)

Assignment(*empID, projID,* hoursAssigned, rating)

Figure 18 The Company Database in 3NF and BCNF.

ployee1, is in BCNF because it has only two determinants, empID and SSN, both of which are candidate keys. All of the other tables in the schema shown in Figure 18 are also in BCNF. Although a schema can always be decomposed by projection into BCNF relations, it is not always desirable to do so, because the decomposition may place a determinant and the attribute(s) it determines in different relations, thus losing a functional dependency. It has been demonstrated that while it is always possible to find a 3NF decomposition that preserves functional dependencies, it is not always possible to find a BCNF one that does so.

G. Multivalued Dependencies and Fourth Normal Form

Although BCNF eliminates anomalies due to functional dependencies, Fagin, Zaniolo, and Delobel independently identified another type of dependency that can cause problems due to repetition of data. Multivalued dependency may arise out of the process of normalization to achieve first normal form. Recall that first normal form forbids multiple values in a cell of a table. One solution is to repeat the rest of the data along with each of the values of the cell, making the multivalued attribute part of the key. If we have two attributes that are, by nature, multivalued, we must create tuples for every combination of values of one with values of the second. A multivalued dependency exists when there are three attributes A, B, and C in a relation R such that for each value of A the set of B values associated with the A value are independent of the set of C values associated with the A value. We say A multidetermines B and A multidetermines C. By definition, multivalued dependencies occur in pairs. We write

$A \twoheadrightarrow B$

$A \twoheadrightarrow C.$

A trivial multivalued dependency A - >> B is one where either B is a subset of A, or A and B together make up all the attributes of R. A relation is in fourth normal form if it is in BCNF and has no nontrivial multivalued dependencies. For example, consider the relation

Emp(empID, skill, dependentName).

We will assume an employee can have several skills and several dependents. An unnormalized instance of this relation is shown in Figure 19. Since this is not a valid 1NF table, we need to normalize it by removing the repeating values from the skill and dependents cells. When we "flatten" the table in this way, we have to repeat all combinations of skill and dependent name for each employee, to avoid the appearance of a relationship between the skill and the dependent name. The resulting table instance is shown in Figure 20. In this table, all three attributes form the key, and we have

empID	skill	dependentName
E101	French	Mary
	Data entry	Tom, Jr.
E105	Systems analysis	John
	Database design	Jill

Figure 19 The Unnormalized Emp Table.

empID $\twoheadrightarrow$ skill

empID $\twoheadrightarrow$ dependentName.

The table is not in 4NF. To make it 4NF, we decompose it into two tables, as shown in Figure 21:

Emp-skill(<u>empID, skill</u>)

Emp-dependent(<u>empID, dependent</u>).

Each of these tables is in 4NF.

H. Lossless Decomposition and Fifth Normal Form

The process of normalization involves examining tables for dependencies and then decomposing them by projection. However, the decomposition cannot be done arbitrarily. For each of the projections we have created in previous examples, it is possible to get the original table back, with exactly its original tuples, by joining the decomposed tables. A projection that can

empID	skill	dependentName
E101	French	Mary
E101	French	Tom, Jr.
E101	Data entry	Mary
E101	Data entry	Tom, Jr.
E105	Systems analysis	John
E105	Systems analysis	Jill
E105	Database design	John
E105	Database design	Jill

Figure 20 The Emp Table in First Normal Form.

Emp-skill:

empID	skill
E101	French
E101	Data entry
E105	Systems analysis
E105	Database design

Emp-dep:

empID	dependentName
E101	Mary
E101	Tom, Jr.
E105	John
E105	Jill

Figure 21 The Emp Database in 4NF.

(a) empID	(a) task#
e10	t1
e10	t2
e12	t1

(b) task#	(b) proj#
t1	p200
t2	p100
t1	p100

(c) empID	(c) proj#
e10	p200
e10	p100
e12	p100

Figure 23 Projections of EmpTaskProj.

be reversed through a join to recreate the original relation is called a lossless decomposition, or lossless join. Not all decompositions are lossless, which means there are projections whose join does not equal the original relation. As an example of a lossy projection and a join dependency, consider the relation EmpTaskProj(empID, task#, proj#) shown in Figure 22. The table shows which employees perform which tasks for which projects. We can decompose the table by projection into tables (a) and (b) of Figure 23. (For the present, we ignore table (c) of Figure 23.) However, when we join those two tables, in Figure 24, we do not get the original table back, which means that this is a lossy projection. We see that we get an extra tuple that did not appear in the original table. This is an example of a spurious tuple, one created by the projection and join processes. Instead of representing real data, it is an artifact of the decomposition process. The original table can be recreated by joining the result with table (c) of Figure 23, as shown in Figure 25.

A join dependency exists when for a relation R with subsets of its attributes A, B, . . . , Z, R is equal to the join of its projections on A, B, . . . , Z. A relation is in fifth normal form if every join dependency is implied by the candidate keys. This means that the only valid decompositions are those involving the candidate keys. Beyond those, there is no advantage to be had by decomposing the relation further. Join dependencies are a generalization of multivalued dependencies and they can be quite subtle, making it difficult to find them. They are believed to be relatively rare, so in practice designers often stop at 4NF or even BCNF or 3NF, especially if the designer wishes to preserve functional dependencies.

VII. OBJECT-RELATIONAL DATABASES

In recent years, the paradigm for programming languages has shifted to object orientation, with languages such as C++ and Java increasing in popularity. Data to be stored in databases have become increasingly complex, as structured data, multimedia, and other types of information have become more common. These trends have resulted in changes to relational database management systems, creating a merging of relational and object-oriented systems, often referred to as object-relational databases. Many relational DBMSs such as Oracle now support some object-oriented features. Since 1999, the SQL3 standard has also included some object features.

One of the first modifications to the strict relational model was the development of "nested relations." In addition to atomic domains, the object-relational model allows some user-defined data types, including structured types. For example, a database designer might define custAddress to consist of street, city, state, and zip code. Similarly, custName might be defined to consist of firstName, lastName. Then a Customer table could be defined to use custName

empID	task#	proj#
e10	t1	p200
e10	t2	p100
e10	t1	p100
e12	t1	p100

Figure 22 Original Table EmpTaskProj.

empID	task#	proj#
e10	t1	p200
e10	t1	p100
e10	t2	p100
e12	t1	p200
e12	t1	p100

Figure 24 First Join using Table a and Table b of Figure 23.

empID	task#	proj#
e10	t1	p200
e10	t1	p100
e10	t2	p100
e12	t1	j200

Figure 25 Join of First Join with Table c of Figure 23 over {empID, proj#}.

and custAddress as attributes, resulting in a schema that contains schemas, which we could write as

Customer(custID, custName(firstName, lastName), custAdd(street, city, state, zipcode), creditRating).

SQL3 also permits attributes to contain arrays of values. For example, in the Item table items might be available in several colors, and the colors might be stored in an array. Tables that contain such structured values or multiple values would not satisfy the traditional definition of first normal form.

User-defined data types can have their own methods, which are special functions defined for members of the type. Structured types can participate in type hierarchies, with inheritance of attributes and methods by the subtype. The tuples of tables may have unique identifiers that are independent of the values of their attributes. Among the data types permitted is LOB, Large Object, a type that can be used to store multimedia objects. Reference types are also permitted, allowing tuples to refer to other tuples.

VIII. CODD'S RULES FOR RELATIONAL DATABASES

Codd proposed rules that a database management system should follow to be considered relational. Although commercial DBMSs, especially those that are object-relational, do not follow all of these rules, it is interesting to examine the standards that Codd proposed.

1. Information representation. All information is represented at the logical level only as values in tables.
2. Guaranteed access. Users must be able to access any data item by providing its table name, primary key value, and column name.
3. Treatment of null values. Null values (distinct from zero or the empty string) must be represented systematically, regardless of data type.
4. Relational catalog. The system catalog is represented relationally and can be queried in the same way as stored data.
5. Comprehensive data sublanguage. Although a relational system can provide other languages, it must have at least one language that supports data and view definition, data manipulation, integrity constraints, user authorization, and transaction management.
6. View updates. All views that are theoretically updatable in the relational model must be actually updatable in the system.
7. High-level update operations. Any base relation or derived relation that can be treated as a single operand for retrieval can also be handled as a single operand for insertion, update, and deletion operations.
8. Physical data independence. Application programs are not affected by changes to storage representation or access methods.
9. Logical data independence. Application programs are not affected by changes at the logical level that do not affect the information content.
10. Integrity rules. Integrity constraints must be expressed in the data sublanguage and stored in the system catalog, rather than in application programs.
11. Distribution independence. Regardless of whether the data are distributed or centralized, the data manipulation language commands used by applications and users must remain logically the same.
12. Nonsubversion. If the system has a low-level language that supports record-at-a-time access, that language cannot be used to bypass integrity constraints expressed in the high-level language.

The overarching rule, which Codd called Rule 0, or the foundation rule, is that any system that is described as relational must manage its stored data using only its relational capabilities.

SEE ALSO THE FOLLOWING ARTICLES

Database Administration • Database Development Process • Database Systems • Data Modeling: Entity-Relationship Data Model • Data Modeling: Object-Oriented Data Model • Network Database Systems • Object-Oriented Databases • Structured Query Language (SQL) • Temporal Data Model and Query Language Concepts

BIBLIOGRAPHY

Chamberlin, D. (1976). Relational data-base management systems. *ACM Computing Surveys,* Vol. 8, No. 1, 43–66.

Codd, E. F. (1970). A relational model of data for large shared data banks. *Communications of the Association for Computing Machinery,* Vol. 13, No. 6, 377–387.

Codd, E. F. (1982). The 1981 ACM Turing award lecture: Relational database: A practical foundation for productivity. *Communications of the ACM,* Vol. 25, No. 2, 109–117.

Codd, E. F. (1985). Is your DBMS really relational? *Computerworld,* October 14, 1–9.

Date, C. J. (2002). *An introduction to database systems,* 7th ed. Reading, MA: Addison–Wesley.

Delobel, C. (1978). Normalization and hierarchical dependencies in the relational data model. *ACM Transactions on Database Systems,* Vol. 3, No. 3, 201–222.

Elmasri, R., and Navathe, S. (2002). *Fundamentals of database systems,* 3rd ed. Reading, MA: Addison–Wesley.

Fagin, R. (1977). Multivalued dependencies and a new normal form for relational databases. *ACM Transactions on Database Sysems,* Vol. 2, No. 3, 262–278.

Garcia-Molina, H., Ullman, J., and Widom, J. (2002). *Database systems: The complete book.* Upper Saddle River, NJ: Prentice Hall.

Maier, D. (1983). *The theory of relational databases.* Rockville, MD: Computer Science Press.

Ramakrishnan, R., and Gehrke, J. (2000). *Database management systems,* 2nd ed. New York: McGraw–Hill.

Stonebraker, M. (Ed.). (1986). *The INGRES papers.* Reading, MA: Addison–Wesley.

Todd, S. (1976). The Peterlee relational test vehicle—A system overview. *IBM Systems Journal,* Vol. 15, No. 4, 285–308.

Ullman, J. (1988). *Principles of database and knowledge-base systems,* Vol. I. Rockville, MD: Computer Science Press.

Ullman, J. (1989). *Principles of database and knowledge-base systems,* Vol. II. Rockville, MD: Computer Science Press.

Zaniolo, C., and Melkanoff, M. (1981). On the design of relational database schemata. *ACM Transactions on Database Systems,* Vol. 6, No. 1, 1–47.

Research, Electronic

Ulla K. Bunz *Rutgers University*
Howard E. Sypher *Virginia Polytechnic Institute and State University*

GLOSSARY

bots Automated web crawlers underlying search engines which comb the content of a given web site, a domain of sites or the Web itself to return the "hit" list.
competitive intelligence The legal and ethical process of collecting and analyzing information on various business competitors.
cookies Short strings of computer script that collect and send information about the user, but only in connection with the originating web site.
listservs E-mail-based mailing lists for a group of people with similar interests.
personal digital assistant (PDA) Small mobile devices mainly used for information storage and retrieval, calendars, and address books, although many are now capable of wireless access.
pull technology In client/server applications, the user's browser must request a web page from a web site before it is sent.
push technology In client/server applications, sending data to a client without the client requesting it. The World Wide Web is based on a pull technology where the client browser must request a web page before it is sent.

In most technologically advanced countries, including the United States, using a computer and Internet technology has become an every-day task for many. Few realize that they are conducting research in the broadest sense whenever they navigate, or "surf," the Web. This article focuses on electronic research from two perspectives: information seeking and data collection.

Information seeking is the activity of searching for and possibly finding desired information on-line, using a variety of different strategies and sources. This article reviews search engine technologies, on-line magazines and periodicals, corporate web sites, and listservs, focusing on the kind of information provided and the evaluation of their credibility.

On-line data collection is the actual gathering of research data, mostly by academicians and market researchers. This is a comparatively young and still evolving area. This article reviews a variety of methodologies, including e-mail questionnaires, web site questionnaires, free response data such as participant observation and on-line focus groups, automated data collection such as cookies and monitoring software, and mobile devices. The focus is on the processes of these methodologies, their advantages and disadvantages, and ethical considerations researchers should evaluate.

The purpose of this article is to provide a base knowledge about the multifaceted topic of using computer and Internet technologies for research. This article is not intended to be a "how to" tutorial, but rather review the current state of and knowledge about electronic research.

I. INFORMATION SEEKING

As the Internet grows, so does the amount of potentially accessible information. This information is available in a variety of subparts of the Internet, including

the World Wide Web, Usenet, chat rooms, and e-mail distribution lists. Finding and collecting credible and relevant information through this network of networks can be a daunting task. This section first focuses on issues of information credibility or trustworthiness and usability, then moves on to discuss a variety of popular tools and venues for information seeking, collection and retrieval, including search engines, magazines and periodicals, corporate web sites and Listservs.

A. Credibility Issues

One of the greatest benefits of the Internet is also its greatest pitfall: anyone can post information to a mailing list or create a web site. This is a liberating experience for those who are not affected by the digital divide in that the Internet allows people to have a voice and access to information and viewpoints that are not available through other media channels. The downside, of course, is that a vast amount of available information is unfiltered, unevaluated, and possibly false. Should this worry us? Unfortunately, there is research showing that many people take the printed word at face value. Hence, on-line content that resembles the standard print media is often taken for granted without much questioning. Unfortunately unlike major newspapers and reputable book publishers, the information on most web sites does not undergo critical evaluation by third parties before being made available to the public. Intended or not, the result can be disinformation. The need to establish some criteria for credibility on the World Wide Web has gained importance for both the information provider and the information seeker. The old adage, "consider the source" clearly applies to information on the Internet.

How do we know if the information posted on a web site is credible? To begin with, the reputation of an organization in the off-line environment is not independent of our assessment of credibility in the on-line environment. The *New York Times* and *Wall Street Journal* are trusted sources in print and on-line when seeking credible believable news. Similarly, we should expect reputable scientific journals, government agencies, educational institutions, etc., with an on-line presence to post credible accurate information. The same could be said for the personal web sites of individuals with well-established off-line reputations.

Unfortunately as in real life, appearances in the on-line environment can be deceptive. Sometimes web sites are deliberately designed to mislead users. Once you move away from well-known brands you move into a gray area where determining a credible source from a less than credible source can prove difficult. Caution is clearly an important asset when searching for viable information. For a web site to be an asset to users it needs to contain more than just accurate and credible information. It must be usable. Web designers try to incorporate certain features into web sites to give them more web credibility. Knowing what these usability features are helps users evaluate the credibility of a website.

Usability refers to both format and design options that guide evaluation of content, facilitate navigation through the web site, and display intuitive, user-centered design. A major implication of usability is the perceived ease with which information can be found on a web site. How many clicks did it take to find what you were looking for? How much text had to be processed? Preferred design usability functions include navigation bars, internal search engines, and site maps. Sometimes, the most efficient pages are not the most beautiful ones, even though efficient pages can be visually pleasing. A glossy page with no content or structure will not have web credibility. Some of the formatting guidelines include preparing text that is scannable, as readers scan rather than read on-line, organizing text in an inverted pyramid style with more specific information further down on a web site or linked at progressive levels of depth, and clearly indicating authorship, sources, and currency of information.

Personal greetings, pleasing design, and fast loading times increase usability because they indicate a concern for the user that blends technical and aesthetic factors. Recently a host of very interactive entertainment sites, employing JavaScript, JAVA applets, and streaming video, have been attracting and retaining viewers as increasing broadband access has made these sites much more viable and "sticky" in the sense of users returning to the site. However, the use of such advanced Internet technology will have ambiguous effects. They may not be appropriate to convey just any kind of information, and not all users can be expected to have sophisticated technology and skills to use such "fancy" web site components.

Even though the Web is becoming more regulated, large parts remain uncontrolled. Certain agencies such as WebTrust were established to provide certification of on-line security. Establishing a similar agency providing credibility certifications regarding the content of web sites would facilitate evaluating a web site's credibility, but may be perceived as limiting freedom of speech by some.

In the end, web credibility is easier lost than gained. It is highly advisable for anyone trying to establish

web credibility on their website to conduct focus or opinion groups to test a site before launching it. For the person seeking information on the Internet, the old "don't trust a stranger" advice still holds. Information from unknown sources has to be taken with a grain of salt.

B. Search Engine Technologies

"Seek, and you shall find," an old expression goes. This seldom holds true in the context of Internet search engines. One may seek, but not find, or find, but to an overwhelming extent. Rather than hardware machines, as the term "engine" may convey, search engines are elaborate software programs that employ a variety of strategies for locating information. Search engines are also an example of pull technology, which the user employs to get desired information from "out there" on the Internet to his or her own screen or hard drive. Many searches employ keyword technology which analyzes a search term or phrase, then quickly examines an index of text references that match. This index is typically constructed from information supplied by automated crawlers, often called "bots" or "spiders," which comb the content of a given web site, a domain of sites or the Web itself. Most bots capture keyword references to site content from titles, frequently used nouns, or the first few paragraphs of a document. These references are stored in an index that undergirds the search engine.

Search engines existed before the Internet, but have only become popular since the invention of the World Wide Web, which itself was made possible by the invention of hypertext markup language (HTML) in 1991. One of the most popular sites to begin a search on the Web, Yahoo!, was created in 1994 by two graduate students out of Stanford in the attempt to organize the massive amount of information on the web. The name "Yahoo" itself stands for "Yet Another Hierarchical Officious Oracle." This web site employs human editors who classify and list sites and bundles them in a search engine called Google. Google is fairly effective in identifying relevant results because it analyzes a web site's value by examining what sites link to a page. Thus, Yahoo! itself is only the index for the search engine Google.

Using search engines to find information on the Web has some definite advantages and disadvantages. An important advantage is the easy and mostly free access to information. However, an important disadvantage is the amount of irrelevant information one obtains in a typical search. What is one to do with a "list" of 150,807 links? Search engine companies have responded in a variety of ways in their efforts to reduce irrelevant hits, such as the content evaluation used in connection with Google. Another example for reducing the number of "hits" or returned results on a search list is Alta Vista, which strips out duplicate links and uses filters to remove certain sites such as those using popular keywords in their description though the words have nothing to do with the content of the page. Without search strategies, trying to find credible and relevant information is difficult. A basic understanding of Boolean operators can help. Boolean operators are the "and," "not," and "or" connectors that allow to connect keywords and find sites based on combination or exclusion of these keywords. It is important to note that most search engines provide a link to advanced or help functions that explain how that particular search engine works most efficiently.

More recently search engines such as Ask Jeeves and Netscape answer natural language queries and try to move away from keyword limitations by using human editors who observe frequently asked questions and link these to relevant sites. Whether people realize this or not, when they use an on-line search engine they often rely on the possibly skewed evaluation and categorization of information by other human beings who, having taken the place of librarians, create abstracts and indexes for search engines.

Search engines tend to cover different parts of the Web at different times, even though there is often considerable overlap. One may search with the same term in more than one search engine and come up with different result lists. Due to this, meta-search engines became popular. These search engines examine a number of other search engines and return the results to the information seeker so he or she would not have to perform the same inquiry multiple times. However, depending on the underlying search strategy this is likely to still produce thousands of irrelevant hits.

Finally, some large sites, such as university sites or dot com business sites, e.g., amazon.com, will have their own internal search engines. With these search engines the scope of the search is limited to the pages within the overall site. More often than not these searches are not very sophisticated, but they are still useful in finding specific information from an institutional web site that might consist of hundreds of pages.

C. On-Line Magazines and Periodicals

On-line magazines and journals come in a variety of forms. In some cases the magazine or "zine," as they

are often called fashionably, are only available on-line. Publications like *Slate* and *Salon* are relatively high profile examples of these efforts. Most of these strictly on-line publications tend to be rather narrow in focus and target a very specific audience. Sometimes these publications are put out by volunteers with a very focused interest, or they may be supported by a nonprofit organization. Seldom is there any expectation of profit. Similarly, some professional academic organizations are turning to on-line journals as a way to disseminate information in a timely fashion. This area of on-line publication is slowly growing but not nearly at the pace that some predicted. Though there are e-journals that are developing scholarly reputations, more often than not a quality on-line publication is simply a duplicate of a quality print publication. Almost every major print magazine has an on-line presence. Sometimes these publications reflect what is found in the print version, and sometimes selected articles appear. In every case efforts are made to link these and get readers to subscribe to the print version. Indeed, a subscription may be required in order to access most of the contents of these publications. Apart from advertising revenues, these subscription fees are the publisher's only way of making a profit from the on-line zine. Depending on the topic, the on-line version of a major print magazine may contain unique articles and might feature more timely information since the on-line environment allows one to move quickly, but there are other examples where the on-line version lags perhaps weeks behind the print version. Many of these publications now come in condensed versions that can be received via e-mail, or "pushed," as they are delivered to the consumers' computers or mobile devices upon request up to several times a day.

D. Corporate Web Sites

More and more information about companies is available on-line. A cursory examination of corporate web sites shows that they vary considerably in terms of their sophistication and purpose. In general these web sites can be classified as static (brochure-ware), publishing (one-to-many communication aimed at reducing the costs of producing, printing, shipping, and updating corporate information), interactive (use of web technology, internal newsgroups, and download areas), or transactional (e-commerce involving two-way financial transactions with customers and/or vendors). The incredible increase in corporate information available, either directly from web sites or through increasingly sophisticated databases, has led to a boom in the area of competitive intelligence. Most companies now devote people and resources to collecting and analyzing information about their competitors and customers. The Internet is now used as the main information gathering vehicle in this process. New Security and Exchange Commission filings, patents, news releases, etc., can now be obtained within minutes from competitive intelligence service companies. Although a less reliable source, competitive intelligence operatives also scan message boards and newsgroups.

E. Listservs

E-mail is the most utilized application on the Internet. Some think of e-mail as a one-to-one or one-to-a-few application but there are many ways to use e-mail. One form of e-mail is Listserv. Listservs are email-based mailing lists for a group of people with similar interests. This list is maintained by a Listserv program. Unless the Listserv is closed to the public and requires permission to access, anyone can subscribe to a Listserv mailing list by sending a "subscribe" command to the Listserv administrative address.

Any e-mail letter sent to the list's address is copied and mass-mailed to the e-mail box of every person subscribed to the list. Everyone else on the list can then reply to that letter, and thus, have a way to engage in open discussions with dozens or even hundreds of people on a variety of topics. This is all done through e-mail. For example, any e-mail letter sent to the (fictive) ulla@lists.internic.net address would be copied and mass-mailed to every single person subscribed to the ulla list. But how are people going to subscribe to this ulla list? Actually, there is a second address just to handle all of the commands for this or any list. That second address is the Listserv administrative address, which, in this case, would be listserv@lists.internic.net. Obviously not all Listservs in the world are at the InterNIC server. Indeed, there are thousands of different Listservs, and there are literally tens of thousands of different Listserv lists. Listservs are a very useful and heavily used vehicle of communication. Unfortunately, Listservs can add significantly to the amount of e-mail received and many people find they need to limit subscriptions as e-mail volume increases.

