IEC 61970-402

Edition 1.0 2008-06

INTERNATIONAL STANDARD

Energy management system application program interface (EMS-API) – Part 402: Common services

IEC 61970-402:2008(E)

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

- Catalogue of IEC publications: www.iec.ch/searchpub

The IEC on-line Catalogue enables you to search by a variety of criteria (reference number, text, technical committee,…). It also gives information on projects, withdrawn and replaced publications.

- IEC Just Published: www.iec.ch/online_news/justpub

Stay up to date on all new IEC publications. Just Published details twice a month all new publications released. Available on-line and also by email.

- Electropedia: www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing more than 20 000 terms and definitions in English and French, with equivalent terms in additional languages. Also known as the International Electrotechnical Vocabulary online.

- Customer Service Centre: www.iec.ch/webstore/custserv

If you wish to give us your feedback on this publication or need further assistance, please visit the Customer Service Centre FAQ or contact us:

Email: csc@iec.ch
Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00

IEC 61970-402

Edition 1.0 2008-06

INTERNATIONAL STANDARD

Energy management system application program interface (EMS-API) – Part 402: Common services

INTERNATIONAL ELECTROTECHNICAL COMMISSION

PRICE CODE W

ICS 33.200

ISBN 2-8318-9850-1

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

ENERGY MANAGEMENT SYSTEM APPLICATION PROGRAM INTERFACE (EMS-API) –

Part 402: Common services

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
5) IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with an IEC Publication.
6) All users should ensure that they have the latest edition of this publication.
7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 61970-402 has been prepared by IEC technical committee 57: Power systems management and associated information exchange.

The text of this standard is based on the following documents:

FDIS	Report on voting
57/928/FDIS	57/947/RVD

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

A list of all parts of the IEC 61970 series, under the general title *Energy management system application program interface (EMS-API)*, can be found on the IEC website.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2.

The committee has decided that the contents of this publication will remain unchanged until the maintenance result date indicated on the IEC web site under "http://webstore.iec.ch" in the data related to the specific publication. At this date, the publication will be

- reconfirmed;
- withdrawn;
- replaced by a revised edition, or
- amended.

A bilingual version of this publication may be issued at a later date.

INTRODUCTION

This standard is one of the IEC 61970 series parts that define an application program interface (API) for utility operational systems. This standard is based upon the work of the Electric Power Research Institute (EPRI) Control Center API (CCAPI) research project (RP-3654-1).

The IEC 61970-4xx series specifies a set of interfaces that a component (or application) should implement to be able to exchange information with other components and/or access publicly available data in a standard way. The IEC 61970-4xx series component interfaces describe the specific event types and message contents that can be used by applications independent of any particular component technology. The implementation of these messages using a particular component technology is described in the IEC 61970-5xx series of documents. Thus, IEC 61970-4xx documents describe a Platform Independent Model (PIM), while IEC 61970-5xx documents describe a Platform Specific Model (PSM).

IEC 61970-402 contains API services that are considered to be foundational. As such, all the other parts in the IEC 61970-4xx series assume their existence. As a result, the services described in IEC 61970-402 are required for any component that complies with the IEC 61970-4xx series.

The component interface specifications refer to entity objects for the power system domain that are defined in the IEC 61970-3xx series: Common Information Model (CIM).

This standard contains normative and informative text. Clauses are marked as normative or informative. A subclause inherits its parent clause’s label unless overridden by the lower level subclause label.

ENERGY MANAGEMENT SYSTEM APPLICATION PROGRAM INTERFACE (EMS-API) –

Part 402: Common services

1 Scope

This International Standard provides the base functionality considered necessary and common that is provided by neither the normative standards incorporated by reference nor the new APIs specified in the IEC 61970-403 to IEC 61970-449 [1)] generic interface standards. An application is expected to use the Common Services in conjunction with the generic interfaces. These application category independent interfaces include:

- IEC 61970-403: Generic Data Access (GDA)
- IEC 61970-404: High Speed Data Access (HSDA)
- IEC 61970-405: Generic Eventing and Subscription (GES)
- IEC 61970-407: Time Series Data Access (TSDA)

To support these objectives, the Common Services are divided into three categories:

a) Resource Identifiers – A common way of identifying classes, class attributes, and object instances.
b) Resource Description – A common way of encoding values associated with classes, class attributes, and object instances.
c) Views – A common way of presenting classes, class attributes, and object instances via hierarchies.

IEC 61970-402 contains API services that are considered to be foundational. As such, all the other parts in the IEC 61970-4xx series assume their existence. As a result, the services described in IEC 61970-402 are required for any component that complies with the IEC 61970-4xx series of standards.

Though the target of this IEC standard includes the control center technical domain, common services encompass a general set of concepts that can be applied to many types of systems. Examples of these systems include:

- Distribution management systems
- Work or asset management systems
- Geographic information systems
- Outage management systems
- Other types of operational business systems.

In recognition that the integration between applications in two or more of these systems is often necessary, the intent of this specification is to address general common service requirements to the extent that they are common to different types of systems while effectively addressing the control center needs.

[1)] At this time, only parts 402 to 408 exist. Additional generic services beyond are not yet under consideration.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61970-1, *Energy management system application program interface (EMS-API) – Part 1: Guidelines and general requirements*

IEC 61970-2, *Energy management system application program interface (EMS-API) – Part 2: Glossary*

IEC 61970-401, *Energy management system application program interface (EMS-API) – Part 401: Component interface specification (CIS) framework*

IEC 61970-403, *Energy management system application program interface (EMS-API) – Part 403: Component Interface Specification (CIS) – Generic data access*

IEC 61970-404, *Energy management system application program interface (EMS-API) – Part 404: High Speed Data Access (HSDA)*

IEC 61970-405, *Energy management system application program interface (EMS-API) – Part 405: Generic Eventing and Subscription (GES)*

IEC 61970-407, *Energy management system application program interface (EMS-API) – Part 407: Time Series Data Access (TSDA)*

OMG, Utility Management System Data Access Facility, document formal/2002-11-08

3 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 61970-2 apply.

NOTE Refer to International Electrotechnical Vocabulary, IEC 60050, for general glossary definitions.

4 Background

This International Standard specifies Component Interface Specifications (CIS) for Energy Management Systems Application Program Interfaces (EMS-API). It specifies the interfaces that a component (or application) shall implement to be able to exchange information with other components (or applications) and/or to access publicly available data in a standard way (see IEC 61970-1 for an overview of these standards). The goal of the creation of this document is to improve the interoperability of control center applications and systems. This specification provides a mechanism for applications from independent suppliers to access IEC 61970 Common Information Model (CIM) data using a common Application Program Interface (API) for the purpose of supplementary processing, storage, or display.

IEC 61970-401, CIS Framework, provides an overview of the CIS documents in the IEC 61970-4xx series. It explains the separation of these specifications into two major groups. One group of standards, IEC 61970-402 to IEC 61970-449, defines a set of generic application independent services that a component shall use for exchanging information with another component or for accessing public data. However, as the generic interfaces do not specify what specific data is exchanged, interoperability between products using IEC 61970-402 to IEC 61970-449 is not guaranteed. The other group, IEC 61970-450 to IEC 61970-499, defines the information content conveyed using the generic services that a particular component or system exchanges with other components. While IEC 61970-402 to IEC 61970-449 specify application category independent message exchange mechanisms, IEC 61970-450 to IEC 61970-499 specify application category dependent CIM derived message contents.

The IEC 61970-4xx series of documents describes the specific event types and message contents that shall be used by applications independent of any particular component technology. The implementation of these messages using a particular component technology is described in the IEC 61970-5xx series of documents. As part of the IEC 61970-4xx series, this document contains a Level 1 Platform Independent Model (PIM) that describes in narrative terms (with text and the Unified Modeling Language (UML)) the interface functionality to be standardized. The IEC 61970-5xx series of documents describes a Level 2 or Platform Specific Model (PSM) that apply IEC 61970-4xx series specifications to particular platforms.

As explained in IEC 61970-401, a major aspect of the IEC 61970-4xx series is that they take maximum advantage of existing industry standards. Of particular importance are standards developed by the OPC (originally OLE for Process Control) and OMG (Object Management Group). However, these standards are missing certain functionalities considered important for the environment in which the IEC 61970 standards will be applied.

IEC 61970-402 contains API services that are considered to be foundational. As such, all the other parts in the IEC 61970-4xx series assume their existence. As a result, the services described in IEC 61970-402 are required for any component that complies with the IEC 61970-4xx series.

5 Common services identifiers (normative)

5.1 DAF resource identifiers interface

This document normatively includes by reference the modelling contained in the Utility Management System (UMS) Data Access Facility (DAF) resource identifiers module[2]. This module defines a generic means that shall be used to identify CIM resources (power system related classes, class attributes, and object instances). The DAF resource identifiers module defines several classes and one service as illustrated in Figure 1.

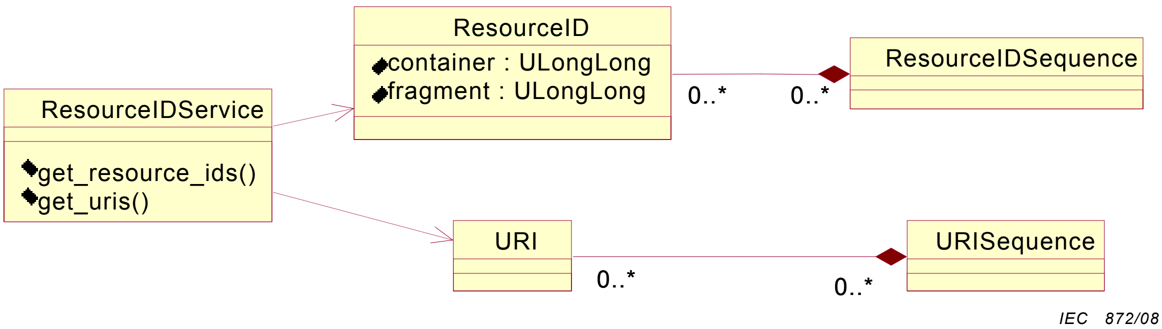

Figure 1 – DAF resource identifiers model

2) See OMG, Utility Management System Data Access Facility, document formal/2002-11-08.