Information received by Listservs is not always credible. The greatest danger are people who claim they are experts, while really they have little background on a topic. Thus, evaluating the content of a Listserv

e-mail is of utmost importance. Just as information obtained in chat rooms, information from Listservs should be taken as indicators or examples, but not as facts until they can be verified through other sources.

II. DATA COLLECTION

Apart from navigating the Internet to find information, the Internet and new technology also provide opportunities to gather an incredible amount of data. The following sections will review the benefits and limitations of a variety of methods and technologies to gather data electronically, including e-mail and web questionnaires, free response methods, automated data collection software, and mobile devices. Finally, this section will conclude with a review of some ethical considerations involved in conducting electronic research.

A. E-Mail Questionnaires

E-mail questionnaires have increased in popularity, but have not replaced traditional mail and telephone surveys. Benefits of e-mail questionnaires in comparison to surface/air mail include reduced cost, quick turnaround time, possibly higher response rate, and direct access. Some of the limitations include awkward format and impaired appearance, little validation, and technology restrictions. The following section will explain these benefits and limitations in more detail, comparing e-mail questionnaires to paper questionnaires where appropriate.

Though traditional paper questionnaires are comparatively cheap as opposed to, e.g., personal interviews, they still require an investment in copying the questionnaire, envelopes, return envelopes, postage, return postage, reminder notification, and additional postage. Especially in international research, postage becomes a high cost factor as mail between different continents is not just slow, it's expensive. The e-mail questionnaire eliminates these costs. The questionnaire can be devised digitally, and is "shipped" or e-mailed to the intended respondent. The respondent returns the completed questionnaire digitally. The researcher may or may not choose to print out the returned questionnaire. Any follow-up reminders also are sent in digital form. The e-mail questionnaire dramatically reduces the survey's cost.

An e-mail questionnaire also reduces project turnaround time. The time normally allotted to printing and processing envelopes is eliminated. In addition, the delivery of an e-mail takes seconds, or at worst only hours depending on network congestion and e-mail delivery protocols. The standard paper mail questionnaire may take days or even weeks to arrive, depending on the distances involved. Turnaround time is also increased, as people tend to lay aside the paper questionnaire, while respondents typically complete the e-mail questionnaire when looking over it a first time. Recent research suggests that the vast majority of e-mail questionnaires are returned within 48 hours of the time they were sent. Finally, while the typical survey researcher has to wait several weeks before sending out a reminder for the paper questionnaire, with e-mail questionnaires this waiting period is reduced to only days.

Two other potential benefits of e-mail questionnaires are closely related. Response rate and direct access are issues of high concern for any researcher. The higher the response rate, the more reliable the collected data. However, especially in the case of a paper questionnaire, the researcher can never be sure if the intended respondent completed the questionnaire. With e-mail, the questionnaire goes directly to the person. No intermediaries including secretaries or family members are likely to receive and complete the questionnaire. E-mail questionnaires typically yield higher return rates because the targeted person was reached, but also because the process of completing an e-mail questionnaire requires less effort than the completion and mailing of a standard paper survey.

However, e-mail questionnaires are not without their drawbacks. E-mail questionnaires, especially those of the academic variety, can appear overly long, burdensome, and intimidating as opposed to quick "opinion" surveys posing only a few questions. One of the major limitations is the "appearance" of the e-mail questionnaire. Research has shown that the way a questionnaire looks, the way in which questions are arranged on the page, the order of the questions, and even the color of the paper and the type of the postage stamps can all have effects on return rates. The typical e-mail questionnaire cannot control for these factors. Questions usually appear in one long list, black on white, with no formatting manipulations, indention, easy boxes to check, or similar customary questionnaire components. This makes the questionnaire appear dull, long, and burdensome. People are more likely to abandon the unfinished reply when faced with such a task. Should they intend to go back to the questionnaire later, it usually requires that they start over, while a paper questionnaire can be completed in several sessions.

Much of the formatting issues are connected to technological restrictions. The plethora of current

e-mail software, operating systems, and even fonts can impair rather than support research activities. A researcher must build his or her e-mail questionnaire so that even a respondent with older software and hardware can participate in the study. Although new HTML-based e-mail allows researchers to construct questionnaires that are much more user friendly and lively, not all e-mail programs or systems can take advantage of this new software.

Another restriction of this methodology is connected to personal privacy issues. Although e-mail provides direct and immediate access, many people perceive unsolicited e-mail as an invasion of their privacy. Also, using an e-mail questionnaire requires all respondents to have e-mail accounts that they regularly check, and for the researcher to have access to these addresses. Clearly, the vast majority of potential respondents do not have e-mail addresses or e-mail access, although that is changing, especially in the United States and Europe. Finally, there is simply no such thing as a universal e-mail directory and while there are places to obtain e-mail addresses for groups of people, most companies, ISPs, and universities attempt to protect e-mail address databases, for obvious reasons. Given these constraints, it is not surprising that most surveys are still conducted with traditional methods simply because using an e-mail questionnaire excludes too many in the target population.

A third limitation concerns the validity of the methodology itself. The parameters and protocols of administering standard paper questionnaires or telephone surveys are familiar to most researchers. E-mail questionnaires, on the other hand, are still comparatively new. Reliability and validity concerns have not been completely answered and as of date of publication, not much is yet known about the characteristics of respondents and nonrespondents. Some preliminary research suggests that mass mailing for e-mail questionnaires has a negative effect on response rate. However, personalized e-mail questionnaires may overcome some of this difficulty, though even with personalized e-mail the researcher can never be absolutely sure who actually completes the questionnaire. More research is needed before we can conclude that the advantages of e-mail questionnaires outweigh their disadvantages.

B. Web Site Questionnaires

Marketing and academic researchers are relying more on on-line web questionnaire surveys as access to the Internet increases. New software allows relatively painless creation and deployment of on-line surveys. Research participants access the web site, fill out questions on-line, and "submit" their answers to a database or an e-mail address. New software programs even allow for these answers to be saved into the configured databases or spreadsheets of sophisticated statistical software, allowing for immediate analysis without the arduous and error-prone task of data entry. Some of the benefits of web questionnaires become apparent immediately: reduced cost for the survey administrator, reduced time, availability, and flexibility. Other advantages include visual design features, which might include enhanced graphics, audio and even video, especially if conducted internally through a corporate intranet, an Internet with limited access privileges. Some of the limitations of e-mail questionnaires apply, including some technological restrictions. Other limitations include security of information, cost demands for survey participants in countries with high Internet access charges, as well as researcher and participant skill demands.

Just as with e-mail questionnaires, the web questionnaire is already in digital form, minimizing costs as long as hosting hardware, software, and Internet access are freely available. If an adequate sample of respondents' e-mail addresses can be obtained, contacting these individuals is relatively easy. This saves both time and money. Time is also reduced because instant access to the questionnaire is granted, and manual data entry, which also runs the danger of human error, is eliminated in the more sophisticated software programs. Distributing the web address or uniform resource locator (URL) to participants is easy by embedding a link into the "request to participate" e-mail. Clicking onto this link will automatically open up a web browser and lead participants to the web questionnaire web site. However, it is possible that some Internet users may find an on-line survey through a search engine and try to complete the questionnaire. Incorporating password protection eliminates this problem.

Web questionnaires also have some other advantages that distinguish them greatly from traditional paper questionnaires. Since the Web is a visual medium, the questionnaire can make use of careful color-use, include graphics or explanations for questions easily, and array questions in a spatially interesting manner. In addition, a web questionnaire can include sound or video, something the paper questionnaire can never accomplish. In some cases this might facilitate certain types of research, for example, because participants don't have to physically come to a certain location to watch a video and then

fill out a response questionnaire. Depending on the quality of the video required for the study, the speed of the connection, etc., respondents can access the questionnaire from a distance and at a time convenient to their schedule. Web questionnaires can also include complex skips between questions when needed, e.g., in the case of contingency questions or in repeat-measures situations where the asking of a question is based upon the previous response, eliminating time-consuming reading of questions that do not pertain to the respondent. These skips are invisible to the respondent in the on-line questionnaire while often proving to be confusing and problematic in traditional mail or e-mail surveys.

An additional feature of web questionnaires is that monitoring software can measure, for example, how long a person takes to answer a question, how many times he/she changes his/her answer, or even how the mouse is moved across the screen. This kind of sophisticated monitoring allows researchers to fine tune these assessments and detect problems that would prove difficult to pinpoint in traditional paper and pencil data collection. This provides an on-line questionnaire with an incredible amount of flexibility since it would be possible, for example, to test the survey with the first group of respondents, analyze the pattern of response, and make appropriate changes in minutes without incurring the costs associated with changes in traditional formats. Of course, in this situation like all other research using human subjects, informed consent means must be employed and consent must be assured before beginning the research, ascertaining that the research goal and procedure was explained to participants who then have agreed to participate in the study.

There are a number of obvious limitations to web questionnaires. First, on-line questionnaires require participants to have access to the World Wide Web. In many countries outside of the United States, access to the Web is still charged through local telephone companies on a minute-by-minute basis. Completing an on-line questionnaire may thus, save the researcher money, but cost the participant. Second, if sophisticated features such as sound or video are included, the person administering or creating the questionnaire takes for granted that the person completing the questionnaire has the necessary connections, hardware, and software to access these questionnaire components. One solution to this problem is to require that the respondents download the required software through links provided at the beginning of the questionnaire. However, often a person skilled at navigating the Internet does not necessarily know how to download and install software, or, if accessing the Web from a computer other than their own, downloads may not be allowed by the network administrator.

While web questionnaires can incorporate colorful and innovative design features, some of these features may prove to be problematic to the main goal—accurate question completion. Some design features may be confusing for certain people. For example, some handicapped people, such as the visually impaired, may have difficulties distinguishing colors, reading small fonts, scrolling up and down or right and left, clicking the mouse into a small space to check an answer option, or printing out a questionnaire in Braille format. Research consistently shows that after the age of 45 many people have difficulty with text colors. Technological and skill limitations may also apply, especially when surveying groups other than college students or employees in the office environment. The elderly, certain socioeconomic groups, individuals for whom English is a second language, or respondents from technologically less developed countries are examples of groups that may have difficulties using web questionnaires.

A final major limitation to web questionnaires may involve concerns about privacy. Many web-based questionnaires give the appearance of anonymity. Respondents can visit the URL from their home, office, or a computer lab and answer the questions. However, it is possible to track and identify these access points. In most cases issues of privacy and trust may not seem to be an issue for a respondent. Yet, if the questionnaire asks for information of a personal nature and requires some kind of identifying code, or e-mail address to access the web site, then such concerns might quickly arise. Just like in the off-line environment, researchers need to think carefully about the kind of data they really need to collect and eliminate unnecessary questions of a personal nature. It is also the researcher's responsibility to build trust with the participants to ensure them their information will not be abused, cannot be stolen, and will not be related back to them. Most researchers in the off-line environment provide participants with a copy of the informed consent sheet participants must sign to assure their willingness to participate in the study. An equivalent to this informed consent sheet should be employed in the on-line context.

C. Free Response Data

1. Participant Observation

The Internet is much more than just e-mail and web sites. Indeed, hundreds of thousands of people each

day engage in Usenet, an asynchronous stored-e-mail based environment, and real-time chats, and these numbers are increasing. Most of these chat and Usenet groups utilize a text environment, but some moderate growth in three-dimensional and graphically intense environments, such as Alpha World or The Palace, have been measured and observed. But even these environments utilize text for communication between participants. The purpose of chat rooms is to exchange information with people who are interested in similar topics. Usenet operates on a similar principle. People join Usenet groups that are centered around certain topics of interest. However, rather than using synchronous communication in chat rooms, Usenet groups are asynchronous. Participants write e-mails that are stored in "trees" with different discussion topics making up different "branches." Each e-mail message is represented by its subject line, and the visual arrangement of these lines with indentations below the original message to which subsequent messages have replied resembles branches on a tree. Participants can access and contribute to their Usenet group any time. Many Usenet groups are support groups.

Chat and Usenet are places in cyberspace where people meet virtually to exchange ideas and opinions. Sometimes personal information, including personal preferences or addresses, is exchanged. This information is of potentially great interest to researchers. Reading conversations in chat rooms or reading Usenet e-mail threads can be like listening in on a private telephone conversation. In order to monitor and conduct their research, researchers usually join a group or chat. In both environments, people that simply read the posted messages without contributing are considered "lurkers." Researchers sometimes lurk, and sometimes contribute, which makes them both participant-observers and/or observers.

Participant observation is a term mostly attributed to qualitative and ethnographic research that seeks to understand and inquire from within, rather than to measure and predict, as quantitative research does. An important issue for the participant observer both on- and off-line is to which degree a researcher becomes integrated into the observed group. Not announcing that one is observing for research purposes immediately brings up ethical issues. Announcing this purpose may alter others' behavior and change the group dynamics, thus resulting in a tainted version of normal group behavior. On-line, researchers tend to follow the ethical guidelines outlined by their professional associations or universities and announce their presence and purpose, such as the American Psychological Association's Ethical Principles in the Conduct of Research with Human Participants. They then work on becoming integrated into the group by participating in the everyday discussions of the group. In the case of Usenet groups they may actively participate for several months or even years. While informing the group of one's research purpose is often ethically required, it does not inform those who failed to read the message or joined the group long after the message was posted. Yet, announcing the research purpose regularly may disrupt the group and prove a distraction to participants. An on-line participant observer has to carefully consider these and other related methodological and ethical issues such as invasion of privacy.

Participant observation usually is a fairly long-term endeavor and this notion of ongoing longitudinal research, a process that can stretch from several months of continuous observation to several years, certainly applies to the on-line environment as well. The benefits of conducting participant observation on-line include reasonably easy access to a great variety of groups or Usenet communities and the ability to easily capture message text for analysis. E-mail messages are archived and can be printed or retrieved later. Chat room conversations can be copied and saved into word processing programs, also to be printed or retrieved for later analysis. Information gathered is analyzed based on content analysis, often looking for themes of ideas or behaviors through the text provided in both chat and Usenet groups. Though personal information is disguised in research reports, the members of the group most likely are still able to identify the person described. Privacy issues are of great concern in on-line participant observation research.

2. Online Focus Groups

A focus group is a somewhat informal technique of free response data collection that can help assess needs and feelings about issues, political candidates, commercial products, etc. In a focus group, the researcher brings together, usually, five to nine users to "focus" on some issue or concerns. Online focus groups typically last about 60–90 minutes and are run by a moderator who maintains the group's focus onto the discussion topic. Focus groups often bring out users' spontaneous reactions and ideas and allow to observe some group dynamics and organizational issues. Online focus groups operate similarly to ordinary focus groups except online interaction is substituted for face-to-face (FTF) interaction. They also employ technology assistance, passwords, etc. that are not found in FTF groups. Online focus groups raise a host of new issues includ-

ing the effects of differences in respondents' computer skills, familiarity with technology, access, cognitive skills associated with composing answers at keyboards, etc. There are also technology issues, such as specialized software, or hardware configuration, to deal with. Despite these issues, there are clear advantages in terms of capturing transcript data in real time and shortening time to analysis.

D. Automated Data Collection

1. Cookies

Information about Internet users is collected in a variety of ways. Visitors to a web site reveal an amazing amount of information about themselves including the browser they use, the previous site they visited, the search engine employed to find the visited site, connection speed, etc. In addition, the web site may deposit a "cookie" into the visitor hard drive in order to collect information. A cookie is a short string of computer script that collects and sends information about the user, but only in connection with the originating web site. Cookies enable functions such as customizing and personalizing web sites, including banner ads and personal greeting; shopping card applications including storing of credit card information and frequent purchasing rewards; games including remembering scores and skill levels. Cookies can also track navigation patterns through a web site or from one web site to another, length of visit to a certain site, type of links selected, text entered, items reviewed or bought, and even the movement of the mouse over the screen. Clearly the use of cookies to collect certain information raises ethical concerns and web sites employing cookies should articulate their purposes.

2. Other Monitoring Software

An increasing number of companies use monitoring software to track employees computer use, web activity, and even the content and destination of e-mails. The software resides on the PC or the server and can collect a variety of data from screen shots and keyword searches of e-mail to the employee's individual keystrokes. These software programs usually e-mail reports of monitoring activity to a supervisor several times a day. Although one might raise certain ethical issues regarding such monitoring of employee behavior, United States courts have generally found such activities to be legal.

Other software programs are employed for gathering information at nodes, large connection points on the Internet. These monitoring programs are for the most part benign in that they gather data in order to monitor and route Internet traffic more efficiently. However, this software can also be used to monitor ("sniff") the contents of Internet packets, the broken apart components of messages, as they move through these nodes. Recently, the United States government has been criticized for proposing the deployment of a fairly sophisticated e-mail monitoring software program called Carnivore. Legal questions regarding government use of such monitoring and interception software are still being worked out in the courts and the United States Congress. Policies regarding the monitoring of personal e-mail, web site postings, web site access, etc., vary considerably between nations. These differences in policies are quite pronounced and promise to lead to interesting discussion between national governments as e-commerce develops further.

E. Mobile Devices

More and more we find that mobile devices such as laptops, Personal Digital Assistants (PDAs), etc., are being used to collect and capture data. For years these devices have been used by industrial and retail personnel to access inventory and order products. In the last couple of years this area has exploded in use as the costs of the devices has dropped. New software is available, and it has become increasingly easier to link these devices to wireless telecommunication networks.

Personal digital assistants or handheld computers are small mobile devices approximately the size of a notepad. They are mainly used for information storage and retrieval, calendars, and address books although many are now capable of wireless access. Probably the best known versions are the PalmPilot, Visor, and various Windows CE devices. PDAs can be used for much more than the functions listed above. Through a wireless connection or by "synching" the PDA with a computer, these devices allow users to receive and respond to e-mail, do text editing, download news and e-books, view the latest inventory information and of course, play games. A variety of additional software is available as freeware or shareware and is able to be downloaded from the Internet, often for free. One can create questionnaires, input data, record observations, and quickly upload this information to a database for statistical analysis. Thus, the PDA opens a new arena for electronic data gathering.

Thus, PDAs provide researchers with an increasingly sophisticated and more flexible tool for data collection in that their small size allows them to be trans-

ported easily and they are less likely than laptops to be damaged. PDAs have become more intuitive and typically employ a special pen for input rather than a keyboard. For the most part they are not especially intrusive and are becoming much more common. Information and data gathered/entered is in digital form, allowing for quicker processing time.

However, using PDAs for research also has some drawbacks. While dropping in price and yet increasing in capability, they are still relatively expensive if the researcher needs to provide each respondent with a PDA for longitudinal research. The screens are of necessity small and can be difficult to read. Finally, most software for the development of questionnaires requires considerable customization.

It is clear that we expect to see a rapid increase in the use of mobile devices to gather, access, and report data. PDAs can be combined with modems for wireless connections, GPS devices for purposes of mapping and related studies, etc. Wireless access with small mobile devices is an exciting area for future development.

F. Ethical Considerations

As the Internet opens new opportunities, it also brings about new issues of concern. Codes of conduct, public policies, and ethical issues of all sorts are being redefined and debated in public and private forums. "Ethical behavior" is generally defined as following agreed-upon codes of conduct that govern one's understanding of right and wrong. In the arena of the Internet, what is right and what counts as acceptable and unacceptable behavior is being redefined and negotiated. This results in ethically ambiguous situations for both the typical user as well as for researchers who need to take measure of behaviors in this new environment.

While there are a variety of ethical concerns in this environment we have focused in this review mainly on issues of privacy associated with the collection of data. Cookies, intelligent agents, bots, and other net software are currently being used to gather information about Internet users. In most cases, this is restricted to demographic information, credit card data, and generalized buying behavior. However, there are proposals to correlate this information with all sorts of other data sources, such as health information, into much larger and more comprehensive databases. Traditional brick-and-mortar companies currently collect much of this information, but it is not collected so easily or with the distinct likelihood of being scrutinized at such a detailed level of analysis. Both on- and off-line, this information is shared with business partners as companies attempt to target consumers with more and more personalized messages. While most corporate web sites provide privacy statements and include information on procedures for collecting data on their web site, very few customers actually read this detailed information.

Few Internet users are aware that their activities, buying patterns, etc., can and are being monitored in such detail by interested third parties. This group of interested individuals consists mostly of marketing firms along with a few academics. These researchers use cookies, monitor web activities, monitor chat rooms, Listservs, and Usenet support groups. Personal information, opinions, and on-line behavior are subject to recording. Many times, they do not advise the people observed of the observation. Many of these researchers contend that the Internet is a public forum, and thus they do not need prior or informed consent to record and systematically study on-line activities. At this time, several United States congressional committees are exploring this issue while some states and municipalities are moving toward the creation of policies to govern data collection. The European Union (EU) has recently developed a set of privacy policies to which all companies engaged in e-commerce in the EU will need to adhere, but in the United States advertising and marketing firms are pushing a "voluntary" solution that will be self-policed by the companies themselves. Most universities have moved toward rather strict codes of ethical data collection, but this is still under discussion in a fairly large number of locations and researchers themselves must draw their own line as to what constitutes the ethical collection of data in the online environment.

III. CONCLUDING REMARKS

The previous pages have reviewed the current state of and information about electronic research, defined here as the use of computer and Internet technologies for information seeking and data collection. The authors distinguish between information seeking, the searching for hopefully credible and relevant information, and data gathering, the collection of sometimes statistical, sometimes contextual data for the purpose of analysis. Credibility issues were reviewed in connection with information seeking, as were ethical considerations with regard to using Internet- and technology-related methodology.

Though computers and Internet technology have become ubiquitous in technologically advanced coun-

tries such as the United States, the field of electronic research is still in its infancy. Few established guidelines exist. Internet users and researchers often depend on their own wit to collect and assess information and data they acquire by using the Internet. The next few years will be crucial in establishing some of these internationally applicable guidelines that will protect users' privacy while providing research opportunities, and protect freedom of speech while guarding against misinformation. Guidelines for electronic research will help settle the as of yet wild but exciting domain of cyberspace.

SEE ALSO THE FOLLOWING ARTICLES

Electronic Mail • Ethical Issues • Internet, Overview • Marketing • Mobile and Wireless Networks • Privacy • Search Engines

BIBLIOGRAPHY

Alexander, J. E., and Tate, M. A. (1999). *Web wisdom. How to evaluate and create information quality on the web*. Mahwah, NJ: Lawrence Erlbaum Associates.

Babbie, E. (1998). *The practice of social research*. 8th Edition. Belmont, CA: Wadsworth Publishing.

Elmer, G. (1997). Spaces of Surveillance: Indexicality and solicitation on the Internet. *Critical Studies in Mass Communication*, 14 (2), 182–191.

Nielsen, J. (2000). *Designing web usability: The practice of simplicity*. Indianapolis, IN: New Riders.

Reid, E. M. (1996). Informed consent in the study of online communities: A reflection on the effects of computer-mediated social research. *The Information Society: An International Journal*, 12 (2), 169–174.

Stanford University (2000). Web Credibility: Research & Design. Available online at http://www.webcredibility.org/.

Virtual Surveys, Limited. (2000). Online research—some options, some problems, case studies. Available online at http://www.virtualsurveys.com/papers/asc.htm.

Resistance to Change, Managing

Ken Hultman
Synthesis, Inc.

GLOSSARY

behavior Actions based on facts, beliefs, feelings, and values.
belief A subjective assumption, conclusion, or prediction.
change Doing something new or differently.
fact An objective reality that can be proven with complete certainty.
feelings Emotional responses.
readiness A state of mind reflecting willingness or receptiveness to change.
resistance A state of mind reflecting unwillingness or unreceptiveness to change.
value A conception about what is important in life.

THE TOUGHEST CHALLENGE of organizational leaders today is to manage at the speed of change. With the breathtaking pace of technological advancement which will only continue to escalate, and the doubling of knowledge every 5 years, leaders face tremendous pressure as they try to gain support for change. Regardless of how good or necessary a change may be, resistance should be expected. A survey of 500 executives conducted by William Schiemann (1992) concluded that resistance was the main reason why organizational changes fail. A study by Hammer and Associates found that 60% of the reengineering projects that failed were due to employee resistance (Moomough, 1999). While preventing resistance completely is an unrealistic goal, the ability to manage resistance effectively has emerged as an essential skill. This article will help the reader do that by offering a conceptual framework for understanding resistance and also practical suggestions for assessing and overcoming it.

I. UNDERSTANDING RESISTANCE

When people talk about resistance, they usually refer to specific behaviors observed in others. Thus, it is commonplace to hear someone say, "He's resisting these new procedures," or "She's refusing to go along with the changes." Behaviors used to resist change fall into two categories: *active* and *passive* resistance. Some examples of both types are included in Table I.

Behaviors such as these indicate that people are resisting change; however, they do not explain *why*. The reason for this is that behaviors are external manifestations of internal issues within a person's mindset. In other words, behaviors are symptoms, while mindset issues are causes. These distinctions lead to a definition of resistance. *Resistance* is a state of mind reflecting unwillingness or unreceptiveness to change in the ways people think and behave. Resistance can be contrasted with readiness, which is a state of mind reflecting willingness or receptiveness to change. Resistance manifests itself behaviorally either by active opposition to change or by attempting to escape or avoid it; readiness is manifested behaviorally either by active initiation of change or by cooperation with it.

Readiness is not the opposite of resistance, since an absence of resistance does not necessarily mean that someone is receptive to change. Other factors, such as lack of information, lack of knowledge or skill, or an immediate need to attend to other matters,

Table I Active and Passive Resistance

Active resistance	Passive resistance
Being critical	Agreeing verbally, but not following through
Fault-finding	Failing to implement change
Ridiculing	Procrastinating/dragging feet
Appealing to fear	Feigning ignorance
Using facts selectively	Withholding information, suggestions, help, or support
Blaming/accusing	Standing by and allowing the change to fail
Sabotaging	
Intimidating/threatening	
Manipulating	
Distorting facts	
Blocking	
Undermining	
Starting rumors	
Arguing	
Raising objections	

could interfere with readiness. Nevertheless, anything which causes resistance can be expected to undermine readiness at any point in time.

The most important factors making up a person's state of mind are his or her facts, beliefs, and values. Facts are objective realities that can be proven with evidence ("We tried that before."), while beliefs are subjective assumptions, conclusions, and predictions ("It didn't work then; it won't work now."). Values are people's conceptions about what is important in life ("I want to be open and honest with everyone about this.").

A clear distinction between beliefs and facts is essential, because people often state beliefs as facts ("I know I'm right, so what's the problem?"). Facts can be proven with empirical evidence, while beliefs are subjective assumptions, conclusions, or predictions. In the thinking process, beliefs are more important than facts because they represent the meaning attached to factual information. We all know that facts are subject to distortion; two people exposed to the same set of facts can arrive at entirely different conclusions. Nevertheless, most adults have beliefs held with such high confidence that they treat them as facts. Beliefs of this kind are less amenable to change because they are regarded as irrefutable. When this happens, people are assuming that their beliefs and reality are identical. In one sense there is some truth to this. People's beliefs are their way of understanding themselves, other people, and the world around them. In essence, their beliefs become reality for them. It is indeed a mistake, however, when people conclude that their beliefs represent the one true concept of reality.

Since beliefs are subjective variables and not objective realities, their usefulness does not depend on whether they are popular but on whether they are *viable* (workable, helpful, and useful). Beliefs should be evaluated by their results, not by how good or bad they may sound. Viable beliefs will be accurately grounded on facts, making them reality based, while nonviable beliefs will lack empirical evidence.

Thinking is a cognitive process, guided chiefly by one's facts and beliefs. In contrast, feelings have to do with emotions. Without feelings people might be able to think, but they would respond like robots. There is a strong relationship between the cognitive process of thinking, on the one hand, and feelings on the other. By themselves, facts are neutral—they do not evoke any feelings. One person could look at a set of facts and ask, "What's all the fuss about?", while another person bellows, "The sky is falling!" It is the assumptions, conclusions, and predictions derived from facts that trigger feelings. The reason for this is that the human mind cannot distinguish between a real and an imagined threat. If people interpret an event as a threat, for example, their bodies release adrenaline and they experience the emotion of fear, whether or not there is anything to be afraid of. Those feelings, in turn, influence people's decisions and behaviors for better or for worse.

In addition to facts and beliefs, values are a crucial part of a person's state of mind. Once established, values become the criteria for making decisions and setting priorities. When evaluating a number of alternatives, people eliminate the ones that do not measure up to their standards. Some common values in organizations today are quality, learning, customer satisfaction, mutual interests, integrity, collaboration, and teamwork. Values are to people what instincts are to animals. Since animals can think and feel in varying degrees, it is the ability to establish values that makes humans truly unique. Without this capacity, people could not be held accountable for their actions.

People have a vested interest in their values. Everyone wants to believe their values are the best ones or even the only ones. Since values are subjective variables, however, they cannot be proven right or wrong. As with beliefs, therefore, values should be evaluated by their viability, not by how noble or lofty they may sound. Rational beliefs and the positive feelings they produce lead to values that foster growth; irrational beliefs and the negative feelings they produce lead to defensive values.

Any belief, value, or behavior that has been previously successful in meeting a person's physical, emotional, psychological, and spiritual needs will resist change. Beliefs, values and behaviors which have been

consistently reinforced through experience will be the most resistant to change. In contrast, beliefs, values, and behaviors which are less reliable in meeting needs will be more amenable to change. The degree of personal investment will be greater for the former than for the latter. People develop such a strong commitment to their most reliable beliefs, values, and behaviors that they have trouble thinking about themselves apart from them. In reality, people and their beliefs, values, and behaviors become synonymous.

When people perceive that one of their established beliefs, values, or behaviors is being threatened, they experience fear and assume a defensive posture. The intensity of the fear can vary from mild to extreme, depending upon the degree of the perceived threat.

II. POSITIVE AND NEGATIVE RESISTANCE

Change can be defined as doing something new or differently. By itself, change is not inherently good or bad. Any change will make people different from what they were before. There is no such thing as a change with a neutral impact: people will either be better or worse off because of it. Anyone suggesting change should always be aware of this. Although change can be evaluated by its consequences, it is impossible to know in advance how a change will turn out. After taking all the relevant factors into consideration, there are times when it is prudent not to change.

Similarly, there are other times when resistance is the best action. Unfortunately, the word *resistance* often has a negative connotation. This is a misconception. Sometimes resistance is the most effective response. If people's beliefs, values, and behaviors provide them with constructive ways of meeting needs, then it is adaptive and healthy to hold onto them and resist change. Some changes could disrupt this process and cause the person to become disorganized and less productive. In these situations it is in a person's best interests to resist change.

Thus, there are times when resistance is a *problem* and times when it is a *solution*. The focus of this article is on resistance which is considered a problem. Section III discusses ways to assess this type of resistance.

III. ASSESSING RESISTANCE

Finding the causes of resistance requires the understanding of people's facts, beliefs and values. Since a person's facts, beliefs, and values cannot literally be seen, the role they play in creating resistance can be difficult to isolate. Fortunately, there is a powerful source of information to help with this task: observation of what people do and say. Everything someone does or says provides clues about the role these variables play in resistance. Therefore, when observing what someone does, ask the question, "What fact, belief, or value is being reflected by what this person is doing?" Similarly, when listening to someone, ask, "What fact, belief, or value is being conveyed by what this person is saying?" By consistently asking these questions and putting the data together, not only will people be understood more fully, but the causes of resistance will be more effectively located when they occur.

It is risky to try to assess the causes of resistance by observing only what a person does. It is easy to misinterpret other people's behavior. Watching people in action definitely supplies clues, but additional information from what they say is almost always necessary to achieve an accurate assessment. While it is possible for resistance to stem from a single fact, belief, or value, this is unusual. More commonly, a number of facts, beliefs, and values are linked together to form a pattern, so it is important to gather all relevant data.

A. Things People Say Indicating Possible Resistance

Here are some things people say that indicate possible resistance to change, classified as facts, beliefs, and values:

1. **Facts**
   a. "My doctor told me I shouldn't subject myself to too much stress."
   b. "All my friends are in this department."
   c. "I never had a course in accounting."
   d. "That isn't in my job description."
   e. "They just filed for bankruptcy."
   f. "Other companies that buy supplies from them say they never deliver on time."
   g. "You said you weren't going to make any more changes this year."
   h. "The turnover rate is already 25% higher this year."
   i. "We haven't received the training we were promised."
   j. "Why should we do that? We haven't received a salary increase in three years."
2. **Beliefs**
   a. "I'm too busy to do this."
   b. "I'm a follower, not a leader."
   c. "Yes, but. . . ."

d. "He just tells people what they want to hear."
e. "He makes those changes just to harass us."
f. "He's more concerned with protecting his security than anything else."
g. "Watch out for him."
h. "In this company, it's not what you know but who you know that counts."
i. "Here we go again."
j. "That will never work here."

3. Values

a. "It's not important to me to have more authority."
b. "That's not on my list of priorities."
c. "It's essential that we're honest with each other."
d. "What you think really doesn't matter to me."
e. "Working with them is something I'd rather not do."
f. "Our mission should be. . . ."
g. "What we need to emphasize is. . . ."
h. "In the future, I'd like to see the organization. . . ."
i. "We should be devoting more energy to. . . ."
j. "Who cares what the goals are? I just do my job."

B. Common Causes of Resistance

Although there are many possible causes of resistance, the following eight reasons are the most common.

1. They Believe the Change Process Is Being Handled Improperly

When change is being proposed, people invariably have three questions: (1) Why? (2) How will it affect me? and (3) What is in it for me? Answering these questions effectively is crucial to preventing or minimizing resistance. Thomas Head (2000) said that resistance is usually caused by the change process (the how) instead of the intervention itself (the what). In addition to not having these questions answered, people resist change when they do not have any input into the decision, they do not like how the change was introduced, the change was a surprise, the timing of the change was bad, or they felt manipulated or deceived by management. People react to things like this with anger and resentment, because the methods used go against their values and violate their need for respect. Today, employees expect to have their views considered and to be treated with dignity. Therefore, people should be asked for input if:

- They will be affected by the change.
- Their commitment is needed to implement the change.
- They have information or ideas to contribute.
- They expect to be involved.
- They could learn from the experience.
- The base of support needs to be expanded or strengthened.

When initiating change, it is important to identify rewards that relate to employee's values, for example, money, benefits, and new opportunities, and to build them into the change process, so employees will have a greater sense of ownership. People will be much more likely to support the change and help make it work if there is something in it for them.

2. They Believe There Is Not Any Need for the Change

Beckhard and Harris (1987) suggested that the degree of resistance is related to three factors: level of dissatisfaction with the status quo, desirability of the perceived change, and the practicality of the change. One of the most common causes of resistance is that people are content with the way things are now—if it's not broken, don't fix it. As long as this is true, change will only be viewed as unnecessary or negative.

Sometimes change is needed to avoid or escape a harmful situation. Some examples are bankruptcy; a hostile takeover; or a decline in market share, profit, revenue, productivity, quality, morale, competitive position, and so on. A lot of the changes being introduced in American organizations now are in response to increased competition from other countries. Leaders maintain these changes are necessary to survive, but a lot of employees simply do not believe this, since they cannot literally "see" the competition. Many of these employees are in denial and need to face reality or be left behind.

3. They Believe the Change Will Make It Harder for Them to Meet Their Needs

As such, they believe the change will make things worse, rather than make things better. Brehm's (1966) research showed that people resist change that they believe represents a loss of personal choice. In situations like these, facts tend to be less significant than the beliefs (assumptions, conclusions, and predictions) derived from them. For example, when managers talk about making the work more "efficient," employees often interpret this to mean that they will

be doing more work. Also, if a change is presented as making the work easier, employees worry about positions being eliminated. Such concerns must be addressed to gain support for change. It is natural for people to think about how they are going to be affected by change, but unviable beliefs about bad things happening are damaging because they are often based on inaccurate, incomplete, or mistaken information. These beliefs lead to fear and to values that emphasize protecting oneself against the perceived threat.

While it seems like ancient history now, even though it was only a few years ago, this was one of the main reasons people resisted learning about computers. Protesting that computers would make their work harder rather than easier, they continued using conventional equipment until they had to choose between becoming computer literate or being unemployed. For many of these people, claiming that computers would make their work harder was actually only an excuse to cover up their fear of the new technology.

4. They Believe the Risks Outweigh the Benefits

People are simultaneously motivated to meet their needs and to prevent bad things from happening, so accurately anticipating or predicting the probable outcomes of their actions is extremely important. Since people can never be absolutely sure how their actions will turn out, there is risk associated with everything they do. There is risk in change, but there is also risk in not changing ("What will happen if I change?", "What will happen if I don't change?"). Staying the same does not make people immune from risk. Nevertheless, people vary greatly in the degree of risk they perceive in a situation. Some people see gloom and doom around every corner, while others see a silver lining. In a team meeting where members are discussing a possible change, some may see a great deal of risk, while others may see little or no risk. The discussion can turn to conflict, as members assert their competing perspectives.

As shown in Fig. 1, the belief continuum, the degree of perceived risk can range from none at all to extremely high. When contemplating change, an appropriate degree of concern is necessary to avoid mistakes, but people are capable of either underestimating or overestimating risks. Those who underestimate risks often regret their action, while those who overestimate risks often regret their inaction. It is natural for people to consider the worse case scenario, if only briefly. It can become a serious problem, however, if people treat the worse case scenario as an imminent reality when the probability of its occurrence is remote.

To avoid either underestimating or overestimating risks, it is essential to have accurate facts and viable beliefs. Effective risk management, contingency plans, and feasibility studies would be impossible without such data. Inaccurate facts and unviable beliefs undermine efforts to plan and implement change. People who underestimate risks ignore relevant concerns, while those who overestimate risks will have too much concern. Either way, organizational growth is hindered.

In a cost/benefit analysis, facts are actual costs, beliefs are anticipated costs, and benefits are values. Expectancy theory holds that people will be motivated when they believe that they can perform at the level necessary to attain rewards, and that these rewards are worthwhile (Lawler, 1973; Vroom, 1964). Something is only a benefit to people if it relates to their values. If people really want something, they will be willing to pay more for it (actual cost) and take more risks to get it (anticipated cost). Figure 2 presents the following four scenarios, which describe the interplay between facts, beliefs, and values in a cost/benefit analysis:

- **Low cost and low benefit.** The expected reaction here is indifference. Under these conditions, the perceived benefit would have to be increased in order to generate any interest in change.
- **Low cost and high benefit.** Conditions are favorable for change to take place.
- **High cost and low benefit.** Resistance to change can be expected. To foster change under these conditions, the perceived benefit would have to be increased to justify the high cost.

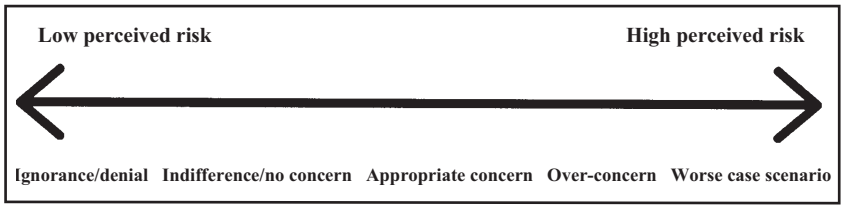


Figure 1 Belief continuum.

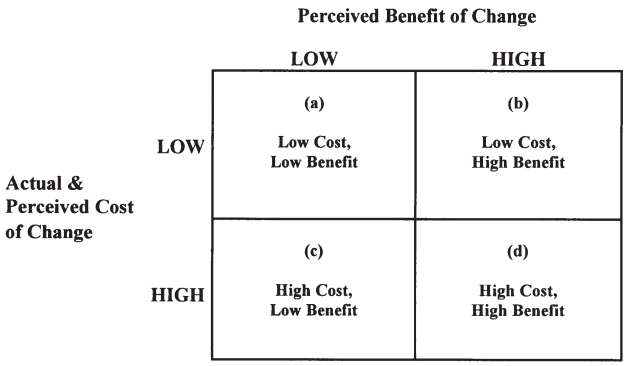


Figure 2 Cost/benefit analysis of change.

- **High cost and high benefit.** These conditions typically result in an approach-avoidance conflict. The decision to change would be difficult, as people weigh the high costs against the high benefits. In such situations, there is no substitute for accurate facts and viable beliefs.

5. They Lack the Ability to Make the Change

When people learn about a proposed change, one of the things they invariably ask themselves is, "Will I be able to do this?" Lack of ability may appear to be resistance, but inability and unwillingness are quite different. Whereas resistance represents an unwillingness or unreceptiveness to change, inability to change stems from lack of knowledge, skills, confidence, and/or necessary resources. *Knowledge* and *skills* have to do with one's actual ability, whereas *confidence* has to do with one's perceived ability. If people believe they can do something, they are more willing to try it; however, when they do not believe they can do it, fear of failure increases their resistance. It is possible for people to possess the ability to change, but still not believe that they can. In many cases, this belief becomes a self-fulfilling prophesy.

Resources can be divided into two categories: working conditions and communication. Working conditions are largely concerned with the availability and allocation of such physical resources as staff, money, time, equipment, and supplies; communication has to do with the interpersonal environment on the job and concerns such things as the effectiveness of supervision, feedback, cooperation, encouragement, support, and the information one receives. People can have the knowledge, skills, and confidence, but still be unable to make a change due to lack of necessary resources.

As Fig. 3 indicates, if both willingness and ability to change are taken into consideration, four scenarios can be distinguished.

- **Both willing and able to change.** In this scenario, the person both wants to change and can change. This is the combination most clearly associated with readiness to change.
- **Able but unwilling to change.** In this scenario, the person can change, but for some reason does not want to. This is the combination most clearly associated with resistance to change.
- **Willing but unable to change.** In this scenario, the person wants to change, but cannot. In other words, the person lacks the knowledge, skills, confidence, and/or resources necessary. Under this set of conditions, it is important to find out specifically why the person cannot change or does not believe he or she can change, so that appropriate corrective measures can be taken. Some managers have wanted to make the transition from a traditional hierarchical structure to a flatter, team-based structure, but they were so used to acting like a boss that they had difficulty sharing control. Some of them succeeded after considerable effort, while others failed in spite of everything they tried.
- **Both unwilling and unable to change.** In this scenario, the person does not want to change and cannot change. This combination poses an interesting challenge diagnostically. It is important to determine if the person is really unwilling *and* unable to change. Sometimes people say they cannot change when in actuality they just do not want to change. They would still resist the change, even if they gained the necessary knowledge, skills, confidence, and/or resources ("I would if I could, but I can't so I won't"). In other cases people are openly critical of a change, but the real issue,

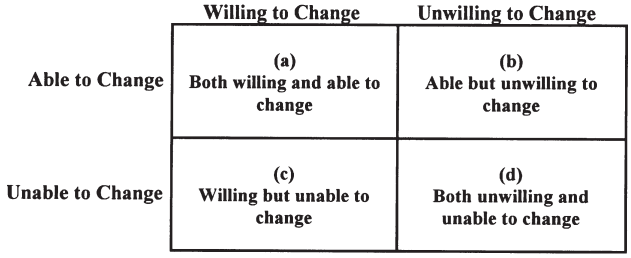


Figure 3 Willingness and ability to change.

which often remains hidden, is that they do not believe they can do it.

At other times, people might want to change if they were able to. Helping them gain the necessary knowledge, skills, confidence, and/or resources, therefore, could produce readiness to change. Organizations moving to self-directed teams also encounter people who could make the change, but do not want to, often because they do not believe this is an effective way to manage. These people either resist the change or take a job where they can continue operating as they have in the past.

6. They Believe the Change Will Fail

People can resist change because they do not have confidence it will work or they do not believe the resources are available to implement the change successfully. The anxiety stemming from these concerns will make it more difficult for people to support the change effort.

When people believe that the change will fail, it is important to make sure that they are not overestimating the risks. Also, as stated previously, sometimes when people are afraid that they might not be able to make a change, they cover this up by insisting that the change will not work. It is usually obvious that there is an underlying issue because they argue against the change no matter what is said. In cases like this, probing beneath the surface to get at the real issue is necessary to deal with their resistance.

7. The Change Is Inconsistent with their Values

Since values represent people's beliefs about what is important, they feel alienated if they believe the change conflicts with those values. Building greater compatibility between employee and organizational values is crucial to gaining commitment to change and to preventing or minimizing resistance.

8. They Believe Those Responsible for the Change Cannot Be Trusted

People will resist change if they believe that leadership either does not have their best interests at heart or is not being open and honest with them about the change and its impact. Lack of trust is an insidious and pervasive problem, robbing organizations out of much-needed commitment and performance. Just how big of an issue is mistrust? In a survey conducted by the Lausanne Institute, 91% of 474 government supervisors reported that lack of trust negatively impacts productivity. An *INC. Magazine* survey reported that 84% of employees do not trust their managers. Lack of trust puts people in a defensive posture. They spend their time protecting themselves from each other rather than focusing on organizational goals. If people have trouble trusting each other during routine times, this becomes an even greater issue during times of change.

IV. OVERCOMING RESISTANCE

The resistance strategy model, shown in Fig. 4, is intended to take a person step by step through a situation in which resistance is either anticipated or has already surfaced.

A. Step A: Define the Change

It is important to be as concrete, complete, and precise as possible in delineating what the change entails. When people state a desired change in vague or general terms, they often trigger resistance inadvertently. Another common mistake is to define the change only in terms of the *end* result. However, most changes involve a series of steps. In order to minimize resistance,

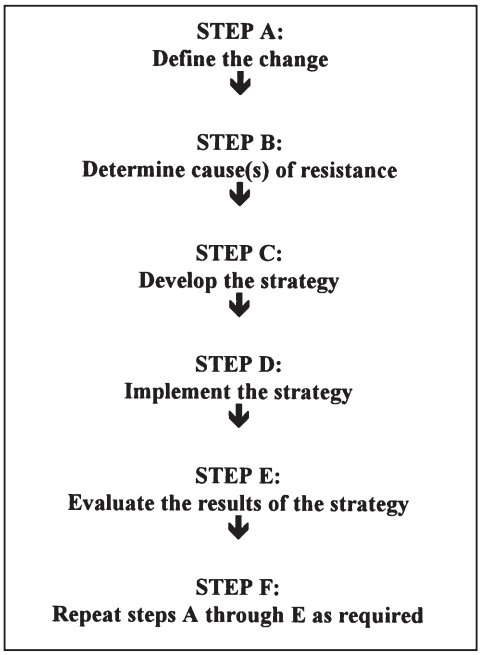


Figure 4 Resistance strategy model.

therefore, it is important to define the steps as completely as possible. Otherwise, people may wrongly conclude that their goal is being resisted when in fact the problem may only be with one step, for which there is a viable alternative.

B. Step B: Determine Cause(s) of Resistance

After defining the intended change, determine who will be affected by the change and anticipate how they will respond to it. Ideally, resistance should be anticipated in advance and steps should be taken to prevent it. If resistance cannot be prevented entirely, the next most desirable outcome is to minimize it.