Table 1 – DAF resource identifiers operations

Operation	Example signature	Throws
get_resource_ids	ResourceIDSequence get_resource_ids (URISequence uris)	LookupError
get_uris	URISequence get_uris (ResourceIDSequence ids)	LookupError

The DAF resource identifiers module employs Universal Resource Identifiers (URI's)[3] and Resource ID's to identify utility resources. A URI is a string used as a name that conforms to the Universal Resource Identifier standard (see reference 5). For example, a URI for the class of transformers might be "http://utility.com/Planning/Production/CIM-schema-cimu09a# Transformer". A resource ID on the other hand consists of a more compact 128 bit (16 octets) ID number. The DAF specification defines arrays of Resource ID's and URI's in ResourceIDSequence and URISequence respectively.

The DAF resource identifiers module requires that Universal Resource Identifiers (URI's) be unique and that within a DAF Server Resource ID's be unique. Furthermore, for any resource exposed by a DAF server there is only one associated resource ID and one associated URI. The DAF ResourceID Service includes two methods called get_resource_ids and get_uris to translate URI's to Resources and visa versa. The get_resource_ids method takes a set of URI's as input and returns an equal and corresponding set of Resource ID's as output. The get_uris method takes a set of Resource ID's as input and returns an equal and corresponding set of URI's as output.

5.2 Extended resource ID service

5.2.1 General

However, in practice it is possible that a single unique resource ID may be associated with many URI's[4]. For example, consider a resource ID service provider deployed for planning and operations applications. In this case, a resource such as a substation bus may be associated with two URI's (names) where each URI is associated with a different legacy application for example "utility.com/Planning/Production#Bus123" and "utility.com/EMS/Production#BusAB7". In order to deal with this more complicated use case, the Extended Resource ID Service adds a view name parameter to the DAF Resource ID get_uris method. The view name parameter is used to determine the scope of the URI.

The extended resource ID service also adds the ability to set URI's given a view name and to allocate resource ID's in a server as shown in Figure 2. Thus, the common services identifiers interface as described in Table 2, can be seen as a complementary extension of the DAF identifiers interface.

---

[3] IETF RFC 2396 for more information on URI's.

[4] For example, consider a IEC 61970-404: High Speed Data Access (HSDA) Server presenting nodes within an IECTC 57PhysicalView hierarchy. Because the HSDA interface does not support a client query to retrieve the Resource ID's of all breakers in a substation for example, a performance bottleneck can occur during HSDA subscription configuration when a client assembles a list of pathnames to subscribe to. However, the Resource ID's for all breakers in a substation can be retrieved using a IEC 61970-403: Generic Data Access query. The Extended Resource Identification Service provider can then be used to translate the retrieved Resource ID's to HSDA server pathnames since the combination of a HSDA Server name and a IECTC 57PhysicalView pathname provides a unique URI. Therefore, the Extended Resource Identifiers Service supplies required functionality to eliminate the subscription configuration bottleneck by allowing a client to pass in a view name to the Resource Identification Service provider when querying for HSDA pathnames.

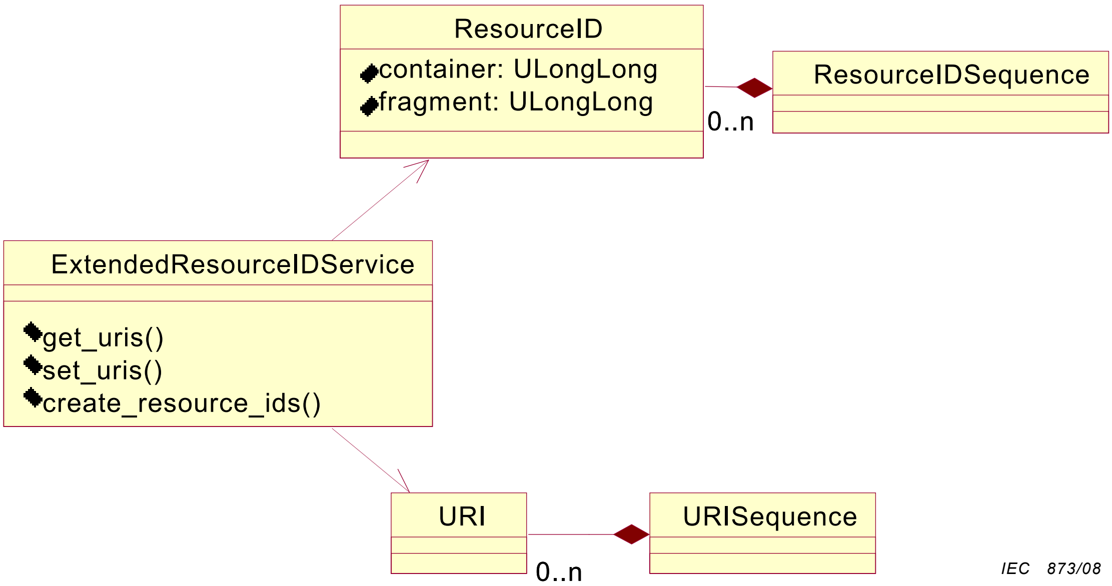

Figure 2 – GDA extended resource identifiers model

Table 2 – GDA extended resource identifiers operations

Operation	Example signature	Throws
get_uris	URISequence get_uris (char view_name, ResourceIDSequence ids)	LookupError
set_uris	void set_uris (char view_name, ResourceIDSequence ids, URISequence uris)	UpdateError
create_resource_ids	ResourceIDSequence create_resource_ids (ResourceID prototype, int how_many)	UnknownResource

5.2.2 get_uris()

This operation accepts a sequence of resource identifiers as well as a view name and returns the corresponding URI-references. The returned sequence is the same length as the input sequence and in the same order. If the server does not recognize a particular resource identifier or is not responsible for its translation, the operation will raise a LookupError exception.

5.2.3 set_uris()

This operation accepts a sequence of resourceID's and URI's as well as a view name[5]. Each URI is associated with each ResourceID within the scope of the view name. The ResourceID and URI sequences must be the same length. If the server does not recognize a particular resource identifier or is not responsible for its translation, an UpdateError is thrown.

[5] It is recommended but not mandatory that ExtendedResourceID clients that also act as IEC 61970-404:High Speed Data Access (HSDA), IEC 61970-407:Time Series Data Access (TSDA), or IEC 61970-505:Generic Eventing and Subscription (GES) servers use IECTC 57View derived pathnames as URI's and that these pathnames are derived from the one of the standard views such as IECTC 57PhysicalView. When the IECTC 57PhysicalView is used to create the pathname, the URI's used shall correspond to the IECTC 57PhysicalView pathname preceded by the name of the server. For more information on the format of path names, see IEC 61970-404: High Speed Data Access.

5.2.4 create_resource_ids()

This operation accepts a resource ID for a prototypical resource as well as an integer indicating how many new resource ID’s to create and returns one or more ResourceID values.

The return value is a sequence of valid ResourceID values. None of the resource identifiers designates an extant resource at the time it is returned. None of identifiers returned is equal to any previously returned identifier. The prototype is the resource identifier of an existing resource. The new resource identifiers will be managed by the same data provider as the prototype[6]. A null resource identifier may be provided, in which case the implementation will select a data provider. The integer how_many indicates the number of resource identifier values to be produced. The operation will return a sequence of at most this many resource identifier values. (If less than how_many are returned, then the client may make another request.)

6 Common services descriptions (normative)

This document normatively includes by reference the modelling contained in the Utility Management System (UMS) Data Access Facility (DAF) Resource Descriptions Module. This module defines a generic means to exchange information about CIM resources (power system related classes, class attributes, and object instances). The DAF Resource Descriptions Model defines several classes and one service as illustrated in Figure 3 and Figure 4 below:

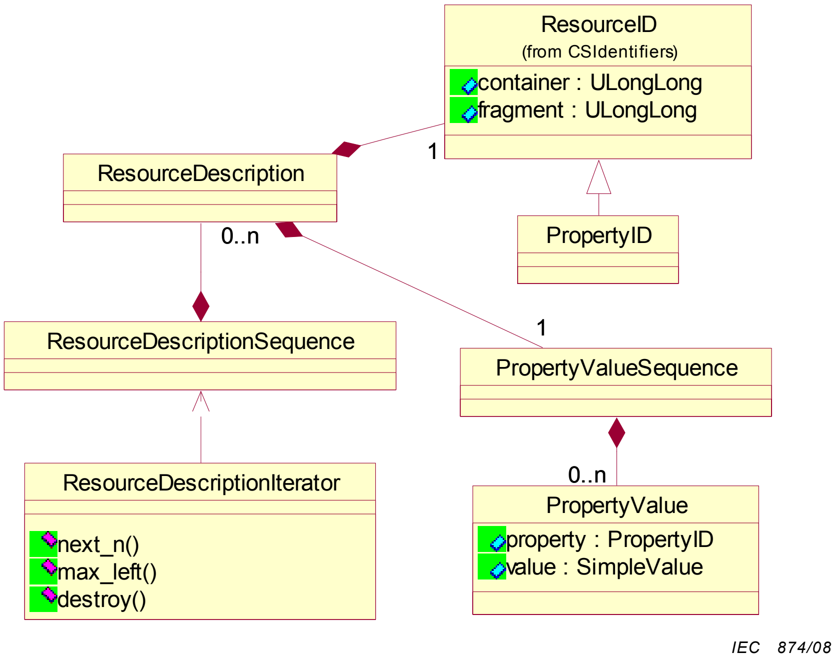

Figure 3 – Resource descriptions

[6] Note that the prototype parameter is only useful when an aggregating resource ID service provider is acting as a Proxy Server. For more information about Proxy Servers, see Annex B.