Recall that resistance can stem from one or a combination of facts, beliefs, and values. Like putting together puzzle pieces, the more evidence gathered from what people do and say, the better it will be to separate causes from symptoms and to develop strategies dealing with causes. Since facts, beliefs, and values must be inferred from what people do and say, collect as much evidence as possible before drawing inferences. It is also important to regard inferences as hypotheses, open to modification if additional information comes to light.

C. Step C: Develop the Strategy

After identifying the cause(s) of resistance, the next step is to develop a strategy to deal with it. There is no magic formula for dealing with resistance. Even if the strategy is well thought out, something can always go wrong. Every situation is unique, and human behavior is too complex to predict with certainty. For any given instance of resistance, there will usually be a number of possible alternatives. Try to be as creative as possible in developing alternatives, so that an effective one is not prematurely ruled out.

Overcoming resistance is essentially a process of impacting people's facts, beliefs, feelings, values, and behavior. Some methods for dealing with facts and beliefs are to verify facts, clarify beliefs, challenge unviable beliefs, and suggest more viable beliefs. Examples of what to ask or say to do these things are given in Table II.

Since feelings are primarily consequences of thinking, they cannot be changed through direct intervention. It does not do any good to say to someone, "Do not feel that way." Instead, changes in feelings result from changes in facts and beliefs. Nevertheless, when feelings surface, it is very important to acknowledge and clarify them. Some ways of doing these things are listed in Table III.

When resistance relating to values is encountered, some methods for impacting them are to clarify values, challenge unviable values, and suggest more viable values. Examples are presented in Table IV. Finally, some approaches for impacting what people do are to clarify behavior, challenge ineffective behavior, and suggest more effective behavior. Table V gives some examples.

In developing potential strategies, keep in mind that facts, beliefs, feelings, values, and behavior are interrelated. Change in one variable often has ripple effects, changing others as well. Thus, value change often leads to change in beliefs, feelings, and behavior; belief change often leads to change in values, feelings, and behavior; and so on. Here are some examples:

1. When people succeed at something they did not believe they could do, this challenges them to reassess the accuracy of their beliefs. They are also likely to have a feeling of accomplishment and perhaps even to realign values to support the new behavior.
2. After people express their feelings about an upsetting change, they are often able to examine facts, beliefs, and values more objectively. This also can prevent them from making an impulsive decision.
3. When people experience a value conflict (e.g., between being honest vs being sensitive), feelings can inhibit their ability to act. Clarifying which value is more important can help them sort out their feelings and resolve the conflict.

At any given time a person will be more receptive to change in one variable than others. A common mistake is developing a strategy around a variable which is not receptive to change. It is a waste of time, for example, to try to change a person's value for security if the person staunchly defends that value. Instead, it might be more effective to indicate that after the reorganization everyone will be expected to engage in more risk-taking behavior.

To help a person become more adept at developing strategies, here are some suggestions for dealing with the eight common causes of resistance discussed earlier:

1. **They believe the change process is being handled improperly.**
   a. Explain how the change will affect them.
   b. Ask for and listen to their concerns.

Table II Interventions Aimed at Facts and Beliefs

Verifying facts	"What information are you basing that on?" "What facts do you have to back that up?" "Do you have all the facts?" "Are your facts accurate and complete?" "How do you know that is true?" "What additional data do you need?" "What evidence do you have?" "How did you arrive at that conclusion?" "How can you be sure?" "Do you need more information before making a decision?" "What are the costs?"
Clarifying beliefs	"What do you think about that?" "What is your opinion?" "What do you think is the primary cause?" "If you tried that, what do you think would happen?" "Do you think that will work? Why/why not?" "What problems do you see with that?" "What do you think he will do?" "What are the consequences?" "What are the risks?" "What opportunities do you see?" "What alternatives can you think of?" "How can this problem be solved." "What conclusion have you reached?"
Challenging unviable beliefs	"Do your beliefs allow you to meet your needs? "Could these facts be explained in any other way?" "What impact do those beliefs have on your feelingts/priorities/actions?" "That is not how I see the situation." "What are the chances of that happening?" "How else could his behavior be interpreted?" "Another way of viewing that would be. . . ." "The facts do not seem to support that conclusion."
Suggesting more viable beliefs	"A more accurate way of looking at that would be. . . ." "The facts tend to support the opposite conclusion." "You believe you have no other choice. Let us explore that to see if it is really true." "Another way of interpreting that would be. . . ." "This is how I see the situation."

c. Apologize for mistakes, or the issue will never go away.
d. Provide additional information (not excuses), as needed.
e. Ask for suggestions in order to avoid similar situations in the future.
f. Be honest about suggestions that can and cannot be accepted, and indicate why. People appreciate a straight answer.
g. Follow through on agreements reached to improve the situation. If people believe they have been given the brush off, it will make things much worse.

2. They believe there is not any need for the change.

a. Explain why the change is necessary.
b. Provide facts about the current challenges confronting the organization.
c. Explain how the change will help the organization survive and grow.
d. Listen and respond to their issues and concerns.
e. Clarify misconceptions about the change.
f. Indicate how the change will allow them to meet their needs more effectively.
g. Determine if they are setting their sights too low.
h. Discover if they are holding back due to fear of losing something.
i. Appeal to their sense of challenge.
j. Ask for their support in making the change work.

Table III Interventions Aimed at Feelings

Acknowledging feelings	"I can see you are upset, can we talk about it?" "You sound angry, tell me what happened." "You are feeling frustrated." "You are really worried about this." "I can feel a lot of tension in this room." "Even though his actions make you angry, you are afraid to say anything." "You are discouraged because you have not been able to make more progress."
Clarifying feelings	"How does that make you feel?" "What are you concerned about?" "It looks like something is bothering you. Do you want to tell me about it." "What impact do your feelings have on your thoughts/decisions/actions?" "I know what you think, but I am not sure how you feel."

3. **They believe the change will make it harder for them to meet their needs.**
   a. Discover if their facts are accurate and complete.
   b. Determine if their beliefs are based on accurate information.
   c. Provide additional information to correct mistaken or inaccurate beliefs.
   d. Offer more viable interpretations of the facts.
   e. Listen and express understanding.
   f. Ask them what positive results they think can come out of the change.
   g. Suggest ways to make the change easier for them.
   h. Ask how to help them implement the change.
   i. Ask for suggestions on how to make the change work better.
   j. Ask for alternatives to the change that might be more effective.
   k. Follow through on agreements reached to improve the situation.
4. **They believe the costs outweigh the benefits.**
   a. Ask them to discuss the costs.

Table IV Interventions Aimed at Values

Clarifying values	"What is most important to you?" "What outcomes do you want to see from this change?" "What is your bottom line?" "How would you prefer to deal with this?" "What is your preference?" "What factors do you think we ought to consider?" "What criteria should we use to evaluate alternatives?" "Which choice is most consistent with your criteria?" "What are your priorities?" "What are the benefits?" "If you had to make a choice, which one would it be?" "It looks like you disagree about what is important." "What would that accomplish?" "What purpose would that serve?"
Challenging unviable values	"I do not think you can have it both ways." "That seems to be less important in organizations now." "Do your values allow you to meet your needs?" "What impact do your values have on your thoughts/feelings/actions?" "The feedback indicates that a lot of managers do not walk the talk." "Your values seem to clash with those of other team members."
Suggesting more viable values	"One approach might be to place more emphasis on cooperation." "I think creating a greater sense of employee ownership would help morale." "Maybe it is time to place a higher priority on gender equity." "There would be some real benefits to becoming more customer focused."

Table V Interventions Aimed at Behavior

Clarifying behavior	"How would you go about doing that?" "What is the first step?" "How would you develop a plan of action?" "Those sound like good ideas, what else could you try?" "What would you say to him" "Why don't we role play that." "After that, what would you do?" "We have talked about how much this situation bothers you. Now let us talk about how you can change it." "How would you go about resolving this conflict?" "You know where you want to go; how can you get there?"
Challenging ineffective behaviors	"Do your actions allow you to meet your needs?" "What happened when you did that?" "I am not sure that will work." "How effective was that?" "I think that would make the situation worse." "I am not sure that would help you accomplish your goal." "In my experience that tends to put people on the defensive." "It seems to me that would be contrary to your value for. . . ."
Suggesting more effective behaviors	"Another way of dealing with that would be to. . . ." "It might be more effective if you. . . ." "Perhaps you could try. . . ." "An alternative way of responding would be to say. . . ." "One way to build more team cohesiveness would be to. . . ." "Perhaps it would help if you expressed more appreciation."

b. Determine if the costs are based on accurate information.
c. Provide additional information to correct inaccurate or mistaken beliefs.
d. Offer more viable interpretations of the facts.
e. Listen and respond to their issues and concerns.
f. Specify the material rewards (money, benefits, etc.) they will receive from the change.
g. Point out how the benefits of the change relate to their values.
h. Ask them if they can think of additional benefits.

5. **They lack the ability to make the change.**
   a. Ask them to express their concerns.
   b. Listen and respond with empathy.
   c. Convey a willingness to work with the person.
   d. Ask open questions to get at all relevant issues.
   e. Probe for unexpressed concerns and invite dialogue.
   f. Express confidence and reassurance.
   g. Offer encouragement and support.
   h. Make resources available (time, money, training, etc.).
   i. Offer coaching/mentoring.
   j. Follow up on agreements.
   k. Provide feedback on progress.
   l. Keep channels of communication open.
6. **They believe the change will fail.**
   a. Ask for and listen to their concerns.
   b. Discover if their facts are accurate and complete.
   c. Determine if their beliefs are based on accurate information.
   d. Provide additional information to correct mistaken or inaccurate beliefs.
   e. Offer more viable interpretations of the facts.
   f. Ask for suggestions to help make the change successful.
   g. Ask for alternatives to the change that might be more effective.
   h. Ask how to help them implement the change.
   i. Encourage them to visualize positive outcomes.
   j. Follow through on agreements reached to improve the situation.
   k. Ask for their support in making the change work.
   l. Express confidence in their ability to implement the change successfully.
7. **They believe the change is inconsistent with their values.**
   a. Ask them to describe the inconsistencies they see between the change and their values.

b. Explore the inconsistencies to determine if they are perceived or real.
c. If the inconsistencies are perceived but not real, provide additional information as needed.
d. When the inconsistencies are real, acknowledge their concerns.
e. Ask for suggestions on how the problem can be resolved.
f. When possible, modify the change.
g. If the change cannot be modified, be honest about it and ask for cooperation.
h. Build common ground.
i. Point out the benefits of the change (there are often more benefits than people realize).
j. Ask questions to determine if there are other issues involved, such as anxiety about the change or lack of confidence.
k. Offer help and support during the change process.

8. **They believe those responsible for the change cannot be trusted.**
   a. Share information openly and directly.
   b. Invite discussion of trust issues.
   c. Listen and convey understanding.
   d. Take responsibility for mistakes.
   e. Walk the talk.
   f. Follow through on agreements.
   g. Resolve disagreements directly and in good faith.
   h. Seek win–win outcomes.
   i. Be consistent and sincere.
   j. Honor confidential information.
   k. Give others credit when it is due.
   l. Involve others in decisions affecting them.
   m. Avoid actions contributing to mistrust (gossip, blaming, etc.).
   n. Act out of integrity and not expediency.
   o. Ask how a climate of trust can be fostered and maintained.

D. Step D: Implement the Strategy

In implementing a strategy, the two most important factors are *timing* and *pacing*. Timing has to do with implementing the strategy. An adequate strategy may intensify resistance simply because it was introduced at the wrong time. Ongoing communication and feedback will help determine the best time to implement a change.

Pacing, which is related to timing, concerns how much of the strategy to introduce at any given time. Even people receptive to change have limits to how much they can handle, since any change involves adjustment. If people are pushed, they can inadvertently create resistance which was not there. In their research, Kotter and Heskett (1992) found that adaptive organizations focused on making incremental changes. Avoid defeat by going too fast. If people show signs of anxiety, stress, or resentment, take a look at the pace of change. Since people will respond differently to the strategy, go slower with some of them. Also some people require a great deal of reassurance during times of change, while others do not seem to need any at all. Allowing for individual differences and modifying the pace accordingly will enhance the effectiveness of the strategy.

E. Step E: Evaluate the Results of the Strategy

Evaluating the effectiveness of the strategy is important in successfully dealing with resistance. Evaluation is not a one-time procedure which is completed after the strategy has been implemented. Rather, it is an ongoing process which begins from the moment of implementing the strategy and continues until the resistance is reduced or eliminated. Throughout the implementation phase, what people do and say will indicate whether or not the strategy is working. Sometimes attempts to deal with resistance will lead to dramatic results which are clearly positive or negative, but more often the results will show up in small changes. In their zeal to lower resistance, some people overlook or play down small changes or even abandon good strategies if they do not meet with immediate success.

This mistake can be avoided by keeping in mind that even small change in beliefs, values, or behavior can be a significant sign that the strategy is working. Since these elements are interrelated, a tiny change may signify the beginning of a process which eventually yields tremendous results. Therefore, notice *any* positive changes, regardless of magnitude, and nurture them to the fullest.

F. Step F: Repeat Steps A through E, as Required

Since any strategy is a hypothesis about how to lower resistance, it will be unclear if it is working until implementation begins. The results of the ongoing evaluation will indicate if modifications in the approach

need to be made. Rarely will a person be able to develop and implement a strategy without making adjustments. Be prepared to encounter obstacles and to deal with them as needed. Some common mistakes are the following:

1. Incorrectly assessing resistance
2. Basing the strategy on a symptom rather than on a cause of resistance
3. Failing to consider all relevant factors in developing the strategy
4. Implementing a strategy not well suited to the situation
5. Failing to adjust the strategy to account for new information
6. Implementing the strategy at the wrong time or at an unrealistic pace

In addition, the strategy may have unanticipated consequences. It could backfire, causing more problems. The cause of resistance may change in the middle of implementing the strategy, or people may respond with unanticipated counter-measures. When faced with setbacks, repeat Steps A through E as often as necessary to accomplish the objective. The biggest mistake is to stubbornly stick with a strategy, even when it is not working. A person will not be able to overcome resistance in others if he or she resists changing an ineffective strategy!

Two steps forward, one step back. Change and resistance—the ebb and flow of organizational life. Change is necessary for survival and growth, but change brings risks, and with risks, resistance. Managing change and resistance are essential skills in today's increasingly complex and competitive world. By implementing the concepts and techniques presented in this article, people can help their organization carve out a path to a successful future.

SEE ALSO THE FOLLOWING ARTICLES

Digital Divide, The • Ethical Issues • Future of Information Systems • Globalization • Human Side of Information, Managing the Systems • Organizations, Information Systems Impact on • People, Information Systems Impact on • Psychology • Sociology

BIBLIOGRAPHY

Beckhard, R., and Harris, R. T. (1987). *Organizational transitions: Managing complex change,* 2nd ed. Reading, MA: Addison-Wesley.

Brehm, J. W. (1966). *The theory of psychological reactance.* New York: Academic Press.

Head, T. C. (2000). Appreciative inquiry: Debunking the mythology behind resistance to change. *OD Practitioner,* Vol. 43, No. 1.

Hultman, K. E. (1979). *The path of least resistance: Preparing employees for change.* Austin, TX: Learning Concepts.

Hultman, K. E. (1998). *Making change irresistible: Overcoming resistance to change in your organization.* Palo Alto, CA: Davies-Black Publishers.

Kotter, J. P. (1990). *A force for change: How leadership differs from management.* New York: Free Press.

Kotter, J. P., and Heskett, J. L. (1992). *Corporate culture and performance.* New York: Free Press.

Lawler, E. E. (1973). *Motivation in work organizations.* Monterey, CA: Brooks/Cole.

Maurer, R. (1996). *Beyond the walls of resistance: Unconventional strategies that build support for change.* Austin, TX: Bard Press.

Moomaugh, R., & Associates (1999). Reengineering: Why it so often fails. Valley Center, CA: Organizational Universe Systems.

Moss-Kanter, E. (1985). *The change masters: Innovation and entrepreneurship in the American corporation.* New York: Simon & Schuster.

Schiemann, W. (April 1992). Why change fails. *Across the Board.*

Senge, P. M., *et al.* (1999). *The dance of change: The challenge of sustaining momentum in learning organizations.* New York: Doubleday.

Vroom, V. (1964). *Work and motivation.* New York: Wiley.

Robotics

Minoru Asada
Osaka University

ROBOTICS is an academic discipline of science and technology related to all kinds of robots. A robot is a programmable machine that imitates the actions or appearance of an intelligent creature, usually a human. Therefore, technological issues for building a robot similar to a human are considered as well as scientific issues related to human beings. Here, the basics of robotics and new trends in robotics are introduced.

I. WHAT IS ROBOTICS?

Robotics is an academic discipline of science and technology related to all kinds of robots. Then, what is a robot? Unfortunately, there is no exact definition on robots, but by general agreement a robot is a programmable machine that imitates the actions or appearance of an intelligent creature, usually a human. Figure 1 shows examples of these robots similar to animals and humans.

Compared to other artifacts with single functions, robots are expected to perform a variety of tasks, among which communication has an important role. A typical example is a humanoid robot, the building of which has been regarded as the final target of robotics for many years, especially in Japan where many robotics researchers have been struggling with making humanoid robots work in our society. Regardless of such efforts, no definition of humanoid robots exists since "robot" itself has no clear definition. From the viewpoint of biology, human beings can be discriminated from other species by three distinctive features or capabilities. They are biped walking, use instrumentation, and have the invention/use of language.

The first two capabilities have been attacked by robotics researchers as challenges in locomotion and manipulation that are the main issues in robotics. The third capability has not been considered as within robotics but linguistics. It seems far away from robotics. However, recent progress of research activities developed by the idea of "embodiment" in behavior-based robotics proposed by Rod Brooks at the MIT AI laboratory in the late 1980s has caused more conceptual issues such as body scheme, body image, self, consciousness, theory of mind, communication, and the emergence of language. Although these issues have been attacked in the existing disciplines such as brain science, neuroscience, cognitive science, and developmental psychology, robotics may project a new light on understanding these issues by constructing artifacts similar to us.

Thus, robotics covers a broad range of disciplines; therefore it seems very difficult to find comprehensive textbooks to understand the area. Some textbooks focus on the fundamental issues on how to build robots while others focus on a limited area in robotics such as kinematics, dynamics, control, vision, or planning. Among them, Russell and Norvig's book from 1995 entitled *Artificial Intelligence—A Modern Approach* gave a good introduction to robotics from the viewpoint of AI. Pfeifer and Scheier's book from 1999 entitled *Understanding Intelligence* shows a constructive approach to understanding intelligence with a variety of robots from the viewpoint of embodied cognitive science. Readers can access resources on the recent activities of robotics around the world through the Web.

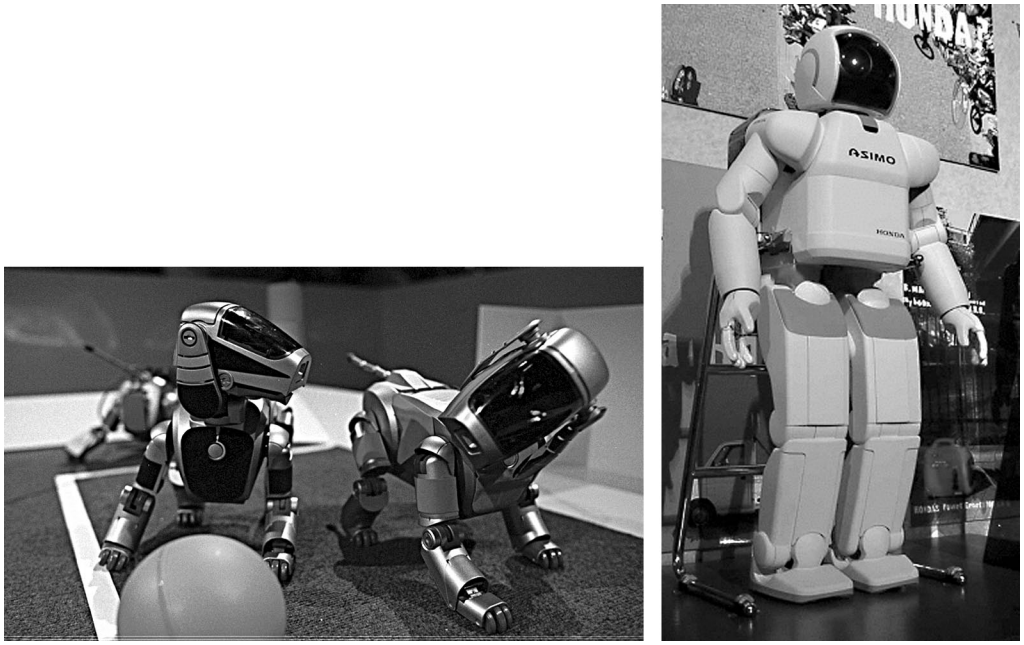


Figure 1 Animal robots and a humanoid robot. (Left) Two animal-like robots play soccer (© 1999, The RoboCup Federation). (Right) A humanoid robot, ASIMO (through the courtesy of Honda, Inc.).

II. BRIEF HISTORY OF ROBOTICS

The term robot comes from the Czech word robota, meaning drudgery or slavelike labor, and from the old church Slavonic robota, meaning servitude. It was first used to describe fabricated workers in a fictional 1920s play by Czech author Karel Capek called Rossum's Universal Robots. The left picture in Fig. 2 shows RUR.

Much earlier than RUR, the basic concept of robot, "automaton," was used as a term for automatic machines. Typical examples were wind-up dolls with a sophisticated mechanical system applied from clock technology in the early fourteenth century and later around the world. A picture of a Japanese tea-serving doll is shown in the center picture of Fig. 2, and its mechanism is illustrated in the design book by Hanzo Hosokawa from 1796 (see Fig. 2, far right). These dolls have basic components that the modern robots have in a different manner. Purely mechanical systems were used to represent the control program that is now coded by computer software.

Modern robotics has mainly two kinds of trends. One trend is the industrial robot working in material handling, assembling, welding, and painting for automation. Robot arms with multiple joints have been developed to operate in plants. The first product was done by Unimate industrial robots in 1961 (see Fig. 3, left). The control was done by step-by-step commands and the 4000-pound arm sequenced and stacked hot pieces of die-cast metal. Now, more than 700,000 industrial robots are working around the world, among them about 60% in Japan. The control procedures have been changing from remote control types to more autonomous ones: teleoperation, teaching-playback, numerical control, and sensor-based control.

While industrial robots focus on immediate applications, the second trend aims at future robots with more autonomy based on robotics and AI technologies. The first robot categorized in this class is "Shakey," built at Stanford Research Institute in California in 1968 (see Fig. 3, right). Shakey has various kinds of sensors on board and a communication line to the remote base. Although it was expected to perform the simple task of finding a box and carrying it to the next room in a simple environment, it failed due to something caused by unexpected events in the real environment. The researchers supposed that the main reason was immaturity of each component such as the sensor system, mechanism and motor control, and planning. Since then these areas have been developed separately as computer vision, robotics, and AI in a narrow sense.

Supported by the recent progress of microelectronics and computer technology in the last two decades, these components can be integrated into the

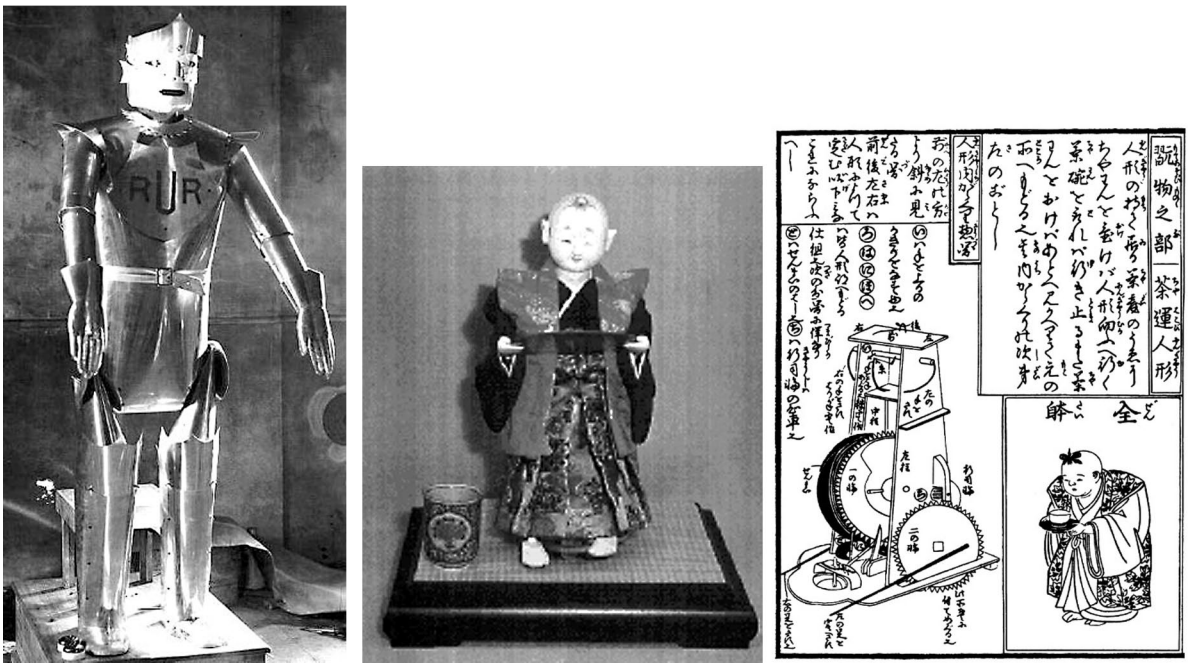


Figure 2 RUR (http://www.thetech.org/robotics/universal/breakout_p01_rur.html) and a Japanese wind-up doll (http://www/inforyoma.or.jp/karakuri/kara02.htm).

complete robot as shown in Fig. 1 that seems close to the ultimate robots we see in the science fiction movies such as Star Wars and more recently A.I., where fluent communication with humans is possible and their appearance is completely the same as humans. However, in order to realize such robots, we still need to solve many kinds of technical and scientific problems such as building an elastic, flexible skin and body and designing the structure for the emerging theory of mind for smooth communication.

III. MAJOR COMPONENTS OF ROBOTS

The three major parts of a complete robot are perception, cognition, and action. These are closely related to each other. Figure 4 shows these three components and the corresponding functions between sensors and actuators. In the following, the basic components of these parts are briefly described.

A. Perception

Robot sensors are classified into three categories: internal, interactive, and external (see Fig. 5). The internal sensors observe the internal state of the robot, typically the posture (position and orientation) and its changes (velocity and acceleration). Since the body structure of the current robot typically consists of many joints connecting links, the posture is defined as a joint vector $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots \theta_n)$ (n is the number of joints). Rotary encoders or potentiometers are often used to measure the angle of each joint. Figure 6 shows two types of rotary encoders, the absolute type and the incremental one. The former outputs the absolute position of the joint angle with binary code while the latter outputs the relative position or displacement by comparing the outputs of two pick-ups A and B.

Using the internal sensors, the robot can control its end effector very accurately and seems to perform everything. However, some sorts of tasks are difficult to do only with position sensing. Consider, for example, the task of scraping paint off a windowpane using a razor blade. To get all the paint requires positioning accuracy of about a micron in the direction perpendicular to the glass. An error of a millimeter would cause the robot to either miss the paint altogether or break the glass. In such a case, the robot needs to perceive touch (force and torque) to control its end effector appropriately. The force and torque sensors are placed between the manipulator and the end effector. Usually, a set of strain gauge is used to measure strain due to the force caused by the interaction between the end effector and the object. Therefore, these sensors are called interactive sensors. Figure 7

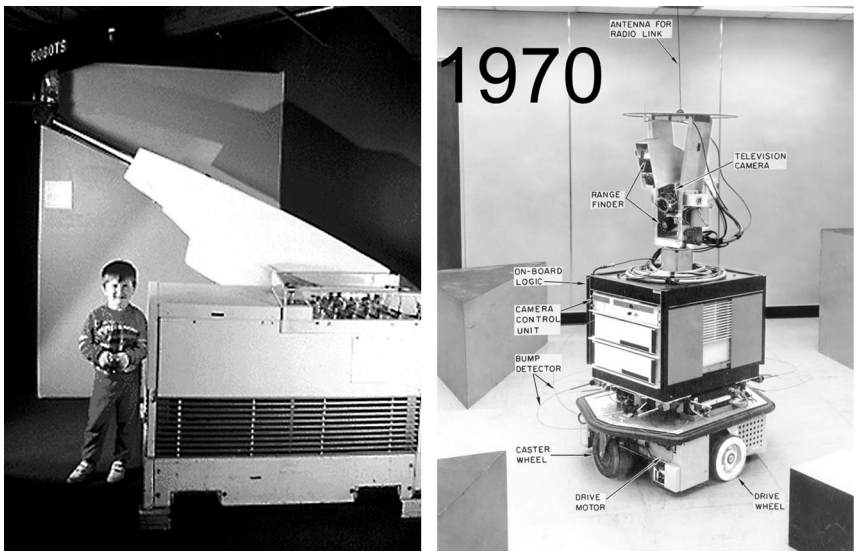


Figure 3 The first modern robots. Left, the Unimate industrial robot (http://www.ar2.com/ar2pages/uni1961.htm); right, the Shakey robot (http://www.frc.ri.cmu.edu/hpm/project.archive/robot/papers/2000/revo.slides/1970.html)).

shows the basic principle of the force and torque sensing with a strain gauge.

The external sensors are necessary for the robot to perceive the external environment to find the object, to monitor the execution, and to recover from the failures. They measure the physical properties of the environment. Vision and auditory sensors are typical, and actual sensors are TV cameras, sonar, and a laser range finder (hereafter, LRF). CCD cameras are most popularly used to capture the image of the environment and some sort of image processing is performed to find and track the object with the help of camera motion control. Typical image processing includes edge detection, color segmentation, finding correspondences between frames for stereo disparity, and optical flow detection. The top of Fig. 8 shows the basic principle of stereo depth calculation. The disparity d is the displacement of the target image between the left and right images, and the perpendicular distance Z from the observer to the target is simply calculated by triangulation, that is, $Z = fB/d$ where f and B denote the focal length of the camera and the distance between the focal points of the left and right cameras.

Stereo disparity (inversely proportional to the distance from the observer to the target) is obtained by finding

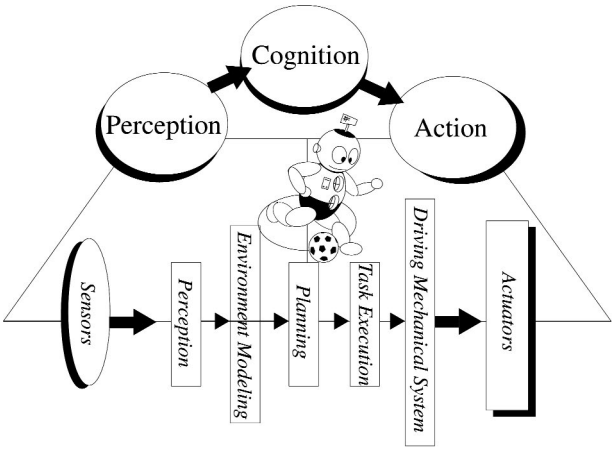


Figure 4 Major three components of a robot.

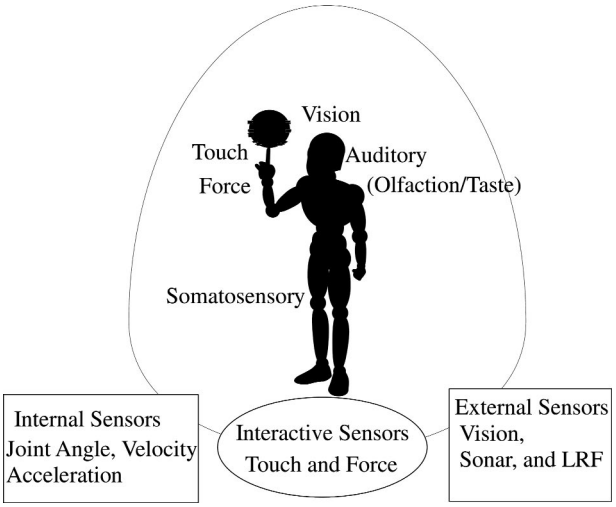


Figure 5 Three kinds of sensors.

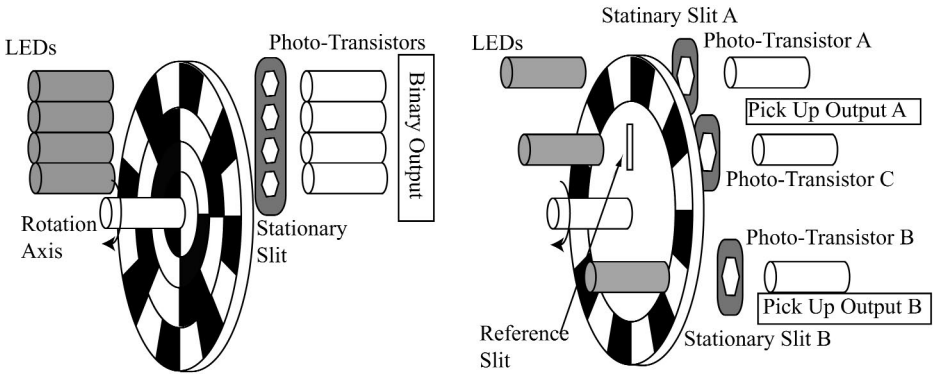


Figure 6 Two kinds of rotary encoders. Left, absolute type rotary encoder; right, incremental type rotary encoder.

the correspondences between the right and left images. The bottom of Fig. 8 shows a pair of stereo images and its disparity image in which the brighter (darker) regions are closer (farther) to (from) the robot. Finding the correspondence is performed by a simple correlation method of the intensity values inside small windows between the left and right image. The correspondence problem also occurs when the robot finds an optical flow that is an image flow pattern on the retina (image plane) caused by the relative motion between the eye and the observed objects in the environment.

Sonar and LRF are active sensing devices while vision is passive. Sonar (sound navigation and ranging) measures the distance or property of the target by transmitting an ultrasonic wave and measuring its echo. It is typically used for mobile robots to avoid obstacles. The LRF is a much more accurate depth sensor which transmits a short, invisible light pulse and measures the time interval elapsed from the moment of transmission of the pulse to the moment of reception of the returned "echo" from the target. Figure 9 shows scan data by LRF on the mobile robot.

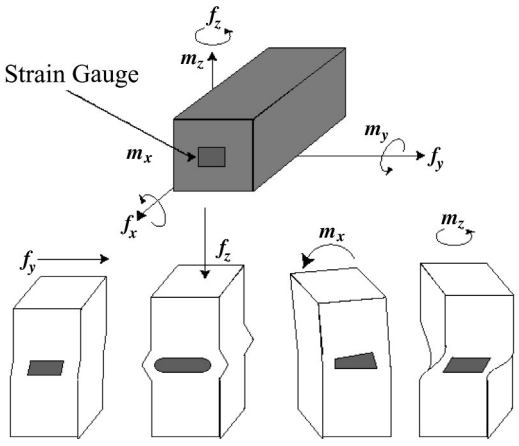


Figure 7 The principle of force and torque sensing.

B. Action

While computers and sensors deal with the information, the body structure with actuators further handles energy. Generally, an actuator is defined as a device that transforms the energy given from the source to kinetic energy. Electric motors are typical actuators. For simplicity, we will assume that each actuator determines a single motion or degree of freedom (hereafter, DOF) that corresponds to one joint motion (rotation or translation). Since it is expected to perform a variety of tasks, the robot needs to control its end effector through many links and joints. Given the link structure (the length of each link and the network topology with joints and links as nodes and arcs, respectively), we like to determine the posture of the end effector.

In a case of a serial link robot arm as shown in Fig. 10, it is simple and straightforward to calculate the final posture of the end effector when a joint vector $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots \theta_n)$ (n is the number of joints) is given. This process is called "forward kinematics." On the other hand, "inverse kinematics" determines the joint vector that realizes the given posture of the end effector. This is not as simple as in forward kinematics because of the possibility of no solution or a nonunique solution due to the redundancy. Since we, human beings, have seven DOFs from the shoulder to the hand, we can move the elbow keeping our hand fixed on the desk. This is a kind of redundancy. The redundant DOF is usually used for other tasks such as obstacle avoidance.

It is not sufficient to perform manipulation tasks in a real environment by only solving the inverse kinematics

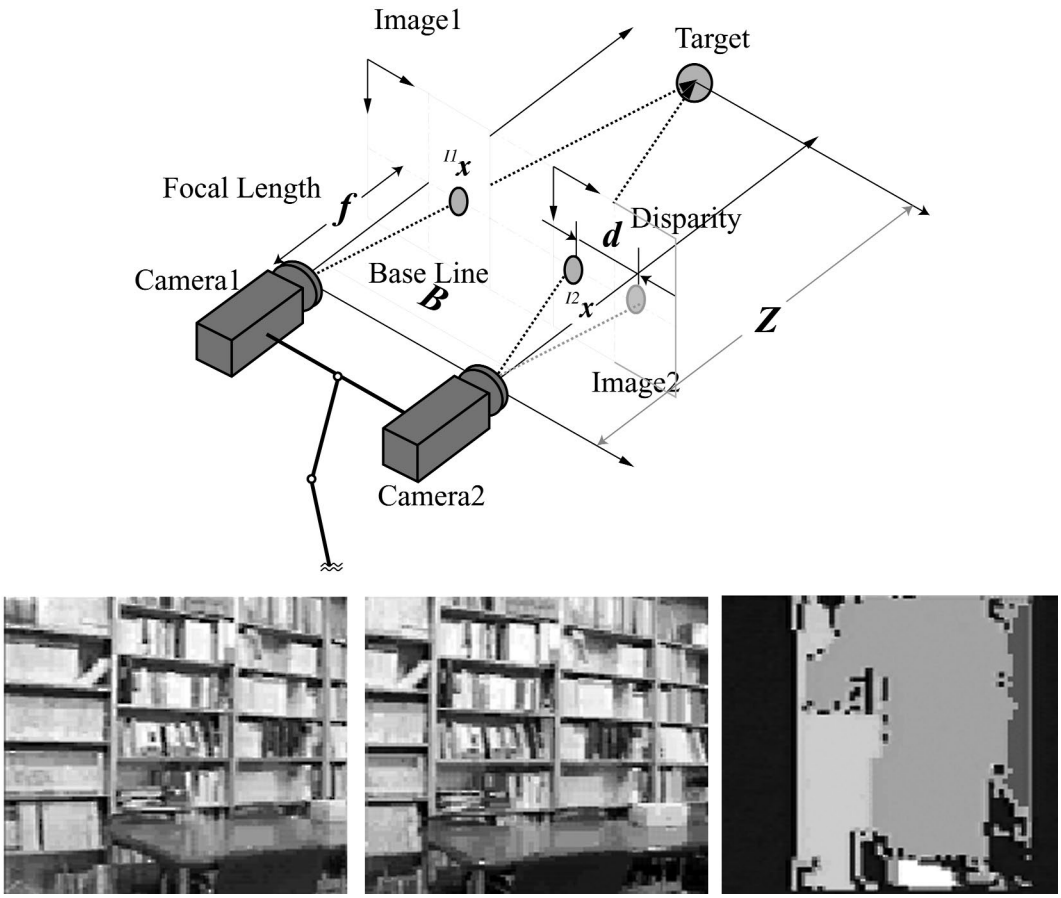


Figure 8 A stereo vision model (top) and a pair of stereo images and its disparity image (bottom).

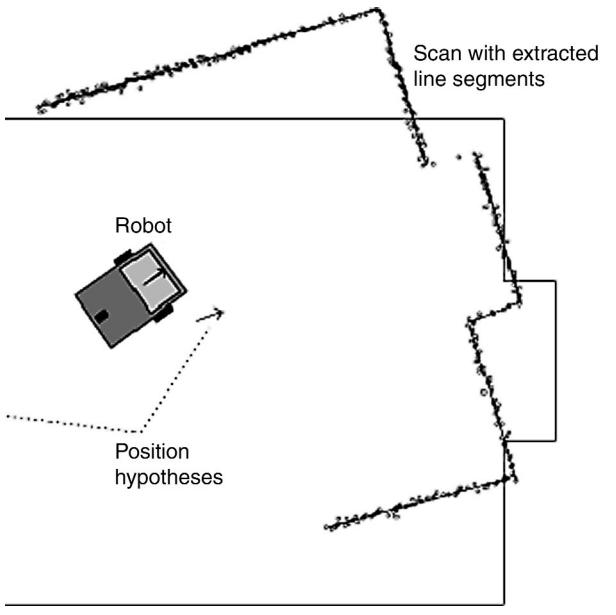


Figure 9 Scan data by LRF on the mobile robot (courtesy of C. S. Freiburg RoboCup team).

because actual parameters for actuators have not been determined. Similar to forward/inverse kinematics, "forward dynamics" obtains the rotational velocity and acceleration of each actuator given the input voltage to the actuator or the torque produced by the actuator, while "inverse dynamics" determines the input voltage to the actuator given the desired rotational velocity and acceleration of the actuator.

C. Cognition

From Fig. 4, cognition can be regarded as a process between perception and action. It seems to include environmental modeling, task planning, and task execution. Environmental modeling means a kind of reconstruction of the environmental structure. The external sensors contribute to the 3-D geometric reconstruction such as structure from motion in computer vision. Although three-dimensional reconstruction from a single two-dimensional image seems interesting, the

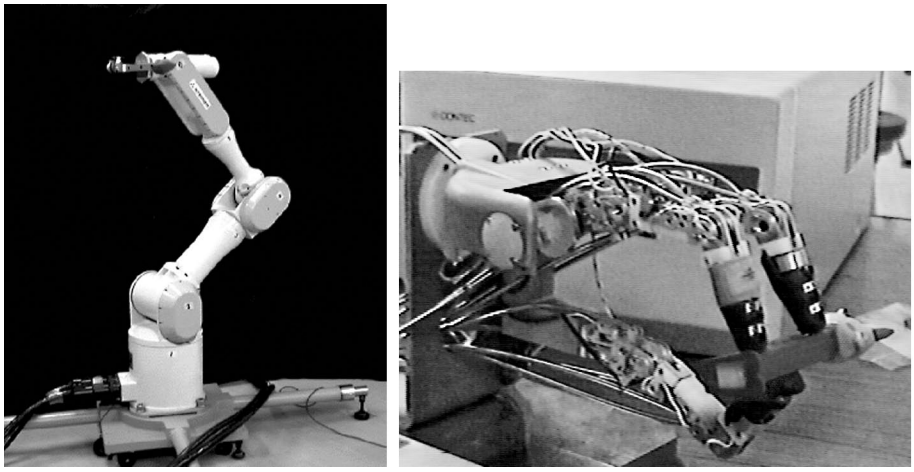


Figure 10 A robot arm and a three-finger hand.

problem is ill-posed; therefore, the numerical solution tends to be unstable and often nonunique. Instead of adding constraints to stabilize the solution that may take more computation time, direct methods to obtain the distance, such as stereo disparity, range from sonar, or LRF, are often used in robotics. Based on the obtained data, the 3-D lines or surfaces are reconstructed for the environment modeling and are used for further processing such as navigation or manipulation tasks.

Given the environmental model obtained by the above process and the action model of how the robot works in the environment, the internal representation that connects the environment model and the action has been considered to achieve the goal. A pioneer work is the state space representation by PLANNER by which the task can be efficiently represented in terms of the the initial position, the goal position, and the possible paths.

Figure 11 shows an example of the state space representation. The initial and goal positions of the object are indicated with possible paths. The object motion is of three kinds: vertical/horizontal translation and rotation (90°). The environment contains the obstacle regions. The state space (center) consists of the object position and orientation (the front and back planes indicate the horizontal and vertical orientation of the objects, respectively). Arcs connecting nodes display the possible paths while no connections between nodes indicate no paths due to obstacles. There are two solutions (right) if the rotation and translation have the same cost.

The most popular state representation for the task accomplishment is a configuration space representation from Lozano-Perez in 1987 who combined the task environment and the motion constraints of a robot in a very smart way. Figure 12 shows an example, where the task is to make a plan to move a target object (a small grey rectangle) from the starting position to the goal in an environment with obstacles (black objects). Instead of moving the target as a whole, the obstacles are expanded by the size of the target (grey regions). As a result, the motion planning is done by considering a representative point of the target (black dot), not caring for the whole body any more. This space is easily extended into more dimensional space when allowing the rotational motion of the target object. In this case, the

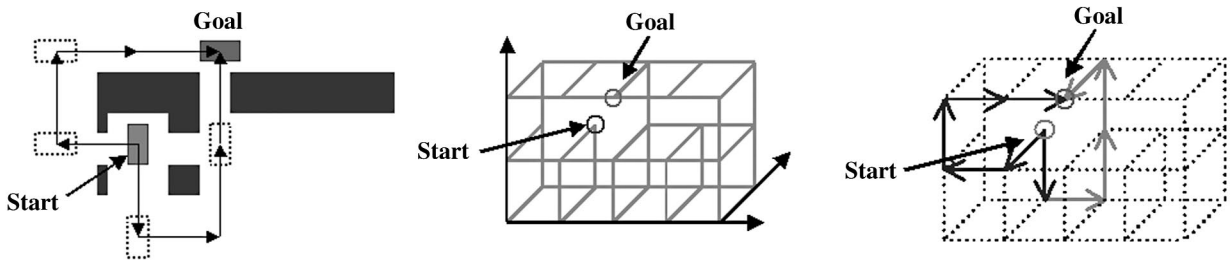


Figure 11 State space representation for motion planning.

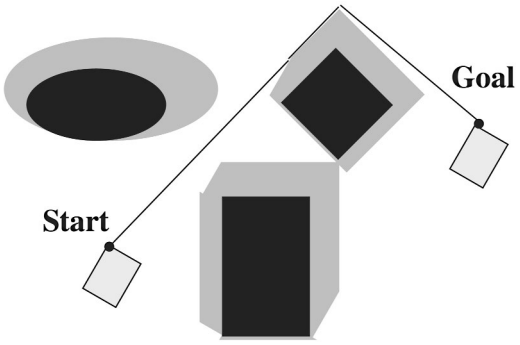


Figure 12 Cspace.

third dimension is added to represent the orientation of the target and the virtual obstacles are constructed by adding a similar plane but with a different orientation to the target object. Still, only the representative point motion is considered in this extended three-dimensional space. This idea is also applied to the motion planning of the robot arm which has many DOFs; therefore there is a high possibility of collisions with not simply obstacles in the environment but also the robot arm itself. In this case, each axis of the C-space represents the joint axis of the robot arm.

More symbolic planning systems are STRIPS, SHRDLU, BUILD, and so on. STRIPS consists of (1) the list representing the current situation of the environment and the robot, (2) robot action, (3) add list, and (4) delete list. STRIPS was actually used in the Shakey Robot Project (see Fig. 3). SHRDLU is a program for understanding natural language written by Terry Winograd at MIT from 1968–1970. SHRDLU carried on a simple dialogue with a user about a small world of objects (the BLOCKS world) as shown in Fig. 13. BUILD is a planning system for block manipulation with a modeling system in order to achieve the goal state via intermediate subgoals which the planning system produced with the knowledge of physics as a counterbalance.