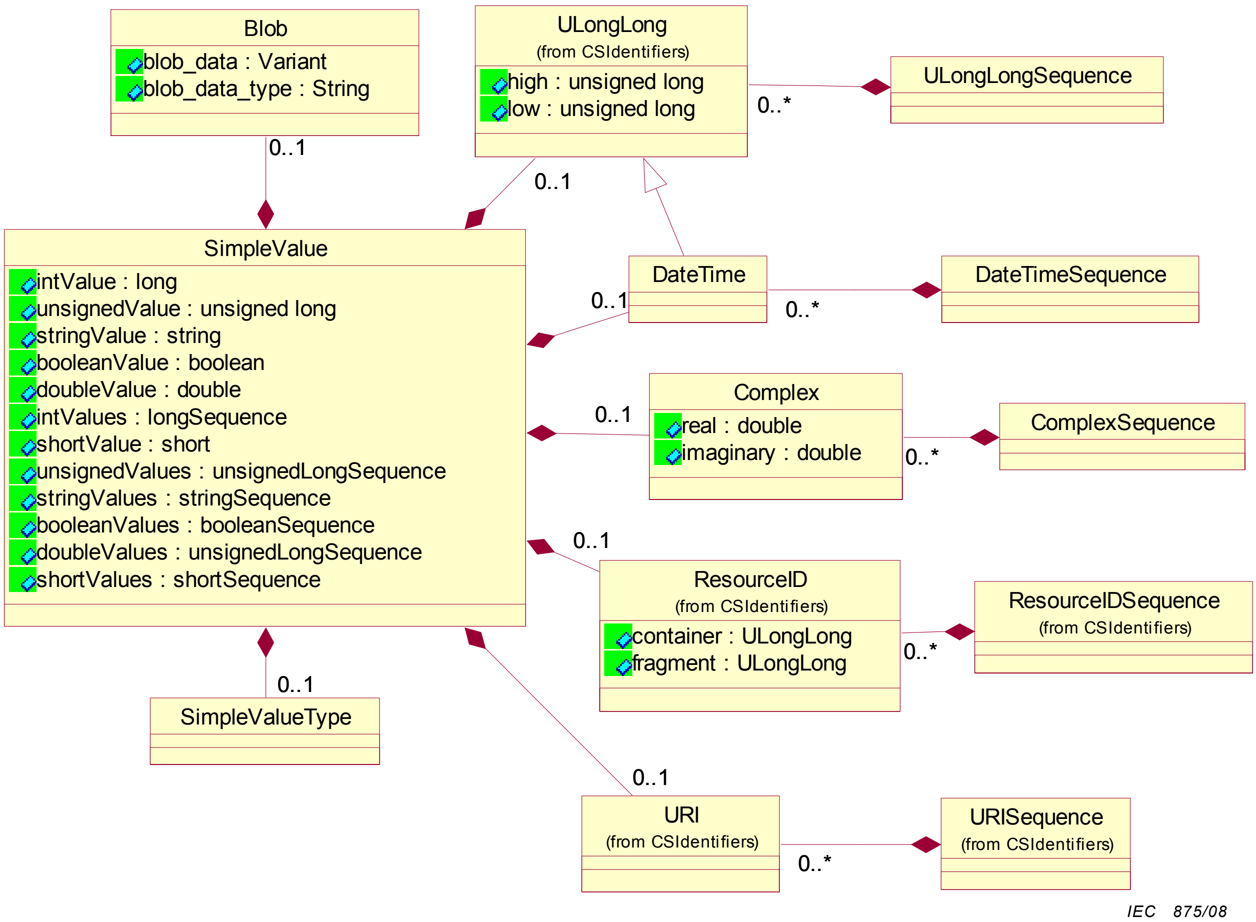

Figure 4 – Resource descriptions:SimpleValue

Table 3 – DAF resource description iterator

Operation	Example signature	Throws
Next_n	boolean next_n (unsigned long n, ResourceDescriptionSequence descriptions)	IteratorError
max_left	unsigned long max_left	IteratorError
destroy	void destroy	IteratorError

In Table 3, time is expressed by a 64 bit unsigned number (a ULongLong) which counts the number of 100 nanosecond units passed since 15 October 1582 00:00 UTC (the date of Gregorian reform to the Christian calendar). This convention is adopted from the CORBA Time Service, which in turn takes it from the *X/Open DCE Time Service*.

Complex numbers are used in a range of scientific applications and especially in power system analysis. In the DAF, complex numbers are expressed in Cartesian form.

SimpleValues are the object of properties; their role in the resource description framework was outlined in the previous clause. The DAF defines a SimpleValue as a discriminated union, where SimpleValueType is the discriminant. Fundamentally, a SimpleValue can be a resource identifier or a literal. The basic type of literal is a string to which the DAF adds explicit representation of the types defined above and (the largest variant of) each CORBA intrinsic type.

A property value specifies a property by its resource identifier and its associated value for a given resource. The types ResourceID and SimpleValue are defined in the DAFBasic module.

Each resource description identifies a single resource, by its resource identifier, and zero or more property values for that resource. The properties must be single valued; no provision is made for returning more than one value per property within a resource description. The class of the resource can be included in the resource description as the value of the *type* property (refer to the schema section for details).

Queries that return information about more than one resource return an iterator. The resource description iterator allows a client to access a large query result sequentially, several resources at a time. This is necessary to control the amount of information that gets passed to it at one time. It also enables implementations to overlap the client and server processing of query results, if necessary.

The client and the data provider should cooperate to manage the lifetime of the iterator and the resources it consumes. The destroy() and next_n() methods allow the client and data provider respectively to indicate that the iterator may be destroyed.

In addition, the data provider may autonomously destroy the iterator at any time (for resource management or other reasons). If a client detects that an iterator has been destroyed, it will not interpret this condition in itself as either an indication that the end of the iteration has been reached, or as a permanent failure of the data provider.

7 IECTC 57Views (normative)

7.1 General

This clause describes a standard way to present CIM resources in a non-opaque or well-known view[7].

7.2 IECTC 57View motivations (informative)

7.2.1 General

The standard interface description languages used by component execution systems such as Microsoft's Common Object Model (COM), OMG's Common Object Request Broker Architecture (CORBA), or Web Services provide mechanisms by which components can announce interface information about the public properties and methods they supply and consume. Using these technologies, a component publishes its interface metadata. Components that wish to attach to the system at large can discover interface metadata to determine how they might interact with system components. Interface announcement and discovery at this level provides a key piece in the construction of an interoperable infrastructure.

However, the use of component interface metadata alone fails to address one important element required to more closely attain interoperability. The missing element unsupported by the off the shelf use of the above component execution system technologies is instance object metadata. While it is useful to know component interface type information, it is also useful to know what specific instances of objects are being supplied by which components as well as schema associated with these objects. Announcement and discovery of individual instance object metadata facilitates a greater degree of "interoperability especially when combined with a common information model. When presented a well-known view of available object instances, components can programmatically discover what data is available, and automated attachment to the system at large is facilitated.

Given the potential limitations of interface description languages with regards to instance object metadata, an additional design construct must be introduced. IEC 61970-403: Generic

[7] By contrast, GES, HSDA and TSDA servers have an opaque view; that is, a GES, HSDA or TSDA server presents data with few fixed requirements as to the "names" or organization of the nodes and node properties. For the most part the "names" and organization are determined by the server. This is called an opaque view. A non-opaque or well known view on the other hand specifies rules for how nodes in a view appear.

Data Access allows a client to browse instance data, however some severs may only expose an HSDA, TSDA, or GES interface. In this case, the IECTC 57Views provide the capability to browse instance data in a standard way.

7.2.2 Problem example (informative)

The CIM consists of a full mesh network of information with multiple N to N relationships between objects. As a result, it is difficult to present a populated CIM model to the user in a simple Graphical User Interface (GUI). On the other hand, hierarchical views are readily understandable. For example, consider the fictitious information model illustrated in Figure 5:

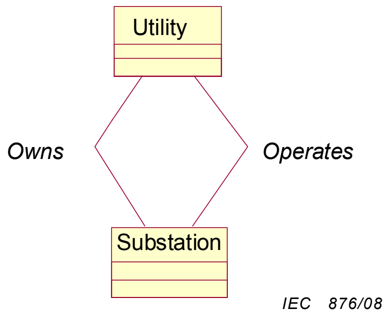

Figure 5 – Fictitious information model

Using this information model, a populated model can be displayed in a tree as illustrated in Figure 6:

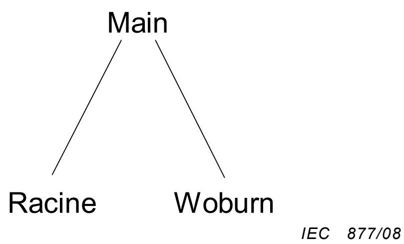

Figure 6 – Fictitious populated model displayed in tree

The view above consists of three nodes. The top node called "Main" is a branch node because it acts as the "parent" to the two "child" nodes below it. The bottom nodes are leaf nodes because there are no nodes below them. Assume we want to subscribe to data related to a substation owned by a utility via a hierarchical view of the CIM. However, from browsing the tree above several questions remain unanswered:

- Is the Main a utility or a substation?
- Assuming Main is a utility, is Racine owned or operated by Main?

Knowing the information model is not enough. In order to effectively display a populated model in a tree we also need to know:

- Type information for nodes
- Which associations are traversed in the tree. Standardized views allow applications browsing trees to answer these questions and in doing so enable automated processing.

7.2.3 IECTC 57View description (normative)

The IECTC 57Views provide an agreed upon way to present a populated CIM model via a hierarchy. An IECTC 57View specification includes:

- Agreement on which CIM classes appear
- Agreement on which CIM associations are traversed and in which direction.

An IECTC 57View consists of a hierarchy that contains a well-known top branch node name[8]. All well-known top branch node names have the prefix “IECTC 57”. " IECTC 57" is a reserved prefix and shall not be used for vendor or user-defined node names.

The well-known top branch node names are: IECTC 57PhysicalView and IECTC 57EventingView. All conforming implementations to the IEC 61970-402 specification support these nodes as specified by IEC 61970-403 to IEC 61970-449[9]. Example IECTC 57top branch nodes are illustrated in Figure 7[10].

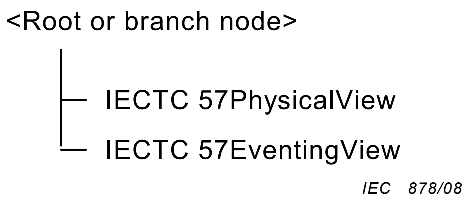

Figure 7 – IECTC 57Views

The IECTC 57PhysicalView contains a hierarchy of power system related instance data in accordance with how it is contained from a physical perspective. The IECTC 57EventingView contains a tree that is used to contain one or more hierarchies of nodes corresponding to information about what events (messages) a server publishes.