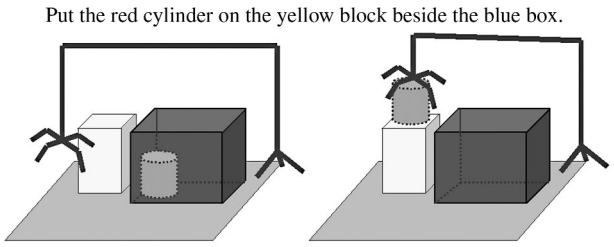


Figure 13 How SHRDLU works.

IV. CLASSICAL APPROACHES TO ROBOTICS

In the previous section, three major components of robotics are explained: perception, cognition, and action. A simple and straightforward way to build a robot control architecture seems to combine perception, cognition, and action in a sequential manner. Figure 4 shows this sequence where the robot observes the environment and reconstructs the environment model from the observed data. Then it makes a plan based on the reconstructed environment model, executes the task, and finally, drives the mechanism to act in the environment. A typical example is a navigation task of a mobile robot with range sensor where a range finder and a TV camera on board capture a range and intensity images, and the depth map as the environment model is reconstructed from the range data, on which the navigation planning is done and the motor commands to implement the planned motion are executed.

This sort of approach is based on the idea of "divide and conquer" that decomposes a complex task into a sequence and/or combination of subtasks, for each of which the human programmer designed a corresponding functional module. This seems comprehensive and logical; therefore the existing approaches have adopted this type of robot control architecture. In the case of the example above, a sequence of information processings is performed under the following assumptions.

- **Completeness of the information.** The information delivered from the previous module to the current one should be complete. If not, the current module may not be able to recover the incomplete information.
- **Accuracy of the information.** The information exchanged between different modules should be as accurate as possible considering the generality of the information processing results.
- **Unified representation.** The above two assumptions do not make any sense unless the unification of the information representation between different modules is guaranteed. Especially, in case of the environmental representation for the external world of the agent, the homogeneous coordinate system has been used as an objective representation common to the perception, cognition, and action modules.

These three assumptions are necessary for the idea of divide and conquer to work in the domain of robot tasks. They cannot be applied when the robot needs to

act in real time such as reflexive behaviors because it takes so much time to obtain the completeness and accuracy of the information although only partial information unknown in advance is usually used. Further, it may take more time to convert the unified representation to a meaningful one for the current module to use.

Therefore, this type of control architecture has been applied to simple and static tasks/environments. For complex tasks in dynamically changing worlds, different control architectures are introduced as explained in the following section.

V. NEW PARADIGMS OF ROBOTICS

A. Behavior-Based Approach

The classical control architecture described above has put its emphasis of the sensory information understanding on the generality of the information processing results. It seems difficult for such a control architecture to realize real time actions such as reflexive behaviors. In the late 1980s, Rodney Brooks at MIT AI laboratory advocated an approach to robot control architecture called behavior-based robotics. The idea is that the overall agent design can be decomposed, not into functional components such as perception, learning, and planning, but into behaviors such as obstacle avoidance, wall-following, and exploration. Each behavioral module accesses the sensor inputs independently to extract just the information it needs and sends its own signals to the actuators. Behaviors are arranged into a prioritized hierarchy so that higher level behaviors can subsume the outputs of the lower level behaviors. Figure 14 shows a hierarchy of behaviors proposed for a mobile robot by Brooks.

The main aim of behavior-based robotics is to eliminate the reliance on a centralized, complete representation of the world state, which seems to be the most expensive aspect of the conventional architecture. For tasks in which the appropriate action is largely determined by the sensor inputs, the slogan, "The world is its own model" is quite appropriate. However, for more complex tasks in which the internal representation of the world state may not have a simple or straightforward relationship to the sensor inputs plays the important role to determine the appropriate action. The classical approaches have designed the internal representation and specified the relationship between this representation and the actions to take. In order for the new paradigm to overtake the classical approach, the following problems should be attacked:

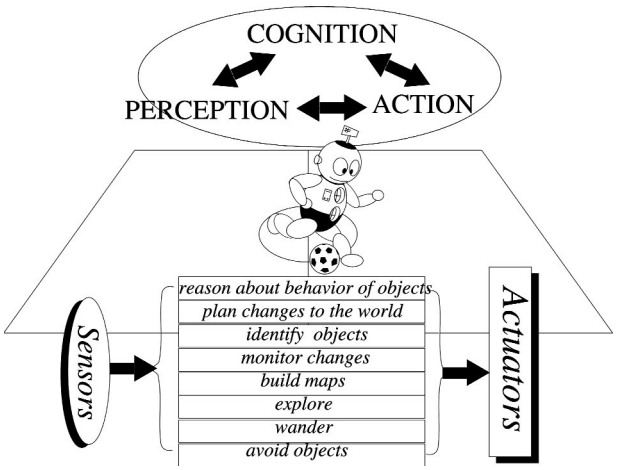


Figure 14 A hierarchical architecture for behavior-based robot control.

- How to specify this relationship, that is, how to learn the behavior
- How to design the internal representation of the world state, that is, how to construct the state space for the agent
- How to scale up the learned behaviors

These are also fundamental issues in robot learning that aim to realize autonomous robots through the interaction between the robot and its environment. In the following, attempts to solve these problems are introduced using examples applied to RoboCup that offers the test beds for multiple disciplines such as AI, robotics, computer vision, mechanics, multiagent, machine learning, and so on. Figure 15 shows a match in the middle sized league, where a four-on-four game is played between Japanese and German teams.

B. Reinforcement Learning

Reinforcement learning has recently been receiving increased attention as a method for robot learning with little or no *a priori* knowledge and higher capability of reactive and adaptive behaviors. In the reinforcement learning method, a robot and its environment are modeled by two synchronized finite state automatons interacting in discrete time cyclical processes. The robot senses the current state of the environment and selects an action. Based on the state and the action, the environment makes a transition to a new state and generates a reward that is passed back to the robot. Through these interactions, the robot learns a purposive behavior to achieve a given goal.

Figure 15 A match in the RoboCup middle sized league (courtesy of RoboCup Federation).

As a computational model for the reinforcement learning, Q-learning is the most popular one. We assume that the robot can discriminate the set **S** of distinct world states, and can take the set **A** of actions into the world. The world is modeled as a Markov process, making stochastic transitions based on its current state and the action taken by the robot. Let $T(s,a,s')$ be the probability of transition to the state s' from the current state-action pair (s,a). For each state-action pair (s,a), the *reward* $r(s,a)$ is defined (see Fig. 16).

The general reinforcement learning problem is typically stated as finding a policy f, a mapping from **S** to **A**, that maximizes the discounted sum of rewards received over time. This sum is called the *return* and is defined as

$$\sum_{n=0}^{\infty} \gamma^{n} r_{t+n}, \tag{1}$$

where r_t is the reward received at step t given that the agent started in state s and executed policy f. γ is the discounting factor; it controls to what degree rewards in the distant future affect the total value of a policy. The value of γ is usually slightly less than 1.

Given definitions of the transition probabilities and the reward distribution, we can solve for the optimal policy, using methods from dynamic programming. A more interesting case occurs when we wish to simultaneously learn the dynamics of the world and construct the policy. Watkin's Q-learning algorithm gives us an elegant method for doing this.

Let $Q^*(s,a)$ be the expected return or *action-value function* for taking action a in a situation s and continuing thereafter with the optimal policy. It can be recursively defined as

$$Q^*(s,a) = r(s,a) + \gamma \sum_{s' \in S} T(s,a,s') \max_{a' \in A} Q^*(s',a'). \tag{2}$$

Because we do not know T and r initially, we construct incremental estimates of the Q-values online. Starting with $Q(s,a)$ equal to an arbitrary value (usually 0), every time an action is taken, the Q-value is updated as

$$Q(s,a) \leftarrow (1-\alpha)\, Q(s,a) + \alpha (r(s,a) + \gamma \max_{a' \in A} Q(s',a')), \tag{3}$$

where r is the actual reward value received for taking action a in a situation s, s' is the next state, and α is a learning rate (between 0 and 1).

Figure 17 shows how the action value function $Q(s,a)$ is constructed through the learning process. The top figure indicates the initial situation of the function. Action values for all pairs of (s,a) are ini-

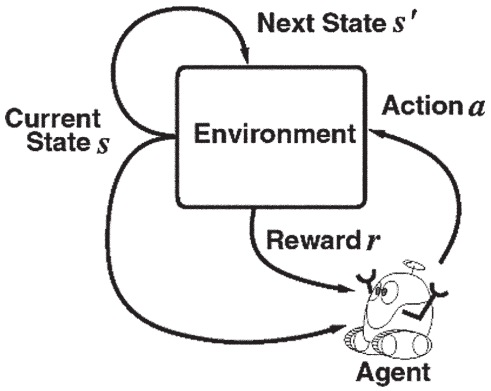


Figure 16 A fundamental model of reinforcement learning.

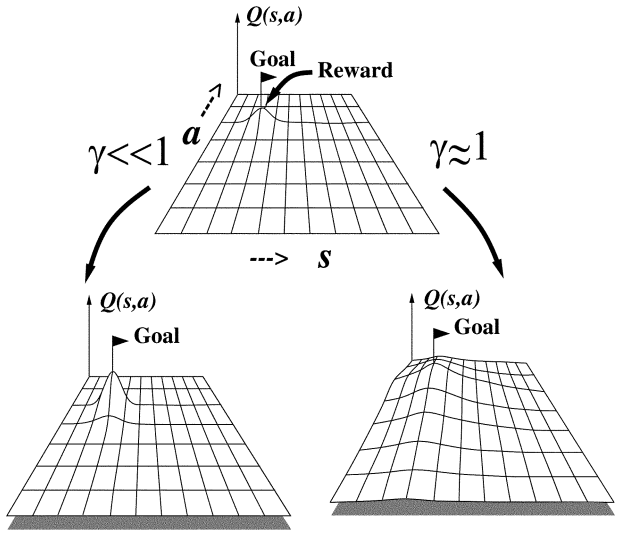


Figure 17 Action value functions in terms of state and action.

tialized to zero except for the goal state where the reward for the task is given. The robot tries to find this goal state and to update the action values to indicate which state is closer to the goal (a higher action value means closer to the goal).

In the above updating formulation for $Q(s,a)$ (3), there are two interesting parameters: α and γ. The learning rate α is set as a large value in the early stage of learning to obtain the world knowledge as much as possible, but gets smaller at the later stage to converge the learning process. This reminds us of the learning process of human beings. When young, we assimilate knowledge from our experiences, but gradually get conservative and finally hardheaded. The final situation corresponds to $\alpha = 0$, no change happening. So, should α be always one? If $\alpha = 1$, the action value is always changed, that is, no memory. This means that the past good experiences might be replaced with the current bad experience. Therefore, an appropriate value between 0 and 1 should be chosen. Theoretically, we need a strict reduction process for the learning rate α, but in actual applications a fixed value is often used.

The second one is the discounting factor γ, which controls to what degree rewards in the distant future affect the total value of a policy. If we substitute a small value for it, the action values spread out from the goal state immediately decays as shown in Fig. 17, bottom right. Therefore, after learning, the agent easily loses its way to the goal because no action values are found at the distant state from the goal. In the case of a large value for γ, the action values from the goal state gradually decay, and therefore the agent can easily find the way to the goal. The latter may contribute to purposive behavior acquisition from the starting position to the goal state while the former to reflexive behavior such as obstacle avoidance.

The difference in the character of the robot soccer player (the details will be explained later) due to the discounting factor γ is shown in Figs. 18a and 18b in which the robot started from the same position. In the former, the robot takes many steps in order to ensure the success of shooting because of a small discount (large value for γ), while in the latter the robot tries to shoot a ball immediately regardless of certainty of success of its shooting because of a large discount (small value for γ). Thus, learning parameters may affect the character of robot players.

C. Behavior Acquisition by Robot Learning

Although the role of reinforcement learning is very important to realize autonomous robots, the prominence of that role is largely dependent on the extent to which the learning can be scaled to solve larger and more complex real robot learning tasks. Many works in the reinforcement learning field have only shown computer simulations in which they assume ideal sensors and actuators, where they can easily construct the state and action spaces consistent with each other. This is the issue of the state space construction problem mentioned in Section V.A. Related to this, acceleration of learning is one of the most serious problems since learning time can be of exponential order in the size of the state space. These related issues prevent reinforcement learning methods from being applied to real robot tasks.

In the following, we show an example task of a robot soccer player that tries to shoot a ball into a goal

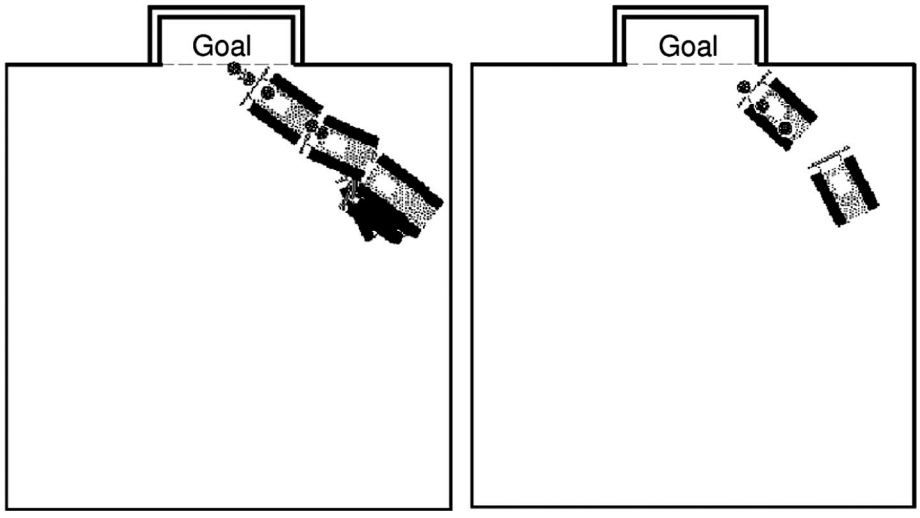


Figure 18 Two kinds of behaviors with different γ's. (Left) shooting ($\gamma = 0.999$); (Right) shooting ($\gamma = 0.6$).

based on visual information and point out how the above issues can be attacked.

The problem here is how to develop a method which automatically acquires strategies for shooting. We assume that the environment consists of a ball and a goal, the mobile robot has a single TV camera, and the robot does not know the location/size of the goal, the size/weight of the ball, any camera parameters such as the focal length and tilt angle, or the kinematics/dynamics of itself. Figure 19a shows a picture of the real robot with a TV camera (Sony handy-cam TR-3) used in the experiments.

First, we construct the state and action spaces for this robot. The image, supposed to capture the ball and/or the goal, is the only source of information the robot can obtain about the environment. The ball image is classified into 9 substates, combinations of three classifications of positions (left, center, or right) and three types of sizes (large (near), middle, or small (far)). The goal image has 27 substates, combinations of three properties each of which is classified into three categories (see Fig. 19b). Each substate corresponds to one posture of the robot towards the goal, that is, position and orientation of the robot in the field.

The robot can select an action to be taken in the current state of the environment. The robot moves around using a PWS (power wheeled steering) system with two independent motors. Since we can send the motor control command to each of the two motors separately, we construct the action set in terms of two motor commands ω_l and ω_r, each of which has 3 subactions, forward, stop, and back. All together, we have 9 actions in the action set **A**.

The state and action spaces are not discrete but continuous in the real world; therefore it is difficult to construct the state and action spaces in which one action always corresponds to one state transition. Actually, the above state and action spaces do not hold to this. Then, we reconstruct the action space as follows. Each action defined above is regarded as an action primitive. The robot continues to take one action primitive at a time until the current state changes. This sequence of action primitives is called an action. This is the case in which the action space is defined by the state space. Since the state space designed by a human programmer is not guaranteed as optimal for the robot, another way to construct the state space is based on the reward (success or failure). For example, we regard a set of perceived situations as a state if continuous execution of one kind of motion primitive defined above leads the goal state (or already obtained state).

Currently, a number of methods to deal with the continuous state and action spaces have been invented and applied to real robot tasks. However, what kinds of state and action spaces should be prepared is still one of the essential problems in robot learning. Another issue is the acceleration of learning since it can be exponential in the size of the state space. If we simply apply Q-learning with the above state and action spaces to the task, we could not make it. First, we placed the ball and the robot at arbitrary positions. In almost all the cases, the robot crossed over the field line without shooting the ball into the goal. This means that the learning did not converge after many trials.

This situation resembles a case where a small child tries to shoot a ball into a goal, but he or she cannot imagine in which direction or how far the goal is because a reward is received only after the ball is kicked into the goal. Further, he or she does not know which action is to be selected. This is the famous *delayed reinforcement* problem due to no explicit teacher signal that indicates the correct output at each time step.

To avoid this difficulty, we construct the learning schedule such that the robot can learn in easy situations at the early stages and later on learn in more difficult situations. We call this *learning from easy missions* (or LEM). If we can accurately order the states from

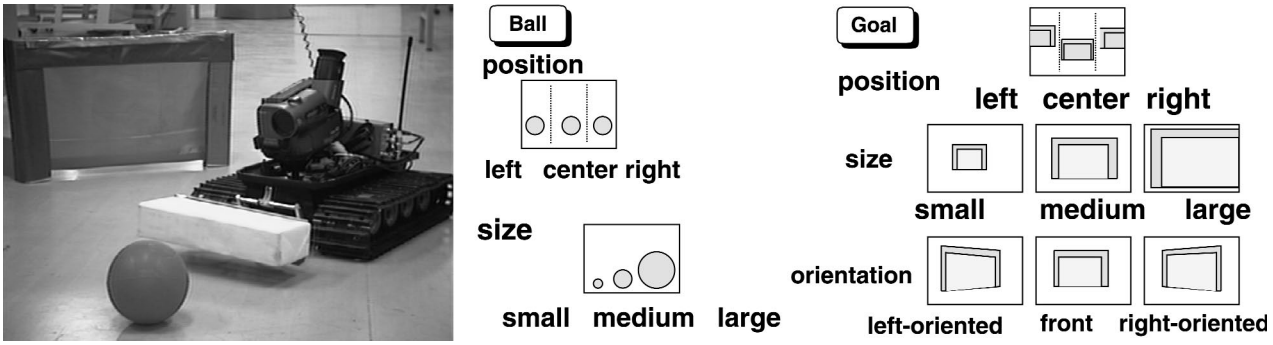


Figure 19 A real robot and its state space. (Left) A robot soccer player. (Right) A state space designed by human.

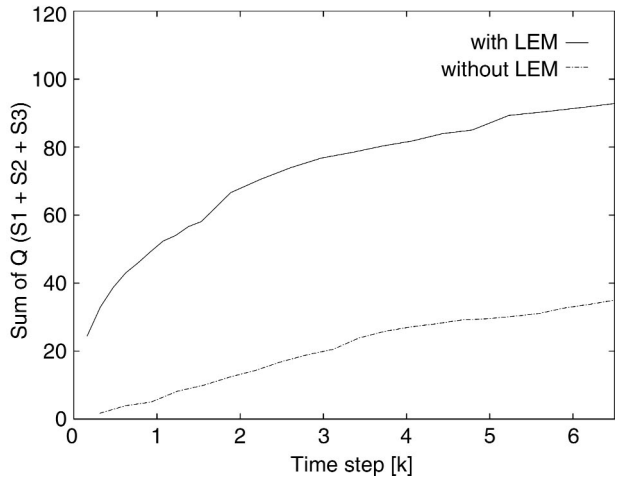


Figure 20 Change of the sum of *Q*-values with and without LEM in terms of goal size.

easier ones to more difficult ones, the learning time can be theoretically from exponential order to linear in the size of the state space.

Figure 20 shows how LEM works, where the states are ordered in terms of the goal size, that is, "goal is large (S_1)" (close to the goal), "goal is medium, (S_2)," and "goal is small (S_3)" (far from the goal), and the summation of *Q*-values of these states. As shown in this figure, LEM accelerated the learning and succeeded in obtaining the desired behavior. The robot could not do that without LEM.

Figure 21 shows the real robot experiments, where a sequence of robot actions and images captured by the robot are shown in a raster scan order from top left to bottom right. The robot approached the ball and tried to shoot a ball into the goal, but failed, then went back until capturing the ball and the goal image, and finally shot the ball into the goal.

LEM is a kind of learning scheduling to accelerate the behavior acquisition, which seems essential to scale up the learning method to more complex and dynamic tasks/environments. In a case of multiagent systems, simultaneous learning to obtain the desired behaviors such as cooperation between two robots seems impossible because the policies of two robots are changing during the learning. Consider the performances of two tennis beginners on the court: neither can improve its skill. In such a case, one of them should take a role of coach which has a fixed policy that enables the learner to improve its skill. This is actually applied to the case of two-robot cooperation or competition such as a combination of pass and shoot and a penalty kick. In the former, one robot (passer or shooter) fixed its policy during the learning of the opponent, and changed their roles after the skill improvement of the learner. In the latter case, the behavior of the goal keeper is controlled so that the shooter can have confidence to shoot a ball into the goal.

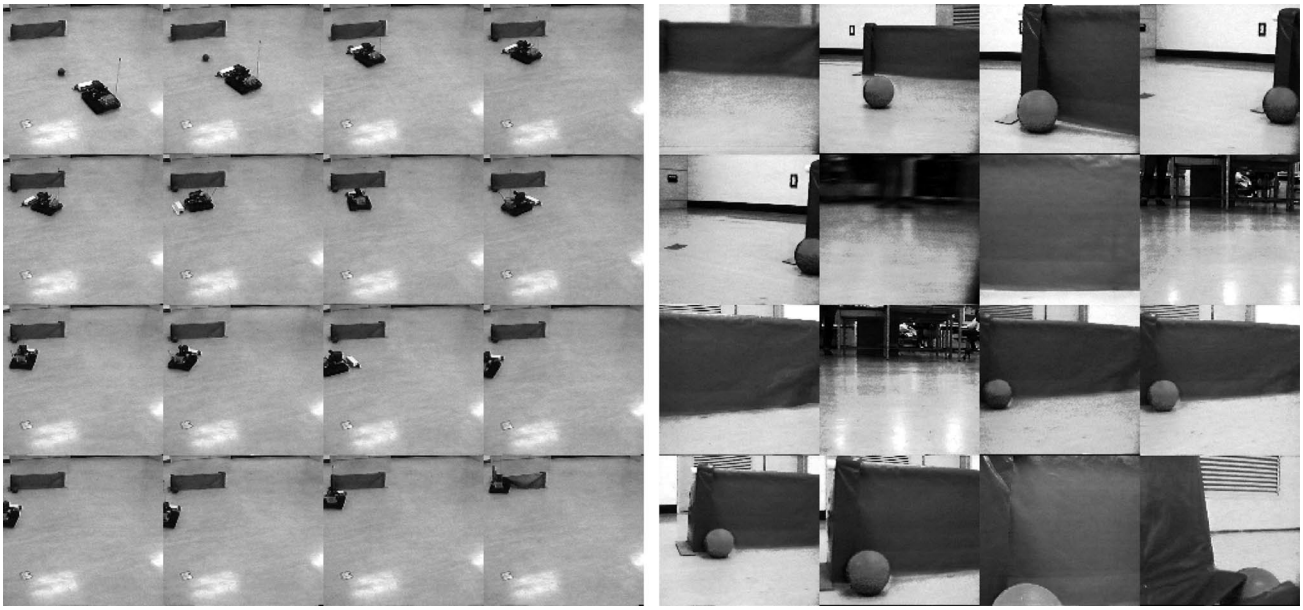


Figure 21 A real robot experiment. (Left) The robot succeeded in finding and shooting a ball into the goal. (Right) Images taken by the robot during the task execution.

Figures 22a and 22b show the performance of cooperative behavior, a combination of pass and shoot, and a sequence of images captured by the passer (the right images of (b)) and the shooter (the left images of (b)). The task of the passer is to pass the ball towards the shooter and that of the shooter is to kick the passed ball into the goal. In this case, the state space construction is much more complicated than the case of a single robot because the state space should include the situation caused by the opponent behavior that may be affected by the learner's behavior, too. Asada *et al.* invented a sophisticated method to construct such a state space by offline learning of the state space construction taking the effect of the learner's action on the opponent's behavior into account.

These behaviors obtained by the robot learning methods can be reused in a couple of ways. One is the hierarchical organization of learning modules: the lower modules deal with physical sensors and actuators while the higher modules learn to determine one of the lower modules is to be activated according to the current situation. The other is an evolutionary approach such as a genetic algorithm or genetic programming, in which the already learned behaviors are used as basic behaviors, and the evolutionary computations are used to select or switch these basic behaviors.

As we can see above, a simple application of learning theory seems limited to the real robot tasks such as cooperation and competition of a multiagent system. A typical example can be seen in RoboCup competitions. In such a case, we need learning scheduling such as LEM and alternate learning, which can contribute not only to simple acceleration of the learning but also to realization of the complex tasks. These can be regarded as environmental issues that help the robot learning.

Another category of environment issues is teaching by showing (or learning by observation from a viewpoint of the learning robot), especially in the case of the robot with many degrees of freedom such as humanoid robots, for which conventional learning methods are difficult to apply due to a huge search space caused by various sensors and many DOFs. This imitation learning seems useful both in designing the behaviors of humanoid robots and in understanding human behaviors.

VI. FUTURE OF ROBOTICS AND HUMAN SOCIETY

Sony AIBO has already been sold to general consumers and Honda ASIMO is also close to being in-

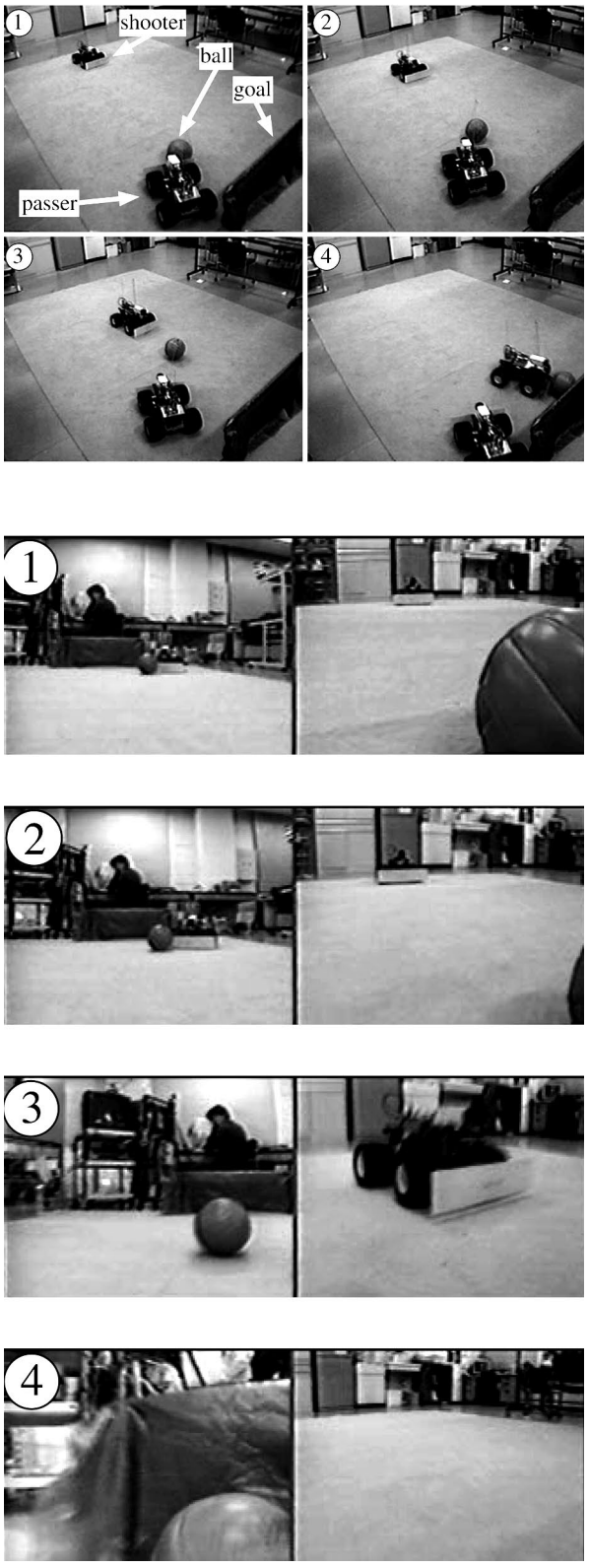


Figure 22 A combination of pass and shoot. Top view, and on board images.

troduced in our daily life. These two types of robots are symbolic in twofold: AIBO was designed as a robot pet with no actual use to help or assist people while ASIMO was designed to do that; therefore HONDA researchers are particular about its size in our daily life at home. However, still many issues about safety are left unsolved. These two types of robots are "visible," that is, much like the robots we can imagine in science fiction books or see in movies such as Star Wars. Their appearances are similar to animals or human beings, but their practical uses seem limited.

On the other hand, there are many "invisible" robots, which might not be suitable to be called "robots," but obviously robot technology is used. On the small size, micro and nano machines will be robots expected to be inside our body for medical application. On the medium size, we have already had various rehabilitation robots for walking, mastication, and so on. So far, the subjects were forced to be patient with their pains during the rehabilitation. By introducing a kind of robot technology, for example, a compliant motion control enabled the rehabilitation without any pains. Professor Atsuo Takanishi's group at Waseda University, Japan, has developed a series of dental robots (http://www.takanishi.mech.waseda.ac.jp/jaws/index.htm). Further, artificial limbs can be controlled by electromyogram of the subject. We do not call such limbs robots, but obviously robot technology has been involved. On the large size, intelligent traffic systems and space ships would be robots inside which human beings reside. These invisible robots have exact missions and have been already or will be useful in our life.

Some robots are both visible and invisible. A typical example is a class of media robots of which the appearance is user friendly but artificial. They are designed as one of our home electronics that help us to communicate with other home electronic machines such as the VCR for reservation of programs, or mail reading and sending through voice generation and recognition. Some of them are connected to hospital or security centers to monitor unexpected happenings, especially of seniors.

Another application area is cooperative robots with human beings such as rescue robots and space missions (telerobotics), where cooperation between robots and operators is indispensable. In the case of rescue robots, a remotely controlled vehicle enters into the dangerous area which human beings cannot access and executes the mission such as searching for victims or operations to avoid or minimize a disaster. A typical case of the latter is the NASA Mars Pathfinder mission, in which a teleoperated robot moves around to search, regardless of the time delay due to distant communication. Figure 23 shows these robots.

Thus, various kinds of robots or artifacts supported by robot technology will be gradually introduced into our daily life more and more. In these robots, two key technologies are the main issues: one is development of new material that enables physical contact with humans, that is, artificial skin and muscles. RoboCup offers a good test bed to check if the human players may not be so damaged by a collision with robot players.

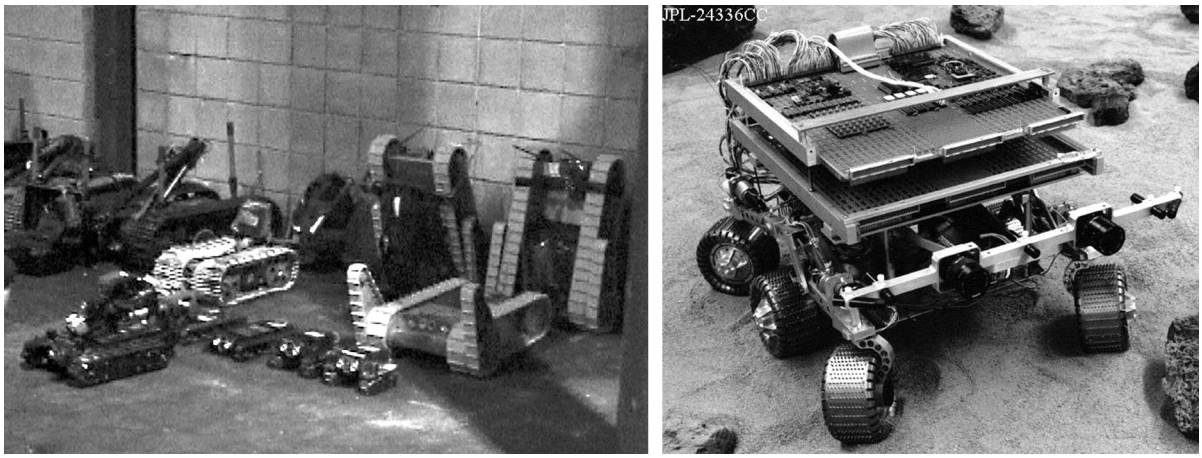


Figure 23 Cooperative robots with human operators. (Left) Rescue robots working at WTC response (courtesy of the National Institute for Urban Search and Rescue's Center for Robot-Assisted Search and Rescue). (Right) A planetary exploration robot Rover Sojourner (courtesy of NASA/JPL/CALTECH.

Of course, a desirable situation is that both sides are safe, but at least the human side should not be so damaged as the robot side. The material can be applied to many situations if it passed this check.

Another is communication skill with humans. Recently, voice recognition and generation techniques have developed rapidly, and their applications seem to be expanding into various areas. However, it still requires limited speakers in a noiseless environment with limited vocabularies. This is not simply a problem of voice recognition and generation, but a problem of language acquisition by a robot. From the viewpoint of development, language acquisition by a robot is a very interesting topic. This implies that if a robot succeeded in acquiring a language, it would be useful not only for artifacts but also for understanding how children can acquire language skills. The latter is one potential for the scientific contribution of robotics.

SEE ALSO THE FOLLOWING ARTICLES

Artifical Intelligence Programming • Engineering, Artificial Intelligence in • Expert Systems Construction • Hybrid Systems • Industry, Artificial Intelligence in • Machine Learning • Neural Networks • Pattern Recognition • Speech Recognition

BIBLIOGRAPHY

Asada, M. (1990). Map building for a mobile robot from sensory data. *IEEE Trans. on System, Man, and Cybernetics,* Vol. 20, 1326–1336.

Asada, M., Noda, S., Tawaratumida, S., and Hosoda, K. (1996). Purposive behavior acquisition for a real robot by vision-based reinforcement learning. *Machine Learning,* Vol. 23, 279–303.

Asada, M., Uchibe, E., and Hosoda, K. (1999). Cooperative behavior acquisition for mobile robots in dynamically changing real worlds via vision-based reinforcement learning and development. *Artificial Intelligence,* Vol. 110, 275–292.

Bellman, R. (1957). *Dynamic programming.* Princeton, NJ: Princeton Univ. Press.

Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE J. Robotics and Automation,* Vol. 2, 14–23.

Connel, J. H., and Mahadevan, S. (Eds.). (1993). *Robot learning.* Dordrecht: Kluwer Academic.

Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., Osawa, E., and Matsubara, H. (1997). Robocup: A challenge problem of ai. *AI Magazine,* Vol. 18, 73–85.

Lozano-Perez, T. (1987). A simple motion planning algorithm for general robot manipulators. *IEEE J. Robotics and Automation,* Vol. 3, 224–238.

Pfeifer, R., and Scheier, C. (1999). *Understanding intelligence.* Cambridge, MA: MIT Press.

Russell S. J., and Norvig, P. (1995). Robotics, *Artificial intelligence—A modern approach* (S. J. Russell and P. Norvig, Eds.), Chap. 25. Englewood Cliffs, NJ: Prentice Hall.

Watkins, C. J. C. H. (1989). *Learning from delayed rewards.* Ph.D. thesis, King's College, University of Cambridge.

RPG (Report Program Generator Language)

Douglas C. Pence
CPU Medical Management Systems

GLOSSARY

report program generator language (RPG) Developed in the 1960s by IBM, RPG is one of the only computer languages ever developed specifically for business. RPG has been continually enhanced as the needs and requirements of the business community have evolved.

THE ACRONYM RPG stands for Report Program Generator and that is precisely what the language was originally designed for. The International Business Machines (IBM) Corporation invented the RPG computing language over 40 years ago.

Even though the RPG language has evolved considerably, it remains one of the few languages designed strictly for business. Applications written in RPG are running on over 1 million systems worldwide. The language runs exclusively on IBM "midrange" systems, and the most current iteration of the language may be found on IBM's iSeries 400 computers. That being said, it is entirely possible that many people have never heard of RPG.

I. ORIGINS OF THE REPORT PROGRAM GENERATOR LANGUAGE

IBM developed the Report Program Generator (RPG) computing language in the early 1960s. To this day, RPG remains one of the only computer languages designed specifically for business use.

On April 7, 1964, IBM announced its System/360 business computer, which was to become a wildly successful endeavor. Costing $5 billion to develop, the System/360 was by far the most successful business computer of its day. And as the primary business language for the System/360, the RPG language had begun to come into its own.

The design of the RPG language was largely columnar in nature and was centered on the design of the 80-column punch card, which was the principal method of communication with computers of that time. By utilizing a handful of specification formats in conjunction with the 80-column format, RPG was able to produce output with a minimal amount of code. The principal concept was simple; do more with less. At the time, this was an important aspect of the language, since programs and the data being processed were usually stored on 80-column cards.

As computing began to evolve in the late 1960s and early 1970s, so did the RPG language. It is important to note that the evolution of the RPG computing language was directly tied to the progression of IBM business computers. As the operating systems and computers were enhanced, so too was the RPG language. The future of both the RPG computing language and the line of IBM midrange business systems were to be irrevocably linked.

In 1969, IBM announced its first midrange business system called the System/3. As with computers,

operating systems, and the RPG language, input/output processing had begun to evolve too. First there was the 96-column card; the card was physically smaller and had 16 more bytes of data. Then came the removable disk packs. Just imagine, 5 million bytes (5 megabytes) on a single disk. The removable disk packs on the System/3 could be swapped in and out at will, allowing software designers considerably more freedom in their program design. The evolution of the RPG language resulted in a new version of the language, called RPG II. The language had been modified to deal with the new ways to store and process data.

In the mid-1970s, IBM announced the System/32, which was the next business system that was to be centered on the RPG language. The System/32 was a smaller, more affordable midrange business system. Depending upon a person's perspective, it could be said that the System/32 was the original personal computer (never mind that the physical size of the System/32 was only a little less than that of the original Volkswagen Beetle). The System/32 had the processor, a fixed-disk hard drive, a printer, and a cathode-ray tube (CRT) display all included in a single system. The fixed-disk hard drive and the CRT were both new technologies and would be deemed woefully inadequate by the personal computer users of today. The fixed disk came in sizes of 5 and 13 megabytes and was physically about the size of a car tire. The CRT came with a whopping 8 lines of 40-character green on black text that was very difficult to read.

Although the description of the System/32 seems hugely inadequate by today's standards, it was extremely popular in the business community. RPG II applications were custom written for almost every type of business. IBM had capitalized on the fact that the business community's decision to purchase computer systems was largely predicated upon having a tailor-made software application available for their type of business.

A few years later, IBM announced the System/34. Along with the announcement of the System/34 came the first generation of multiuser RPG code. The displays were considerably larger than that seen on the System/32 and had 1920 addressable characters that could be displayed at any given time. For the first time, data entry could be done right at the workstation. Interactive data entry was a considerable departure from the batch methods of the past. It allowed developers to create programs that would edit entries as they were keyed while the input documents were readily at hand. The gain in productivity was widely lauded in the business community. Programs written in RPG allowed them to do more with less.

The changes that had been introduced to RPG II to facilitate the multiuser capability required that a new type of file be introduced, aptly named the workstation file. The new workstation file could be written to and read, much like files on other types of electronic media. These enhancements to the RPG language were major, and the productivity gains were significant.

RPG was the native language on the System/34, even though the system also supported other languages such as BASIC and COBOL. There were two principal reasons for this:

1. Many of the RPG business applications that were written for the System/32 had been easily migrated to the System/34. Implementation of interactive workstation programming and multiuser support only enhanced their value to the business community.
2. RPG is what is commonly referred to as a high-level language (HLL). In other words, RPG code is written in source code that is then compiled into an object module. Since much of the processing is done at compile time, programs are able to run more efficiently and better utilize system resources.

Scalability was another reason for the popularity of the System/34. This new system could be configured to be considerably larger than its predecessor, the System/32. A system that could support 256 megabytes of hard disk storage as well as 256 megabytes of memory was a considerable jump from what was being offered just a few years before.

RPG split into diversely different universes in the late 1970s; that of the System/36 and that of the System/38. Once again, the evolution of IBM hardware and operating systems was to have a direct and critical impact on the development of the RPG computing language.

The System/36 was mostly thought of as an extension of the System/32 and System/34 business systems. RPG II code on the System/36 was very similar to the code that was run on the System/34. A few enhancements had been made to the RPG II language, but most of the developments had been in the area of scalability and an on-line help system that was considered state of the art at that time. The System/36 became known as extremely dependable and easy to use, both of which were extremely important to the business community.

Meanwhile, back in Rochester, MN (where the IBM midrange systems were all developed), there was a considerable buzz about a new technological marvel

named the System/38. It was considered to be revolutionary for a number of reasons, not the least of which was a new concept referred to as a fully integrated database.

An integrated database meant that data stored on the system was actually defined and described at the operating system level, instead of from within the programs. This issue was important because file changes meant that the database definition only needed to change and not necessarily all of the programs using the file. Obviously, programs using the new or redefined attributes would need to be changed, but not the definition in all programs using the file. The concept driving the fully integrated database is that the programmer does the work once and does not have to do it again. Data is defined only once, regardless of the programs and applications that use it.

To utilize this new integrated database capability, the RPG language had to change. The new version was logically named RPG III. In addition to the ability to work with files externally defined in the integrated database, RPG III included a relatively new concept referred to as "structured code." Structured code was introduced to the RPG language to increase productivity and enhance program maintenance.

As it worked out, the IBM System/36 was a huge commercial success, while the System/38 languished. The System/38 was perceived as slow and difficult to use. IBM had underestimated the amount of memory required by the new operating system and had shipped systems that did not have enough memory to operate in the environments they were designed for. To make things worse, programmers experienced in the midrange market were not fully prepared for the learning curve required to utilize the externally defined display and database files. IBM made another error in judgment when it put the System/38 on the open market before many of the programming tools were fully ready. Unfortunately for IBM, the System/38 never fully recovered from these initial faux pas.

In 1988, IBM announced the AS/400. Code named Silverlake, the AS/400 was designed to deliver the ease of use that the System/36 was known for combined with the technical innovations that had been announced with the System/38. To make the transition easier on System/36 programmers, the system was shipped with an optional "System/36 Environment" that was designed to make the average System/36 programmer feel more at home. Unfortunately, IBM repeated the same mistake they made on the System/38 by initially shipping a number of the low-end systems with inadequate memory, causing some of the initial reviews of the AS/400 to be labeled as a "dog." IBM was quick to address the problem this time, however, and the AS/400 went on to become the most successful business computer of all time.

In 1995, IBM announced the next generation of the RPG language. RPG IV and the Integrated Language Environment (ILE) was IBMs first real attempt at making the AS/400 a homogenous platform, appealing to vendors that produced software written in languages other than RPG. The ILE was designed to allow developers to create systems composed of programs written in a variety of different languages. For the first time in the history of midrange systems, developers could design systems that included programs written in a wide variety of computing languages. While the ILE added flexibility did attract new systems and languages to the platform, RPG has remained the predominant language on the platform.

Thousands of RPG custom applications, written over four decades, continue to be what drives the popularity of the RPG language. Even though the RPG language has evolved, much of the code written in the 1970s, 1980s, and 1990s was retrofitted for Y2K and continues to run on systems all over the world. Tools have been introduced to modernize the user interface (UI) of this legacy code, but much of the underlying code remains intact with very few changes.

RPG continues to evolve, and many new applications written in RPG IV have been designed to take advantage of the evolutional development that has continued to occur. A few of these enhancements include free-form expressions, access to new data types, and application program interfaces (APIs) that allow the programs to utilize functions like data encryption, user spaces, or functions written in C.

II. STRENGTHS OF THE REPORT PROGRAM GENERATOR IV LANGUAGE

Since RPG was designed specifically for business, it is and always has been very good at file management and reporting. It takes very little RPG code to read or write to a file, or report data from one or more data files.

The RPG language was designed with programmer productivity in mind. File management and printer controls are largely handled by the system with little or no programmer intervention. The integrated database that is native to the operating system of the AS/400 makes file definition and declaration simple. The programmer simply tells the program which files to use and, optionally, when to read or write to them. The system handles the rest of the file management.

The RPG language has consistently evolved to keep up with changes in technology. The ILE was introduced to allow for programs in different languages to coexist in the same run-time environment. Free-form specifications were introduced to allow for more flexibility and productivity. Access to new data types was implemented, as the types of objects stored on a computer grew. APIs were added so RPG programs could communicate with a wider array of systems and programs written in other languages.

III. WEAKNESSES OF THE REPORT PROGRAM GENERATOR IV LANGUAGE

RPG has traditionally been a "green-screen" language. The native workstation programming provided for in the RPG language could probably best be described as UI challenged. There are numerous aftermarket tools that let the programmer provide a graphical UI in Windows or through a browser (such as Internet Explorer or Netscape), but there is no native support for running "thin client." These aftermarket products have provided a viable upgrade path for companies running legacy RPG code, but these tools are probably not the best answer for new systems being developed today.

To be fair, it must be said that IBM has made a valiant attempt at modernizing the RPG UI with a subset of the Websphere development suite called VisualAge for RPG. And while VisualAge for RPG is GUI and has all of the advantages of RPG IV, it is not much help for developers working with legacy applications. In those cases, the entire UI would need to be redesigned.

The other major weakness of the RPG language is that hardware vendors other than IBM have never embraced RPG as a mainstream computing language. There have been attempts by other vendors to utilize RPG, and there have even been vendors that developed RPG for the Windows operating system, but in the past, none of these attempts have been met with a great deal of commercial success.

At the time this article was written, there had been over 600,000 AS/400s (now relabeled the iSeries 400) shipped worldwide, making it the most popular business system ever. And even though many other languages will run on IBM's AS/400, RPG is still the predominant language in the IBM midrange market. Despite this fact, RPG is a very unique language, making it difficult to find new programming talent. Most schools are hesitant to teach a language that is considered by some to be both proprietary and antiquated by virtue of its 40 years in existence.

IV. DEFINING THE REPORT PROGRAM GENERATOR IV LANGUAGE

RPG is unlike any other computer language. Beyond the fact that it was designed specifically for the business community, its syntax, format, and run-time cycle are unique to the RPG language as well.

For the remainder of this article, the RPG IV will be the language discussed, unless specifically stated otherwise. That being said, many of the properties and characteristics discussed here have been around since the inception of the RPG language.

RPG IV is columnar in nature and employs several different formats called specifications. Each specification performs a particular function, and the columns for each of the specifications are different. With exception to continuation lines and comments, each specification is composed of a 96-column record. The character in position six of each record is used to designate the type of specification. The records do have to be laid out in a specific order, but most of the record types are optional, depending upon the desired function of the program. At this time, there are seven types of specifications, and they appear in the desired sequence in Table I.