Below the top branch node, a conformant implementation includes a tree that is used to contain one or more hierarchies of nodes corresponding to classes, class attributes, and object instances defined in the IEC 61970-301 Common Information Model (CIM). IECTC 57Views may have any number of levels.

7.2.4 IECTC 57View examples (informative)

Figure 8 shows an example of an IECTC 57PhysicalView hierarchy that contains a node named Main (with a ResourceType property value of “Company”), which contains one sub-control area, named North (with a ResourceType property value of “SubControlArea”), which contains a substation named Airport (with a ResourceType property value of “Substation”). The substation contains two devices: Breaker 12 (with a ResourceType property value of “Breaker”) and Transformer 22 (with a ResourceType property value of “Transformer”). Each device is associated with a measurement (with a ResourceType property value of “Measurement”) and measurement value (with a ResourceType property value of “measurement value”). The hierarchy could contain any number of companies, sub-control areas, substations, devices, measurement and measurement value nodes.

---

8) Node names are called ItemID’s In OPC and are called labels in DAIS.

9) Note that a conforming application may present other non standard views in addition to those standardized in this document.

10) IECTC 57PhysicalView and IECTC 57EventingView top branch nodes can also appear at the root. In this case, there would be three independent trees corresponding to the three views.

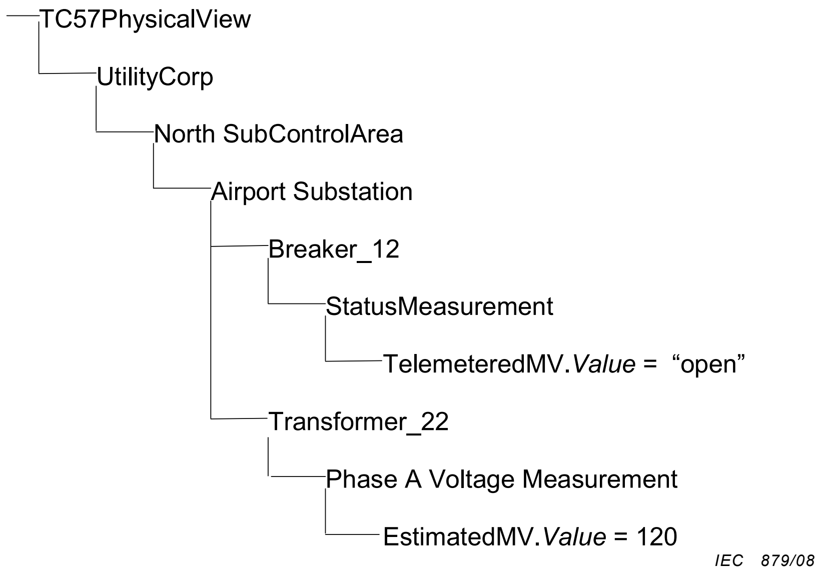

Figure 8 – Example TC 57 PhysicalView (informative)

In Figure 9, the IECTC 57EventingView example hierarchy contains three types of events: WorkOrder and its sub-classed event PreventativeMaintenanceWorkOrder as well as BreakerTestReport. Each event contains a collection of attributes. A WorkOrder Event contains WorkOrderID and Description attributes. A PreventativeMaintenanceWorkOrder contains WorkOrderID and Description inherited from its parent WorkOrder Event as well as AssetID and ProceedureID attributes.

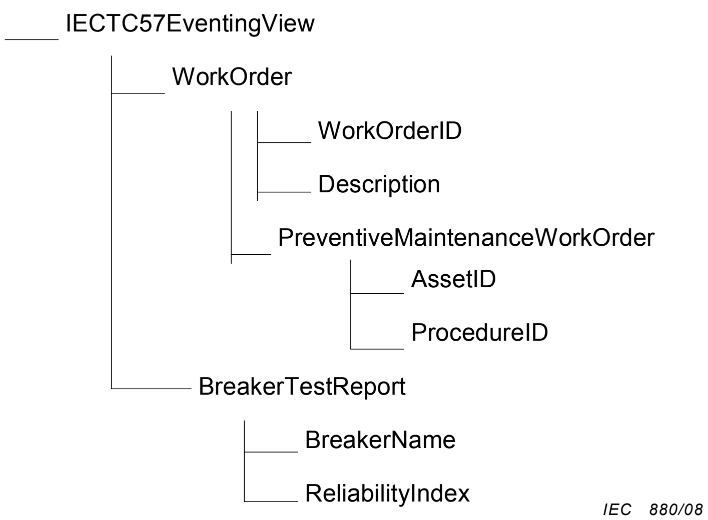

Figure 9 – Example IECTC 57EventingView (informative)

Particular resources or attributes referred to in an IECTC 57PhysicalView or IECTC 57EventingView may be simultaneously found in any of the other hierarchies. For example, "Breaker.Name" and "BreakerName" are found in the example IECTC 57EventingView. The particular Resource ID for the breaker name attribute is the same in both cases. In the IECTC 57EventingView, the fully qualified pathname is "IECTC 57EventingView/ BreakerTestReport/BreakerName". The term "fully qualified pathname" means that the value of the pathname is specific and unique enough to exactly identify the node within a specific hierarchy.

7.2.5 Node properties (normative)

7.2.5.1 General

An IECTC 57View contains nodes as well as node properties. While nodes are independently addressable, node properties can only be accessed via their parent node. Depending on the technology used to implement a server, the node properties discussed in this section may be exposed as standard or custom node properties. This section presents a technology neutral description of node properties required of a compliant view. However, how these node properties actually appear in a view is technology dependent[11].

A pathname permits a client to directly access properties for a node. Fully qualified pathname addressing may be used for any node in an IECTC 57View. Access to a node in a view may be accomplished either by starting at the Root and browsing down or directly by using a fully qualified pathname. When a server that exposes an IECTC 57View returns a pathname, it will always be a fully qualified pathname[12].

7.2.5.2 IECTC 57View versioning

Each instance of an IECTC 57View consists of a hierarchy that may change over time. Therefore, clients may need to browse the populated hierarchy to discover the nodes for each server they connect to. For example, a client may browse the IECTC 57EventingView to discover the list of the events a server publishes. Using this list of events, the client may then browse the attributes for the found event types.

Each IECTC 57top branch node includes a node property that specifies the version of the sub-tree. The value of the Version property changes as a result of metadata or instance data changes in the tree[13]. The format for the version node property shall be (see Table 4):

Table 4 – Format of view version

Version top branch node property	ASCII string containing four characters for the major version number and four characters for the minor version number. The major version number shall be separated from minor version number by a "." character. For example: "1234.1234"

7.3 The IECTC 57Physical view (normative)

7.3.1 General

The IECTC 57PhysicalView pathname is a tree that is used to contain one or more hierarchies of classes, class attributes, and object instances defined in IEC 61970-301 CIM. The hierarchies under the IECTC 57PhysicalView may have any number of levels. Each branch node under IECTC 57PhysicalView has a property called IECTC 57ResourceType whose value identifies the type (for example, company, area, substation, bay, breaker, measurement, or user defined types). Each node also has a property called ResourceID. This method provides for optionally expanding or collapsing the Physical View. This means that users have the capability to add levels (for example, a level for bays) or omit levels (for example, a level for sub-control areas) to the IECTC 57PhysicalView tree without losing its "discoverability". This feature is only intended to provide a means to communicate a model, no attempt is made to define or enforce the IEC 61970-301 CIM standard regarding the model structure.

---

11) See IEC 61970-404: High Speed Data Access Table 1 for more information about technology mapping.

12) See IEC 61970-404: High Speed Data Access for more information about pathname formats.

13) Exactly under what conditions a server increments the version number is not specified. However, it is recommended that servers increment version numbers at least whenever resource metadata changes occur.

7.3.2 IECTC 57Physical view hierarchy

7.3.2.1 General

An IECTC 57View contains a subset of CIM classes linked by a subset of CIM relationships. This section presents a technology neutral description of the particular classes and associations exposed in a compliant IECTC 57PhysicalView view.

An IECTC 57PhysicalView view shall traverse the following CIM classes along the following CIM class associations. Instances of CIM classes appearing higher in Table 5 appear higher in the IECTC 57PhysicalView tree. While all classes may not appear in the tree, the relative ordering of the classes must be preserved.

Table 5 – First level associations used in the TC 57PhysicalView

From association	Class	To association
	Company	Operates_PSRs
OperatedBy_Companies	SubControlArea	Contains_Substations
MemberOf_SubControlArea	Substation	Contains_VoltageLevels OR Contains_Equipment
MemberOf_Substation	VoltageLevel	Contains_Equipment OR Contains_Bays
MemberOf_VoltageLevels	Bay	Contains_Equipment

Table 6 lists equipment classes that can appear below a Substation, Bay, or VoltageLevel. However, Table 6 does not specify ordering of classes below a Substation or Bay. The last column indicates that any of the equipment classes may act as a parent for measurement class objects.

Table 6 – Second level associations used in the TC 57PhysicalView

From association	Class	To association
MemberOf_EquipmentContainer	Switch, Jumper, Fuse, Breaker, Disconnector,LoadBreak Switch, GroundDisconnect	Contains_Measurements
MemberOf_EquipmentContainer	PowerTransformer	Contains_Measurements
MemberOf_EquipmentContainer	Conductor, ACLineSegment, DCLineSegment	Contains_Measurements
MemberOf_EquipmentContainer	ProtectionEquipment, SynchrocheckRelay, CurrentRelay	Contains_Measurements
MemberOf_EquipmentContainer	HeatExchanger	Contains_Measurements
MemberOf_EquipmentContainer	RectifierInverter	Contains_Measurements
MemberOf_EquipmentContainer	RegulatingCondEq, StaticVar Compensator, Synchronous Machine, Compensator	Contains_Measurements
MemberOf_EquipmentContainer	Connector, BusbarSection,Junction	Contains_Measurements

Table 7 lists equipment classes of objects below a Power Transformer.

Table 7 – Third level associations used in the TC 57PhysicalView

From association	Class	To association
MemberOf_PowerTransformer	Transformer Winding	TapChangers
MemberOf_TransformerWinding	TapChanger	Contains_Measurements

Table 8 summarizes the equipment classes of objects that may act as parents to measurement objects.