A. The Report Program Generator Cycle

The RPG cycle has been around since the beginning of the RPG Language. The idea behind the RPG cycle was to provide the developer with a preset series of processes where much of the programs work could be performed automatically by the system. The premise was logical. Most business programs are going to require the same logical flow: perform initial overhead functions, process variable definitions, process file input/output operations, and produce output.

Table I Seven Types of Specification

Specification designator	Specification description
H	Header specification
F	File specification
D	Definition specification
I	Input specification
C	Calculation specification
O	Output specification
P	Procedure interface specification

The RPG cycle provides the processing foundation that the RPG module is based upon. The RPG cycle will initialize variables, open and optionally read files, process the logic cycle, reset indicators, and produce output all within the constructs of the RPG cycle. Figure 1 is an example of the RPG logic cycle .

Programs may optionally be designed where a file may be designated as a primary input file, and a record will be read with each program cycle. When the last record is read, the "last record" indicator will come on, the last record calculations and output operations will occur, and the program will terminate.

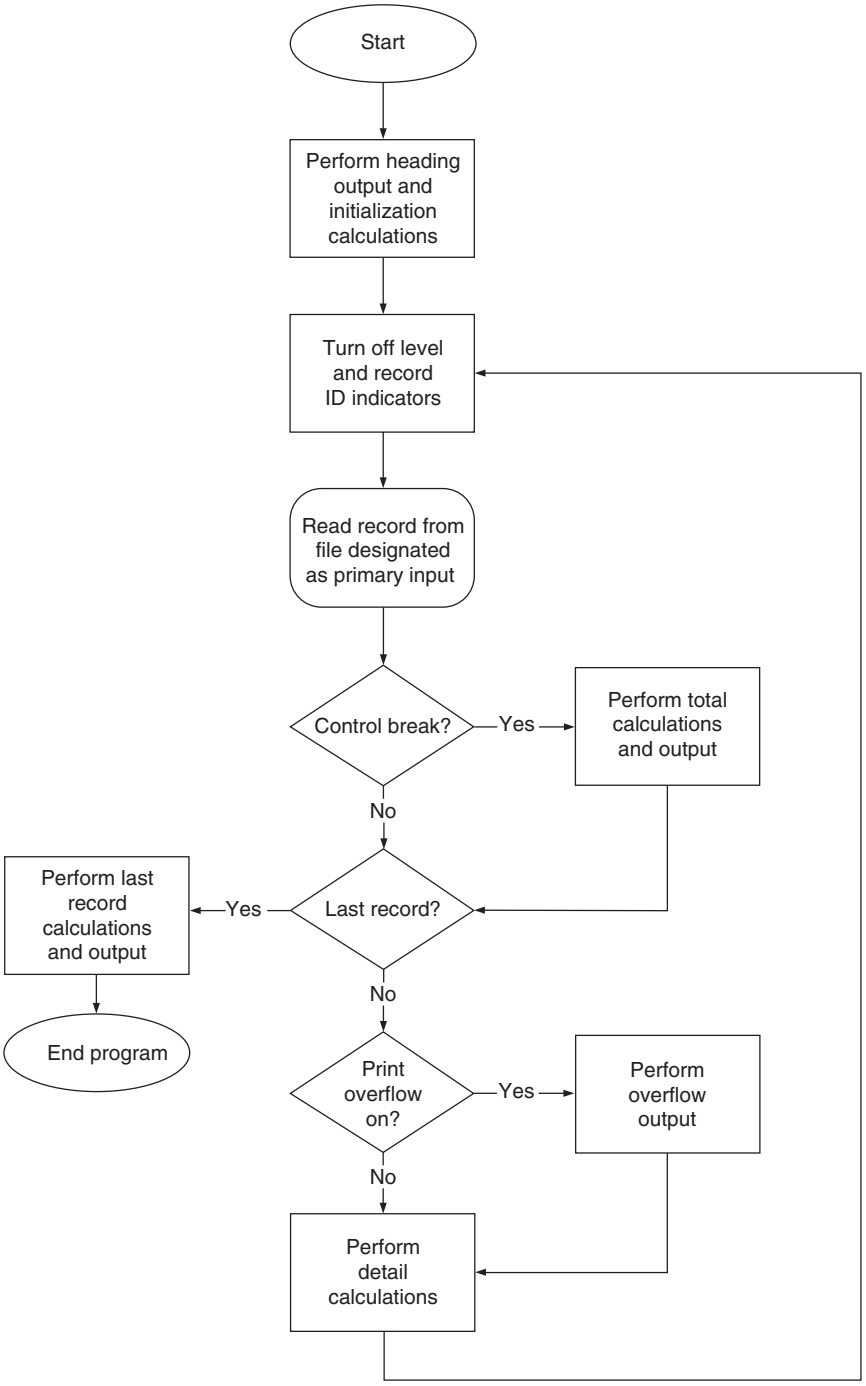


Figure 1 RPG logic cycle.

The simplicity of the RPG programs written in the 1960s has gone the way of the troglodyte and the dinosaur, making the RPG cycle a much less important part of today's RPG program. Today's business programs require a much more diverse group of input/output objects, and many of them are processed randomly rather than sequentially. It is for this reason that taking full advantage of the RPG cycle has largely fallen out of favor. The RPG cycle is still useful for handling program overhead jobs like opening and closing files, performing initialization processing, and performing last record calculations and output, but the cycle is rarely used as extensively as it one was.

B. Report Program Generator Is a High-Level Language

RPG is what is commonly referred to as an HLL. What this means is that the source code is written in RPG, and then the source code is run through a *compiler* that edits and, errors in the code notwithstanding, compiles the source code into what is called an object module. It is the object module that actually gets run when a program gets called.

Programs written in an HLL perform much of the overhead processing, required when a program is called, at the time the program is compiled. This is far more efficient than performing all overhead processing when the program is called. Programs that perform all overhead functions when the program is called are generally referred to as *interpretive languages*. The performance of programs written in HLL is substantially better than that of programs written in an interpretive language.

C. Naming Conventions

Naming in RPG is pretty flexible, as long as the programmer does not mind staying within a 10-character confine. RPG does insist that variables do begin with an alphanumeric character or a handful of special characters such as @, #, or $. Upper- and lowercase can be mixed to enhance readability, but the compiler will interpret all characters as uppercase during the compile, meaning that case alone will not differentiate variables.

D. Indicators

RPG provides a series of logical variables called indicators. Indicators may have only two values: on or off. There are two kinds of indicators: those defined by the system and those defined by the programmer.

System-defined indicators have been provided to serve a particular purpose. The last record (LR) indi-

Table II Indicator Definitions

Indicators	Use
KA–KN KP–KY	Tells the system that a function key has been pressed when running an interactive program. For example, if F1 is pressed, indicator KA will be turned on indicating that this is the case. The F2 key is represented by KB, and so on.
L0–L9	Level indicators are used in conjunction with the RPG cycle. Level indicators may be assigned to control fields within an input file to indicate when a change occurs in the control field. The indicator will remain off as long as the value of the control field remains the same. When the value of the control field changes, the specified control indicator will come on, telling the program that a level break has occurred.
LR	If a program was designed to utilize the RPG cycle, the last record (LR) indicator will be automatically turned on when the LR of the primary input file has been read. In other cases, the LR indicator is used to tell the program to perform LR calculations and terminate.
MR	The matching record (MR) indicator may be used when a program is written to utilize primary and secondary input files in a program designed to use the RPG cycle. The MR indicator is assigned to control fields within the two files. The MR indicator will be automatically turned on when the control fields from the two files match.
1P	The first page indicator (1P) is used for printing reports. The 1P indicator is generally used to print heading information that is to appear on the first page of a report.
OA–OG OV	Overflow indicators are used to tell the program that output to a specific print job has reached the end of the page. The reason there are multiple overflow indicators is because a single program may output numerous print outputs.

```
HKeywords+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Figure 2 The header specification.

cator, for example, is used to tell the program to complete the RPG cycle and terminate. Table II takes a look at a few of the other system-defined indicators.

E. User-Defined Indicators

User-defined indicators were originally restricted to 01–99. An indicators value could only be a "1" (on) or "0" (off). These days other logical variables can be used as indicators as well, but they may not be used as extensively as the original indicators 01–99.

User-defined indicators can be used for a variety of uses such as controlling output or communicating with workstation display files. The indicators' role within the RPG programs logic is not what it once was, however, due to the advent of the built-in function, which will be discussed later.

The role of indicators has diminished somewhat throughout the years of evolution of the RPG language, but they are still useful under certain conditions.

F. The Header Specification

The header specification is optional and is used to describe the RPG module being built. The header specification is a unique specification, in that it is not columnar in nature. In fact, the entire specification is keyword dependent. The format of the header specification may be seen in Fig. 2.

Keywords for this specification are used to describe a variety of program environmental functions, for instance, the defaults for how date or time fields should appear, defaults currency symbols, or the copyright statement that should be embedded into the object code module.

As shown in Table III, column position 6 of the header specification is columnar, and positions 7–80 are reserved for keywords controling the RPG module functions.

G. The File Specification

The file specification is also optional, but must be used if the RPG module will reference database, display, or printer files. Like most specifications, the file specification is a combination of columnar-defined fields and free-form keywords. An example of a file specification is shown in Fig. 3.

As shown in Table IV, column positions 1–43 of the file description specification are columnar, and positions 44–96 are reserved for keywords describing the file in question. The keywords utilized are largely dependent upon the file being described. For example, a printer file may have keywords describing the print environment, where a database file may have keywords describing blocking factors or a key location.

H. The Definition Specification

The definition specification does exactly what its name implies. Variables in RPG modules may be defined several different ways. For example, if your RPG module includes a file specification for an externally described file, the programmer does not need to define fields from that file within the module in order to reference them. It is done for the programmer when the module is compiled. On the other hand, the program may reference a program-defined file, in which case the programmer would use input specifications to define the variables within the file. The programmer may also define variables from within the calculation specifications of the program, although that practice is generally discouraged within the RPG IV programming community. Other variables are defined by using the definition specification.

The definition specification is used to define variables, compile and run-time arrays, data structures, and a variety of other data elements to the RPG module. Once again, the specification includes both fixed column attributes as well as free-format keywords used to further define, describe, or initialize the variable being defined. The keywords allowed on each line are

Table III Column Definitions for the Header Specification

From	To	Description
6	6	Form type code identifying the header specification functions
7	80	Optional keywords that control how the RPG module functions

```
FFilename++IPEASFRlen+LKlen+AIDevice+.Keywords++++++++++++++++++++++++
```

Figure 3 The file specification

predicated on the type of variable being defined. An example of the definition specification is shown in Fig. 4 and Table V.

I. The Input Specification

The input specification is used primarily to define files that have not been defined using the AS/400s integrated database. In these cases, the data is basically a "flat file" where all data definitions must be made within the program or module itself. Although this practice is generally discouraged, it still may be found under certain conditions.

Input specifications are also used to assign the control break or "level" indicators. The fields from the externally described files are presented along with the appropriate level indicator that is to be turned on when a control break occurs.

The input specification may also be used to rename or redefine fields from an externally defined file. One example of this might be where the programmer experiences name conflicts from two different input sources. One of the fields would need to be renamed to prevent potential conflicts when the program was run.

J. The Calculation Specification

The calculation specification is the heart and soul of most RPG modules. As the name implies, it is where the programmer codes his or her instructions to tell the system what they want the program to do. All operational instructions are put forth in the calculation specification. This includes, but is not limited to, file operations, mathematical computations, data manipulation operations, and requests for output to printers or other output devices.

The calculation specification is a combination of columns and free-form expression. The operation performed is predicated upon an element called the operational descriptor or "op-code." The op-code consists of one of two possibilities: (1) an operational conditioner controlling logic such as `IF, THEN,` or `ELSE`, or (2) an operand telling the program what function the specification is to perform such as `READ, GOTO,` or `DO`. In either case, the columns in positions 1–35 are fixed and nonnegotiable. Positions 36–80 may be either columnar or free-format keywords depending upon the op-code that was used. This may seem a little confusing, but just remember that the idea is to do more operations with less code. A partial definition of the columns is presented in Fig. 5 and Table VI.

Table IV Column Definitions for the File Specification

From	To	Description
6	6	Form type code identifying the file specification
7	16	Name of the file
17	17	File usage: I for input processing, O for output, U for update, and C for combined usage
18	18	Defines how the file is to be used: P for primary input, S for secondary input, F for fully procedural, and so on
19	19	Designates that the end-of-file indicator be turned on when the end of a file designated for primary or secondary input is reached
20	20	An "A" may be specified, telling the program that records may be added to this file
22	22	Tells the system if the file is defined within the program or externally by the database
23	27	Length of the record to be processed
29	33	Length of key fields
35	35	Tells the system if access to the file will be keyed or sequential
36	42	Type of device or file
44	80	Keywords further defining the file

```
DName++++++++++ETDsFrom+++To/L+++IDc.Keywords++++++++++++++++++++
```

Figure 4 The definition specification.

V. THE REPORT PROGRAM GENERATOR SAMPLE PROGRAM

The sample program in Fig. 6 represents a simple RPG IV print program. This program was designed to print a list of a master file, named CUSTOMER. The program will print the list by record key or alphabetically by customer name, depending upon the value passed to the program when the program is called. A sample of the output for this program may be found in Fig. 7.

The source code for the sample program begins with file specifications for two different logical views of a customer file. The first view of the file (CUSTOMER) is by key, and the second view (CUSTOMERLF) is by customer name. The first specification for the customer file tells the system that the file will be used for input, it is a full procedural file (instead of primary or secondary input), it is externally defined in the integrated database, the access is keyed, and the file may be found on disk. The specification also includes the USROPN keyword, indicating that opening and closing of the file will be done internally within the program, instead of letting the system do it. The program is coded that way because only one of the views of the customer file will be used, depending upon the value of the parameter passed to the program.

The second file specification is for the CUSTOMERLF file, which is a logical view of the CUSTOMER file that represents an alphabetical index by customer name. The specification is very similar to the first, with one notable exception. The programmer has added the RENAME keyword to rename the record in the file from CUSREC to CUSBYNAME. This was done because the compiler would not allow duplicate record names to be used.

The last file specification is for the system print file, QSYSPRT. The specification tells the system that the file is used for output, the file is defined within the program, and the file is a printer. The programmer also added the OFLIND keyword assigning the *INOF indicator to the printer overflow condition. The indicator will be automatically turned on when a page break is sensed.

Next, we have definition specifications. The first field defined is named header. The field is a stand-alone definition, is 40 characters in length, and is alphanumeric. The field has also been initialized with a constant that will be used later. Next, the PRINTDATE and PRINTTIME variables are defined with the date and time data types, respectively. The TIMFMT and DATFMT keywords are added to tell the system that the programmer is interested in using the *USA formats of these data types. Last, but not least, the SEQUENCE, TIME, and TOTAL variables are defined. The TIME and TOTAL variables are both designated as signed numeric, with zero decimal positions.

The calculation specifications begin with the PLIST op-code, telling the system to look for the following

Table V Column Definitions for the Definition Specification

From	To	Description
6	6	Form type code identifying the definition specification
7	21	Name of the object being defined
22	22	An “E” is designated here if the object being defined is done so externally from the program
23	23	A “U” is specified if the object being defined is a data area
24	25	Characteristic of data being defined; these include constants, data structure, stand-alone fields, and more
26	32	Starting location (within the structure)
33	39	Ending location (within the structure) or length
40	40	Type of data being defined; types include character, numeric, date, time, integer, packed decimal, time stamp, and many more
41	42	Decimal positions (for numeric data)
43	80	Keywords further describing the object being defined

```
CL0N01Factor1+++++++Opcode&ExtFactor2+++++++Result++++++++Len11D1HiLoEq
```

Figure 5 The calculation specification.

parameter list. The PARM op-code designated on the next line defines the input parameter named sequence (which was previously defined in the definition specifications). The following calculation specifications make up what is commonly referred to as the detail calculations. Within the detail calculations, the programmer coded the program to execute a loop by utilizing the DOU or Do Until op-code. The loop tells the program to execute the code between the DOU and ENDDO op codes until an end-of-file condition is encountered. As the loop is executed, the %EOF or end-of-file condition will be changed when the end of the file is encountered. The way the programmer coded the loop, the value of the %EOF built-in function will determine when the code within the loop is terminated.

Within the logic loop, the programmer reads either the CUSTOMER or CUSTOMERLF file depending upon the value of the parameter passed to the program when the program is called. If the SEQUENCE parameter is passed in as an "N," the CUSTOMERLF file will be read and the list will be printed alphabetically by customer name. Otherwise, the list will be printed by the CUSTOMER file key. Once the record is read, the TOTAL variable is incremented by 1. The TOTAL variable will be printed at the end of the report to indicate how many records were processed. The EXCEPT op-code tells the program to print all output that has been designated with the EXCEPT name of DETAIL. When the ENDDO op-code is encountered, control will be transferred back to the top of the loop. When control is passed out of the loop, the LR indicator (*INLR) is turned on and the program is terminated.

Next, in Fig. 6, we see the initialization subroutine, *INZSR. The initialization subroutine is actually performed prior to execution of the detail calculations previously discussed. Within this subroutine, the programmer moves the system date, *DATE, into the print variable named PRINTDATE. Next, the programmer performs the TIME op-code, which moves the system time to a work field named TIME. The work field is unformatted, however, and the time is represented with hours, minutes, and seconds. So to get the data into a more readable format, move the value placed in the TIME variable into the print variable named PRINTTIME. Next, check the value of the input parameter and open the appropriate file. The HEADER variable is also modified depending upon the input variable, and then the heading for the first page of the report is printed. Subsequent page headings will be printed automatically by the system with automatic page overflow handling.

For the purposes of this sample program, the programmer elected to define the printer file within the program. He or she could have just as easily created an externally described printer file, negating the need for output specifications.

The output specifications begin with the QSYSPRT file name. The HEADING except name tells the system

Table VI Column Definitions for the Calculation Specification

From	To	Description
6	6	Form type code identifying the calculation specification
7	8	Level indicator (L1–9), subroutine designation (SR), or blank
9	11	Conditional indicators; these indicators may be used to control whether the specification is executed or not
12	25	Factor 1 is an optional entry conditionally used by certain operations; for instance, a key list or value may be indicated when trying to retrieve a record from an indexed file
26	35	Operation code and extender; the op-code is used to indicate the operation to be performed and the extender is used to modify or further describe the operation
36	49	Factor 2 is used with certain op-codes depending upon the requirements
50	63	Optional result field; this field is required depending upon the op-code utilized; for example, the result field may be the name of a variable that should contain the result of a numeric operation
36	80	Extended factor 2 is available to certain op-codes; as a general rule, extended factor 2 would contain an expression; this field may be continued on to the next specification if needed

```
FFilename++IPEASFRlen+LKlen+AIDevice+.Keywords+++++++++++++++++++++++++
fCustomer  IF   E           k Disk    UsrOpn
fCustomerLFIF   E           k Disk    UsrOpn Rename(CusRec:CusByName)
fQsysprt   O    F  132        Printer OflInd(*InOF)

DName++++++++++++ETDsFrom+++To/L+++IDc.Keywords+++++++++++++++++++++++++
d Header           s              40a   inz('Customer List by')
d PrintDate        s                d   datfmt(*usa)
d PrintTime        s                t   timfmt(*usa)
d Sequence         s               1a
d Time             s               6s 0
d Total            s               6s 0

CL0N01Factor1+++++++Opcode&ExtFactor2+++++++Result++++++++Len++D+HiLoEq
c     *Entry        Plist
c                   Parm                    Sequence

c                   DoU       %EOF
c                   If        Sequence = 'N'
c                   Read      CustomerLF
c                   Else
c                   Read      Customer
c                   EndIf
c                   If        not %EOF
c                   Eval      Total = Total + 1
c                   Except    Detail
c                   EndIf
c                   EndDo

 * Turn on Last record indicator
c                   Eval      *InLR = *On

csr   *InzSr        BegSr
 * Format print date and time
c     *usa          Move      *Date         PrintDate
c                   Time                    Time
c     *hms          Move      Time          PrintTime

 * Establish sequence of the data and print the heading
c                   If        Sequence = 'N'
c                   Open      CustomerLF
c                   Eval      Header = %Trim(Header) + ' Name'
c                   Else
c                   Eval      Header = %Trim(Header) + ' Key'
c                   Open      Customer
c                   EndIf
c                   Except    Heading
csr                 EndSr

OFilename++DF..N01N02N03Excnam++++B++A++Sb+Sa+.....
oQsysprt    E           Heading        2 02
o          OR    OF
o                      PrintDate           12

o                      Header              65
o                      PrintTime           80

o           E          Heading        1
o          OR    OF
o                                         15 'Customer Number'
o                                         45 'Customer Name'

o           EF         Detail         1
o                      Customer#           15
o                      Custname            65

o           T     LR                     2
o                      Total          j     20
o                                         34 'Total Records'
```

Read customer file by name or key, depending upon the value of the parameter passed

Initialization subroutine is called prior to executing detail calculations

Heading is printed during the initialization subroutine or when overflow is turned on

Figure 6 Sample RPG module.

04/03/2001 Customer List by Name 02:45 PM

Customer Number	Customer Name
18	BAYSIDE REALTY
120	BILLS MUFFLERS
16	CHULA VISTA SPRINKLER SUPPLY
12	ENCINITAS PAINT
19	ESCONDIDO NURSERY
20	HOME DEPOT
10	JEFF'S COMPUTER CABLES
13	JOLLYTIME LANDSCAPE
140	NORTHERN LIGHTS
130	RONS BUSINESS EQUIPMENT
17	SANTEE ELECTRICAL
11	SORRENTO ELECTRICAL SUPPLY
15	SOUTH BAY PLUMBING
21	WESTERN LUMBER
14	WESTONHILL PROPERTY MANAGEMENT

15 Total Records

Figure 7 Output from sample RPG module seen in Fig. 6.

to perform the associated heading output anytime the EXCEPT op-code is executed. Also note that the output extension lines directly following the exception output specifications. These lines tell the program to automatically print the header information if the overflow indicator (OF) is turned on.

The detail output is performed anytime the EXCEPT op-code is executed with the DETAIL except name. The F immediately following the exception designator tells the system to check for a printer overflow condition each time a detail line is printed. If the condition is encountered, the overflow indicator will be turned on.

At the end is the total last record output. This output will print automatically when the LR indicator is turned on and the program is being terminated.

VI. SUMMARY

The sample program in Fig. 6 reflects very little of what the RPG IV language is capable of, but it does demonstrate some of what the language was originally developed to do. The RPG IV language was designed to produce readable output with a minimal amount of code. It was designed to make efficient use of processors, input/output devices, and a programmer's time.

Is there a future for the RPG language? Only time will tell. The language has evolved over the course of four decades and continues to be enhanced on a regular basis. Will RPG ever successfully migrate to other platforms and operating systems? Multiple vendors (including IBM) are currently selling tools that allow RPG to be coded on a Windows operating system. Only time will tell if these vendors become commercially successful.

The RPG language is one of the only languages developed specifically for business. It seems only fitting that it will be business that makes the ultimate decision as to whether the RPG language will survive.

SEE ALSO THE FOLLOWING ARTICLES

COBOL • Programming Languages Classification

BIBLIOGRAPHY

Cozzi, R., Jr. (1999). *The modern RPG IV language.* 2nd ed. Carlsbad, California: IIR Publications.

Farr, G., and Topiwala, S. (1996). *RPG IV by Example.* Loveland, Colorado: Duke Publications.

Meyers, B., and Yeager, J. (2000). *Programming in RPG IV.* 2nd ed. California: Duke Publications.

Pence, D., and Hawkins, R. (2000). *RPG IV at Work.* California: IIR Publications.

Smith, B. R., *et al.* (2000). *Who knew you could do that with RPG IV? A sorcerers guide to system access and more.* Rochester, Minnesota: International Business Machines Corporation.