Table 8 – Fourth level associations used in the TC 57PhysicalView

From class	Association
SubcontrolArea	Contains_Measurements
Substation	Contains_Measurements
VoltageLevel	Contains_Measurements
Bay	Contains_Measurements
Switch, Jumper, Fuse, Breaker, Disconnector, LoadBreakSwitch, GroundDisconnect	Contains_Measurements
Transformer	Contains_Measurements
TransformerWinding	Contains_Measurements
TapChanger	Contains_Measurements
Conductor, ACLineSegment, DCLineSegment	Contains_Measurements
ProtectionEquipment, SynchrocheckRelay, CurrentRelay	Contains_Measurements
HeatExchanger	Contains_Measurements
Rectifier/Inverter	Contains_Measurements
RegulatingCondEq, StaticVarCompensator, Synchronous Machine, Compensator	Contains_Measurements
Connector, BusbarSection, Junction	Contains_Measurements

Table 9 lists classes of objects below a substation or bay, however, the table below does not specify ordering of classes below a substation or bay.

Table 9 – Fifth level associations used in the TC 57PhysicalView

From association	Class	To association
MemberOf_PSR	Measurement	Contains_MeasurementValues
MemberOf_Measurement	MeasurementValue	

7.3.2.2 IECTC 57PhysicalView properties

IECTC 57PhysicalView nodes have the following properties. See Table 10.

Table 10 – Format of TC 57PhysicalView properties

Resource ID	ASCII string representation of hex digits: {xxxxxxxx-xxxxxxxx-xxxxxxxx-xxxxxxxx}.
Resource type	ASCII string containing CIM Class Name for the node. For CIM class nodes, this is always set to RDFS:Class. For CIM class attribute nodes, this is always set to RDF:Property.
Type resource ID	ASCII string representation of the resource type associated with the Resource Type in hex digits: {xxxxxxxx-xxxxxxxx-xxxxxxxx-xxxxxxxx}. For CIM class nodes, this is always set to the resource ID associated with RDFS:Class. For CIM class attribute nodes, this is always set to the resource ID associated with RDF:Property.
Generic data access type	ASCII string containing simple value type name. Only used for class attribute nodes.

7.4 The IECTC 57Eventing View (normative)

7.4.1 General

As previously specified, one method a client may use to determine the events and event parameters produced by a server is to discover them via browsing of an IECTC 57EventingView tree. Announcement and discovery of application category event types is a required function in order to achieve a interoperable component architecture. Browsing of an IECTC 57EventingView tree may be used to discover events generated by non-power system specific components (for example, an ERP system).

The IECTC 57EventingView is a tree that is used to contain one or more hierarchies of nodes corresponding to information about what events (messages) a server publishes. The IECTC 57EventingView pathname is a tree that is used to contain one or more hierarchies of event classes that consist of arbitrary collections of attributes typically from the IEC 61970-301 object model. The hierarchies under the IECTC 57EventingView may have any number of levels. Event class nodes may have event class nodes or attribute nodes as children. Event class attributes are contained in leaf nodes within the IECTC 57EventingView branch. In the case of the Eventing View, the pathname itself identifies the type. IECTC 57EventingView hierarchies provide for expanding and collapsing the Eventing View. Users may expand the model by creating new event classes, (for example, a new type of work order event), or omit levels (for example, a specific event type defined in a CIS).

7.4.2 IECTC 57EventingView properties

Every event in the IECTC 57EventingView consists of a set of modelled and indexed attributes. Modelled event attributes are used to convey data modelled in the CIM. Indexed event attributes are used to convey data not modelled in the CIM. An indexed attribute is typically used to convey non-persistent parameters or results passed between applications. For example, indexed attributes may consist only of an array of unnamed values. That is, there may not be a URI accessible via the Resource ID Service associated with each indexed attribute.

Whether an event attribute is modelled or indexed is indicated by the value of its “Modelled” property. If an event attribute is modelled in the CIM then the value of its Modelled property is set to “True”. If an event attribute is not modelled in the CIM then the value of its Modelled property is set to “False”.

IECTC 57EventingView nodes have the following properties. See Table 11.

Table 11 – Format of TC 57EventingView properties

Resource ID	ASCII string representation of hex digits:{xxxxxxxx-xxxxxxxx-xxxxxxxx-xxxxxxxx}.
Generic data access type	ASCII string containing Simple Value type name. Only used for event attribute nodes.
Modelled	Indicates whether an event attribute is modelled in the CIM that can be mapped to a resource ID using the ResourceIDService. For attributes modelled in the CIM, the value of this property is the ASCII string “True”. For attributes not modelled in the CIM, the value of this property is the ASCII string “False”. Only used for event attribute nodes.

8 Context support (normative)

IEC 61970 compliant applications shall be capable of operating in any context without modification. Component support for multiple contexts (operational, planning, test, etc.) shall be done at the component level and not at the business object level. That is, context names shall not appear in any of the standard views presented by a component instance. Furthermore, the ID for any resource shall be the same regardless of context. Only a single context may appear in the view of a server or proxy server (described in Annex B, Proxies).

The goal of this Context Support protocol is to ensure that independently developed clients and servers can be combined to form a coherent system, isolated from other systems that may be operating in the same host or network. Such a coherent set of software is referred to as a context in IEC 61970-1 of this series of standards, *Guidelines and general requirements.*

Annex A
(informative)

TC 57 Views discussion

A.1 General

This standard specifies a programmatic interface that a compliant component or component adapter must implement. The interface allows servers to expose a populated information model such as the CIM (the power system class metadata specified in the information model as well as the related instances). The latter concept is embodied in the term “view”.

The CIM includes a complex set of inter-related metadata and related instance data. That is, the CIM contains a “mesh network” or “lattice” of nodes. Figure A.1 illustrates a non-CIM-compliant example of a full mesh network of nodes where object types (dark gray boxes) are realized by object instances (light gray boxes). In the figure, relationships are shown as lines between object types and instances. There is more than one path between any two nodes so it is difficult to say that there is a top or bottom. The display of all of an unpopulated (just object types) or populated (including object instances) CIM provides two example mesh networks, since many CIM classes have many associations to many different nodes.

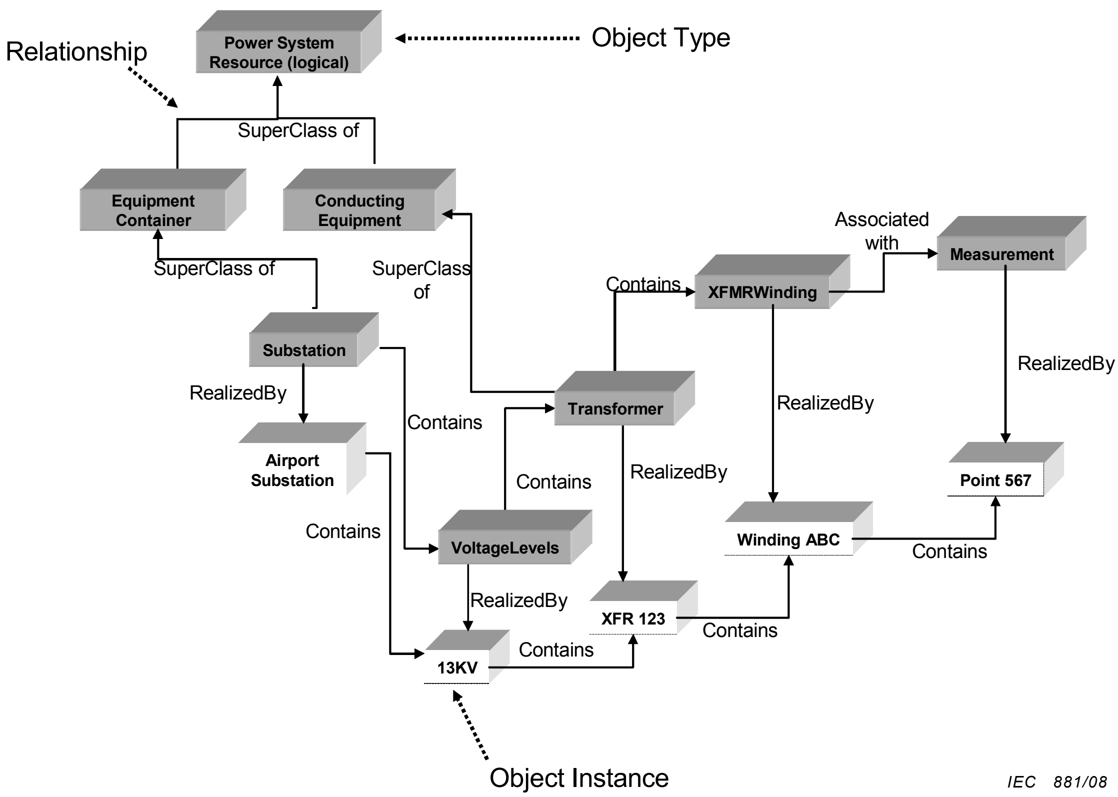

Figure A.1 – Non-CIM-compliant example of a full mesh network of object classes and instances

The IECTC 57standard views provide an agreement on how to communicate CIM-based hierarchies via an interface that relies on hierarchical browsing such as GES, TSDA, and HSDA. These hierarchies provide a simpler utility specific (CIM-based) way of viewing and configuring the exchange of data. The TC 57 Views provide a restricted means for exposing the CIM schema and instance data.

The TC 57 views provide help to an age-old utility problem that instance data is often not consistently or persistently named. For an example, consider Figure A.2 below:

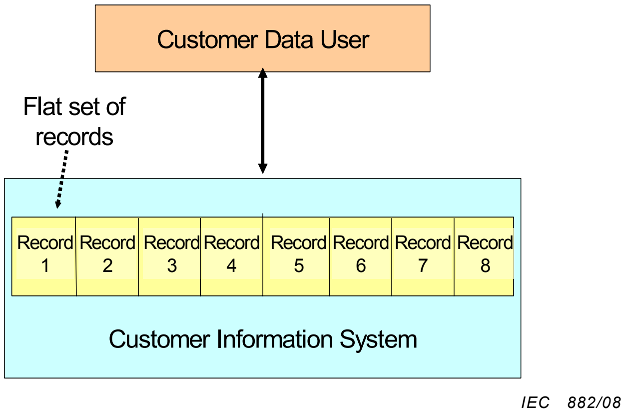

Figure A.2 – Non-CIM-compliant example of traditional view of utility data

Figure A.2 shows a non CIM compliant example of how data is often presented by a typical legacy application. In this case, customer data is presented by the application as a flat set of records without much information about how customers relate to information modelled in the CIM such as network topology or maintenance history. However, if one is interested in correlating the reliability of power delivered to customers to repair records or a network element such as a distribution feeder, then it is useful to be able to put customer records in the context of a model. The TC 57 Views provide a standard way of exposing data such as customer records within a known hierarchy. A non-CIM-compliant example of this concept is illustrated in Figure A.3.

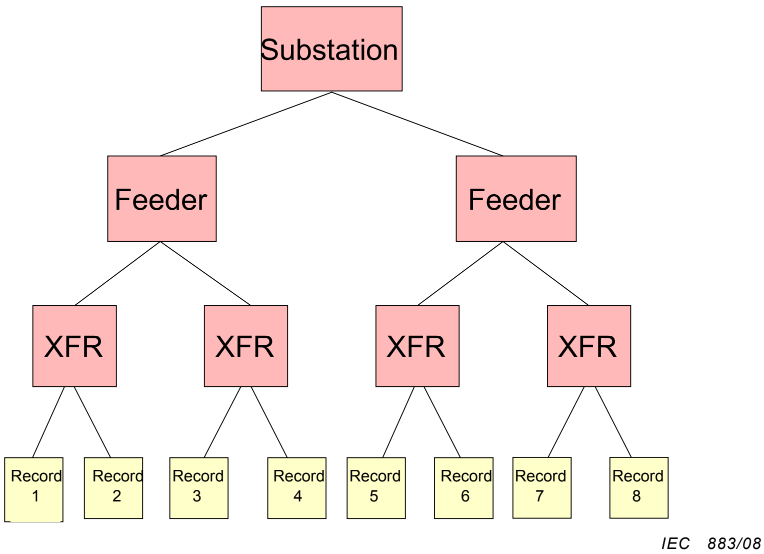

Figure A.3 – Non-CIM-compliant example of customer data within a network view

By providing agreed-upon paths to data, the TC 57 Views help engineers find the data they need. For example, the TC 57 PhysicalView orders power system related instance data in accordance to how it is contained from a physical perspective. The idea is that a power system engineer can find a breaker without having to remember a potentially convoluted or inconsistent naming scheme. The TC 57 EventingView view is associated with the Generic Eventing and Subscription (GES) interface, the IEC 61970 standard interface for publishing and subscribing. This tree allows an application to describe what event types (messages) it publishes as well as the content of each event type.

A.2 The Use of Views

The IEC 61970 interfaces provide the capability to discover the metadata and instance data a server exposes via browsing. However, browsing a server's complete full mesh view may consume a large amount of time. For example, consider a view deployed at a large utility. If one counts all the measurements in the view, the number might exceed one million. The number of possible paths to these measurements can be an order of magnitude more. The inefficiency associated with individually discovering each data measurement by browsing a full mesh view can significantly degrade the client user experience. Rather, it would be better if a client that wishes to subscribe to measurement data updates could determine the URI (paths from a hierarchical view root to destination) of each measurement in a more efficient way. But how can this be done when there is no standard way of naming each measurement point that indicates where a measurement occurs in a power system model? The solution is provided by the employment of a known common information model such as the CIM and a standard view such as the TC 57 PhysicalView view.

Consider the simple view illustrated in Figure A.4. In this example with regard to metadata, it is possible for a company to own or operate a device, and devices can have measurements associated with them. With regard to instance data, Eastern Electric is a company that owns TransformerABC that has a temperature measurement called Point 7.

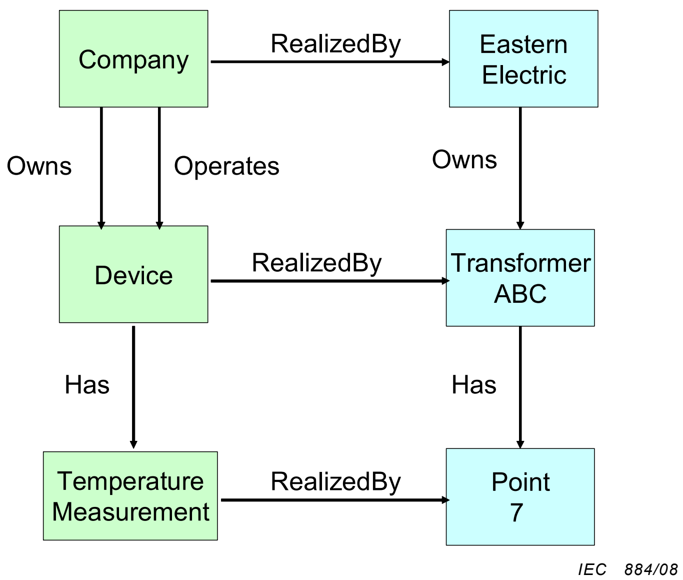

Figure A.4 – Non-CIM-compliant example of a full mesh view

From this full mesh view, at least two different hierarchical views could be constructed: one that specifies "company owns devices which have temperature measurements" and another that specifies "company operates devices which have temperature measurements". Since both are possible, it is impossible for an off-the-shelf client application to quickly discover the paths from root to every measurement for a large full mesh information model. On the other hand, if the client can assume that a server exposes a known hierarchy, say "company owns devices which have temperature measurements”, then the client can perform a very limited number of queries to discover all the paths to all measurements. In this very simple example, once configured with the company name, a client need only query twice. In the case of the CIM, agreement on the inclusion of only a limited set of associations that can be traversed can dramatically improve the user wait time for a client discovering measurements in a large view.

Another example of the use of the TC 57 Views consists of the possibility to completely automate SCADA data client subscription from a TC 57 Namepace compliant SCADA data server if the client application can import a portion of the power system model from a model provider such as an EMS or data engineering tool. In this case, a client has the capability to automatically build the paths to all the measurements in the SCADA Server and avoid browsing altogether. In fact, the server side of this use case has been tested during the EPRI-sponsored CIM/GID Interoperability Testing as illustrated in Figure A.5.

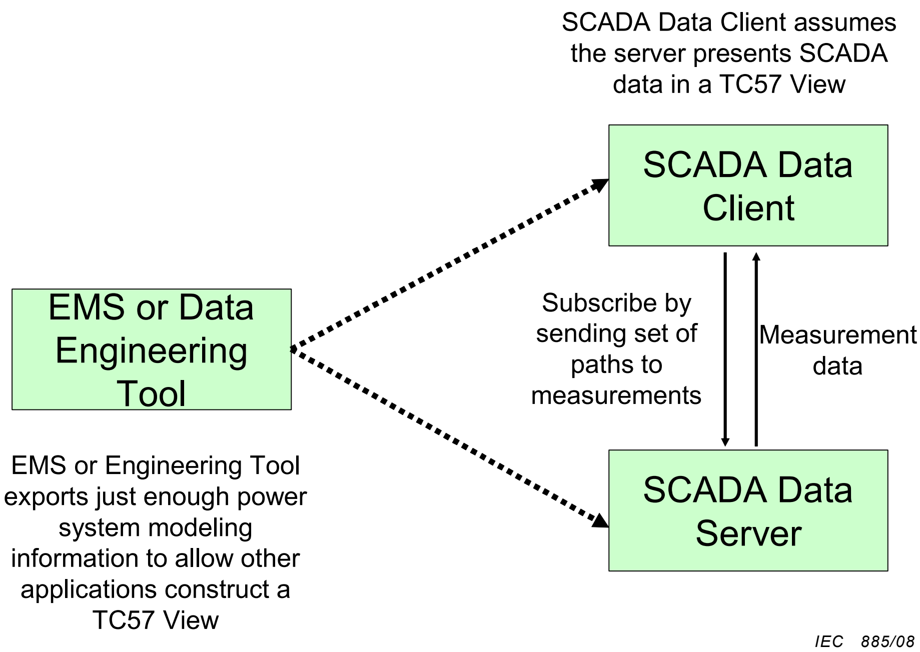

Figure A.5 – Subscribing to data using share path information

An additional benefit of providing customer data in a standard view is that off-the-shelf components can be created independently of individual customers’ requirements. Such standardization is necessary if one hopes to foster a market for off-the-shelf applications.

In addition to just providing a way for applications to find data, the TC 57 Views enhance subscribing to data. For example, the GES interface takes maximum advantage of CIM as presented in a TC 57 View. That is, GES provides a power system-oriented mechanism for data subscription configuration that power system engineers can use. The diagram below illustrates a sample TC 57 PhysicalView. The view would typically be displayed in a subscription configuration graphical user interface (GUI). By displaying a view of the power system model in a message subscription configuration GUI, the user can set up subscriptions without having to know the often complex subscription configuration script syntax used by a generic application integration tool. Off-the-shelf generic integration tools know nothing about power system models. By providing a power system specific user-friendly layer on top of the generic application integration tool, power system engineers can do things such as subscribe to a daily power quality report without requiring the assistance of an information technology professional.

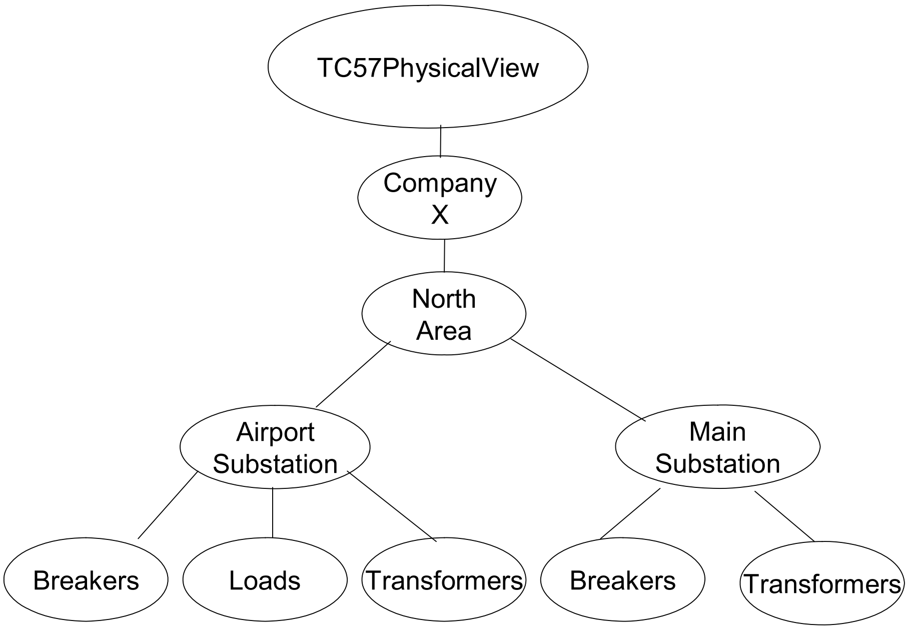

Figure A.6 – TC 57 View used as a subscription topic tree

Using the example view illustrated in Figure A.6, the user could subscribe to data related to just a particular substation, or set of devices in a substation just by specifying the path from company to the substation or devices in a substation – potentially all done via a user friendly GUI. CIM enabling subscription configuration makes data more accessible and "empowers the desktop"

Annex B
(informative)

Proxy architecture

B.1 Proxies

A proxy server is intended to enable multiple, independently developed servers of IEC 61970 Parts 402-449 API's to be combined within one context (for example operational, testing, planning contexts). In the simpler component configurations, no proxy is required. Multiple clients can share a single server that has a comprehensive schema and population. For example, the server might be a wrapper on an existing EMS.

However, a server that handles only part of the overall data needed by clients may be combined with others to form a complete system. This leads to a configuration in which multiple clients share multiple servers. It would arise when a IEC 61970-4xx interface is used to wrap individual servers in a control center rather than a single system such as an EMS, and the client may only need EMS data.

B.2 Rules for proxy servers

A proxy server presents a simple façade to the clients on the one hand, and combines the services of multiple servers on the other. At one extreme, very simple proxies could be envisaged. At the other, the proxy may actually be a wrapper for a more extensive application integration environment, an "integration bus" or a "data federation service".

In any case, the proxy server must provide the following capabilities.

- The proxy must hide the existence of multiple servers from the clients so that they can operate unchanged in both single-server contexts and multiple-server contexts.
- The proxy must encapsulate configuration information about which servers are present and what data they are responsible for supplying. Servers and clients must be insulated from the configuration details.
- A proxy must not require additional interfaces to be exposed either to clients or the ultimate servers. It acts as both a server and a client for the ResourceIDService interface defined in this specification.
- A proxy must present a unified view of all the data in a particular context to the clients, combining information provided by the ultimate servers. A query must return all the requested information that is available, even though it may be obtained from multiple servers.
- A proxy may implement various policies and optimisations to delegate queries, but these must be transparent to clients and the ultimate servers.

B.3 Proxy resource identification service

A proxy server must implement a ResourceIDService. Each translation request will be delegated to one or more of the ultimate servers and the results combined. The translation methods are designed so that each server can return results for those resources known to it without raising an exception for unknown resources.

Annex C
(informative)

The IEC 61970 services and mapping IEC 61968 verbs

C.1 General

IEC 61970 and IEC 61968 are complementary in that both have defined integration standards based on the use of common data semantics specified in the IEC Common Information Model (IEC 61970-3xx and IEC 61968-11).

As described in IEC 61968-1, IEC 61968 addresses application integration via the exchange of messages for the purposes of solving operational as opposed to analytic requirements. IEC 61968 defines a set of messages that get exchanged between applications to keep the applications synchronized and to avoid redundant data entry.

IEC 61970 takes a very complementary approach. However, instead of discussing standard messaging, IEC 61970-1 discusses standard services. While the IEC 61968 standard messages can be transferred via IEC 61970 services for the creation of an application integration infrastructure, additionally, the IEC 61970 standard services can be used for the creation of a data analysis infrastructure. IEC 61970 accomplishes the former by defining the Generic Eventing and Subscription service (GES) and the later by defining the Generic Data Access (GDA) service.

GES is a publish/subscribe service for synchronizing applications as part of an application integration process. GDA is a request/reply service for synchronizing application data and for allowing analysis applications to query for application data as part of a combined application integration and data warehousing/data mining process. This annex describes when to use GDA versus GES as well as a unified approach based on a mapping of IEC 61968 verbs[14] to IEC 61970 services. By sharing a common design framework, a message based application integration and data mining solution can be built simultaneously. This combined approach reuses shared application wrappers to leverage the investment in each without requiring all data to be copied to a data warehouse. Separately, the cost of developing individual wrappers for data warehousing and for application integration can be prohibitive.

C.2 Loosely coupled integration

Both IEC 61968 and IEC 61970 are focused on "loosely coupled" integration. IEC 61968 and IEC 61970 components do not have control over one another. Instead, systems such as Work Management and Geographic Information Systems may only make requests of each other via their shared knowledge of the CIM. Not allowing direct control limits the coupling between components by maximally treating each as black boxes where the internals of each are hidden. In this way, system management and system reconfiguration can be minimized in the face of changes to components.

To accomplish communication only via a shared knowledge of the CIM, IEC 61968 and IEC 61970 define a set of generic abstract verbs/services that are independent of how a component actually operates or where it resides. In other words, both sets of specifications seek to limit the component coupling by defining services that are data focused instead of command focused. To provide a concrete example, neither IEC 61968 nor IEC 61970 provide a

[14] See IEC 61968-1, *Application integration at electric utilities – System interfaces for distribution management – Part 1: Interface architecture and general requirements*

generic "Run" command. Instead they allow one component to request, change, or delete some part of the CIM maintained in another component.

C.3 Master and driven systems

As referred to above, one difference between IEC 61968 and IEC 61970 is that IEC 61968 has defined nouns (for example "Work Order") and verbs (for example "Create") for messages exchanged between systems (IEC 61968-1-10) while IEC 61970 has defined a set of technology neutral services for exchanging data (IEC 61970-4xx) and technology specific service mappings (IEC 61970-5xx). Both IEC 61968 verbs and IEC 61970 services are generic in nature in that they are independent of the application category they can be applied to as well as the data exchanged. This annex describes IEC 61968 verbs and IEC 61970 services, the system architecture that they both assume as well as how they can be mapped to one another.

IEC 61968 does not define any particular infrastructure but instead defines a set of verb (functions) and noun (message) pairs. Central to how IEC 61968 defines application exchange is a special application called a master system. A master system is one that is exclusively responsible for creating, deleting, as well as maintaining state for a particular set of CIM based objects. For example, a purchasing application is responsible for creating and deleting purchase orders and an EMS modelling tool is responsible for maintenance of logical device related resources associated with a transmission system. The compliment of a master system is a driven system. For example, a measurement archive acts as a driven system because much of its configuration can be driven by the EMS. Sometimes, an application will act as both a master and a driven system. For example, an AMS is responsible for maintaining asset records. However, an AMS may act as a driven system with respect to an EMS if the AMS maintains a mapping from physical to logical devices.

IEC 61968 defines a set of verbs to cover a complete data exchange model from a master system point of view and from a driven system point of view. Specifically, one set of present tense verbs are used by a driven system to request action by the master system and another set of past tense verbs are used by a master system for publishing when a set of CIM objects (resources) have been created, deleted, or changed. Present tense verbs used to act on master systems (the system of record for the given CIM resource) will typically result in a CIM-based document or resource to be created or updated in the master system if the request is processed successfully by the master system. Past tense verbs used to act on the driven systems will result in all referenced and/or replicated documents or resource to be updated.

Table C.1 lists the verbs for IEC 61968 and their mapping to IEC 61970 services. This list of verbs can be used to form the finite number of message types under the IEC 61968 standard. The following assumptions apply when using these verbs:

- For a given message document or its parts, there is usually one system that owns the creating, updating, and cancelling/deleting/closing of that document or one for each part. The system ownership could also be extended to the attribute level if necessary to allow for multiple systems updating a document in a workflow scenario.
- A message document has a life cycle in the integration systems and is identified by a unique resource ID across systems upon its creation or request for creation.

Lastly, it should be noted that verbs provide a short hand way of referring to GID services in process flow documents. At the time of actual component interface development and deployment, verbs can be replaced with the corresponding service. The next section describes the mapping between WG 14 verbs and WG 13 services.

C.4 Mapping the exchange of IEC 61968 messages to IEC 61970 generic data access

IEC 61968 accomplishes application integration via the exchange of messaging that contain documents. A document is a container for a collection of CIM class property instances. The contents of an IEC 61968 document can be seen as a view of part of the CIM specialized for a particular use case.

A document's life cycle is managed by a master system. When using IEC 61968 verbs, each create/change/delete can be considered a transaction. The document container is used to store transaction metadata such as when the transaction occurred, a description of the current state of the objects being acted upon, and comments related to the transaction.

IEC 61970 also accommodates transactions via the invocation of services. Using the component architecture described in IEC 61970-1, transaction metadata is handled as part of the interaction between a component and the component container. Thus, IEC 61968's document oriented approach solves similar issues as IEC 61970 service-oriented approach. In fact the only significant difference in this regard is the amount of transaction metadata this is exchanged. However, neither approach is always preferred. While many systems don't require as much transaction metadata as is provided by a IEC 61968 document exchange, but some do.

It is up to the system designer and or application vendor to decide if it is more appropriate to expose application data via the exchange of messaging (a GES interface) or via a request response interface (a GDA interface). They may decide to do one or another, or both. By exposing data via GDA, an application adapter for data analysis may be simultaneously created. One the other hand, data analysis may not be within the scope of the integration problem to be solved. The important point is that a well known mapping is defined so that a GES interface used to exchange IEC 61968 messages can be wrapped with GDA and visa versa. In this way even if GES/61968 message wrappers are created to address an application integration only deployment scenario, off the shelf GES/GDA gateways can be deployed to address a future data analysis deployment scenario.

C.5 Mapping IEC 61968 verbs to IEC 61970 services

Table C.1 gives the Mapping IEC 61968 verbs to IEC 61970 services.

Table C.1 – Mapping IEC 61968 verbs to IEC 61970 services

Verbs	Meaning	GID service used
CREATE	The CREATE verb is used to send a request to the master system to create a new document. The master system may in turn publish the new document using the verb CREATED. The master system may also use the verb REPLY to respond to the CREATE request, indicating whether the request has been processed successfully or not.	When a client wishes to create a new instance of an object in a master system, it submits a forward difference in a GDA Apply Updates invocation. As a result of receiving this forward difference, the Master system may create a new object. If a new object is created then the master system must publish a GDA model change event and also optionally publish a GES simple event containing the new object. Reply is handled via exception. That is, a reply is only sent by the master in the event that a GDA service invocation has failed. Note that when the master system creates something new, it will most likely need to invoke create_resource_ids to create a new resource ID for the new object being requested.
CHANGE	The CHANGE verb is used to send a request to the master system to make a change in the document based on the information in the message. The master system may in turn publish the changed document using the verb CHANGED to notify that the document has been changed since last published. The master system may also use the verb REPLY to respond to the CHANGE request, indicating whether the request has been processed successfully or not.	When a client wishes to change an instance of an object in a master system, it submits a forward difference in a GDA Apply Updates invocation. As a result of receiving this forward difference, the Master system may change an object. If the object is changed then the master system must publish a GDA model change event and also optionally publish a GES simple event containing the changed object. Reply is handled via exception. That is a reply is only sent by the master in the event that a GDA service invocation has failed. If the action is to be applied to MeasurementValue.value or MeasurementQuality, then HSDA is more typically used.

Table C.1 *(continued)*

Verbs	Meaning	GID service used
CANCEL	The CANCEL verb is used to send a request to the master system to cancel the document. The master system may in turn publish the cancelled message using the verb CANCELED to notify that the document has been cancelled since last published. The master system may also use the verb REPLY to respond to the CANCEL request, indicating whether the request has been processed successfully or not. The CANCEL verb is used when the business content of the document is no longer valid due to error(s).	The CANCEL verb is specific to when system state is maintained in a document. The status of a document may be changed using GDA. When a client wishes to cancel an instance of an document in a master system it submits a forward difference in a GDA Apply Updates invocation. As a result of receiving this forward difference, the Master system may change the status of the document. If the status of the document is changed to cancelled then the master system must publish a GDA model change event and also optionally publish a GES simple event containing the cancelled document. Reply is handled via exception. That is a reply is only sent by the master in the event that a GDA service invocation has failed.
CLOSE	The CLOSE verb is used to send a request to the master system to close the document. The master system may in turn publish the closed message using the verb CLOSED to notify that the document has been closed since last published. The master system may also use the verb REPLY to respond to the CLOSE request, indicating whether the request has been processed successfully or not. The CLOSE verb is used when the business document reaches the end of its life cycle due to successful completion of a business process.	The CLOSE verb is specific to when system state is maintained in a document. The status of a document may be changed using GDA. When a client wishes to close an instance of an document in a master system it submits a forward difference in a GDA Apply Updates invocation. As a result of receiving this forward difference, the Master system may change the status of the document. If the status of the document is changed to closed, then the master system must publish a GDA model change event and also optionally publish a GES simple event containing the closed document. Reply is handled via exception. That is a reply is only sent by the master in the event that a GDA service invocation has failed.
DELETE	The DELETE verb is used to send a request to the master system to delete the document. The master system may in turn publish the closed message using the verb DELETED to notify that the document has been deleted since last published. The master system may also use the verb REPLY to respond to the DELETE request, indicating whether the request has been processed successfully or not. The DELETE verb is used when the business document should no longer be kept in the integrated systems either due to error(s) or due to archiving needs.	When a client wishes to delete an instance of an object in a master system, it submits a reverse difference in a GDA Apply Updates invocation. As a result of receiving this reverse difference, the Master system may delete the object. If the object is deleted then the master system must publish a GDA model change event and also optionally publish a GES simple event containing the deleted object. Reply is handled via exception. That is a reply is only sent by the master in the event that a GDA service invocation has failed.
GET	The GET verb is used to send a request to the master system to get the current data for a given document reference code or a set of documents. The master system may in turn publish the document using the SHOW verb, if the document is available, or use the verb REPLY to respond to the GET request, indicating that the document is not available.	When a client wishes to get the current state of an instance of an object or set of objects in a master system, it submits a GDA Query. The reply to a GDA Query contains the results fo the query. If the action is to be applied to MeasurementValue.value or MeasurementQuality, then a HSDA Read is more typically used.

Table C.1 *(continued)*

Verbs	Meaning	GID service used
CREATED	The CREATED verb is used to publish the creation of a document as a result of either an external request or an internal action within the master system of that document. If this is the first time that data for this document reference code has been published as the result of internal or external request, then it would use the same document reference as the CREATE message. This message type is usually subscribed by interested systems and could be used for mass updates. There is no need to reply to this message type.	After creating an object or set of objects, a master system publishes a message to notify driven systems. If the message sent includes all of the data related to a created document, then this verb maps to the publication of a GES Simple Event. However, GDA provides a mechanism where only a summary event is sent. Since the amount of data sent for a created document can be large, and create can occur frequently, publication of created messages can consume a great deal of the network bandwidth. Consequently, GDA provides this optimisation.
CHANGED	The CHANGED verb is used to publish the change of a document as a result of either an external request or an internal action within the master system of that document. This could be a generic change in the content of the document or a specific status change such as "approved", "issued", etc. This message type is usually subscribed by interested systems and could be used for mass updates. There is no need to reply to this message type.	After changing an object or set of objects, a master system publishes a message to notify driven systems. If the message sent includes all of the data related to a changed document, then this verb maps to the publication of a GES Simple Event. However, the GDA provides a mechanism where only a summary of the changes is sent. Since the amount of data sent for a changed document can be large, and changes occur frequently, publication of change messages can consume a great deal of the network bandwidth. Consequently, GDA provides this optimisation. If the action is to be applied to MeasurementValue.value or MeasurementQuality, then HSDA is more typically used.
CLOSED	The CLOSED verb is used to publish the normal closure of a document as a result of either an external request or an internal action within the master system of that document. This message type is usually subscribed by interested systems and could be used for mass updates. There is no need to reply to this message type.	After closing a document, a master system publishes a message to notify driven systems. If the message sent includes all of the data related to a closed document, then this verb maps to the publication of a GES Simple Event. However, GDA provides a mechanism where only a summary event is sent. Since the amount of data sent for a closed document can be large, and closes can occur frequently, publication of change messages can consume a great deal of the network bandwidth. Consequently, GDA provides this optimisation.
CANCELED	The CANCELED verb is used to publish the cancellation of a document as a result of either an external request or an internal action within the master system of that document. This message type is usually subscribed by interested systems and could be used for mass updates. There is no need to reply to this message type.	After cancelling a document, a master system publishes a message to notify driven systems. If the message sent includes all of the data related to a cancelled document, then this verb maps to the publication of a GES Simple Event. However, GDA provides a mechanism where only a summary event is sent. Since the amount of data sent for a cancelled document can be large, and cancels can occur frequently, publication of cancel messages can consume a great deal of the network bandwidth. Consequently, GDA provides this optimisation.
DELETED	The DELETED verb is used to publish the deletion of a document as a result of either an external request or an internal action within the master system of that document. This message type is usually subscribed by interested systems and could be used for mass updates. There is no need to reply to this message type.	After deleting an object or set of objects, a master system publishes a message to notify driven systems. If the message sent includes all of the data related to a deleted document, then this verb maps to the publication of a GES Simple Event. However, GDA provides a mechanism where only a summary event is sent. Since the amount of data sent for a deleted document can be large, and closes can occur frequently, publication of change messages can consume a great deal of the network bandwidth. Consequently, GDA provides this optimisation.
SHOW	The SHOW verb is used to publish the most current content of a document as a result of either an external GET request or an internal action within the master system of that document. This message type is usually subscribed by the requesting system(s) or other interested systems. There is no need to reply to this message type.	Publication of a GES Simple Event. If the action is to be applied to MeasurementValue.value or MeasurementQuality, then HSDA is more typically used.

Table C.1 *(continued)*

Verbs	Meaning	GID service used
REPLY	The REPLY verb is used to publish the processing result of an external request to the master system to create, change, delete, cancel, or close a document. The REPLY message type could contain specific confirmation information as to whether the request is processed successfully or not and provide alternatives if applicable. This message type is usually subscribed by the requesting systems. There is no need to reply to this message type.	Reply maps to an exception thrown by a GDA Server or a query result. If the action is to be applied to MeasurementValue.value or MeasurementQuality, then the reply would more typically be from an HSDA provider.
SUBSCRIBE	The SUBSCRIBE verb is used to publish the request to ask the master system of a document to publish a CHANGED document whenever there is a change to the document. It implies that the master system will not publish the CHANGED document unless there are one or more subscribers for the changed information.	GES Subscription. If the action is to be applied to MeasurementValue.value or MeasurementQuality, then HSDA is more typically used.
UNSUBSCRIBE	The UNSUBSCRIBE verb is used to publish the request to ask the master system of a document to stop publishing a CHANGED document whenever there is a change to the document. It implies that the master system will not publish the CHANGED document when there are no subscribers.	GES Subscription. If the action is to be applied to MeasurementValue.value or MeasurementQuality, then HSDA is more typically used.

The mapping table above non-exclusively describes how IEC 61968 messages may be converted into IEC 61970 service invocations. For example, current MeasurementValue.Values's are required to be available from an HSDA server, but they may be available via any of the other interfaces. Interoperability is based on what information must be available from servers not what can be available from servers. For MeasurementValue.Value, a IEC 61968 GET maps to a HSDA Read, but a GDA query can be appropriate as well.

C.6 IEC 61968 verbs and IEC 61970 services versus commands

Analysis of the IEC 61970 services and the IEC 61968 verbs reveals the absence of a generic “Run” command. This has a profound affect on the design of services that a IEC 61970 or IEC 61968 component may expose. For example, a state estimator would not expose a service called “RunStateEstimator”, but instead only exposes a GDA request/response service that allows a client to request the creation of a new state estimator result document via a GDA apply_update service. In IEC 61970, the term "Generic" in GID means that no other methods beyond the GID are required for manipulation of CIM data in a loosely coupled environment.

The IEC 61968 “Create” verb maps to IEC 61970 GDA apply_updates - that is, a client is requesting the state estimator to create a new state estimator result document. Conceptually, a client requesting to create a new state estimator result document in a state estimator is equivalent to a client requesting to create a new substation in a data modeling tool.

IEC 61970 and IEC 61968 assume that a client can only make requests on a server that the server is free to ignore. In the case of a master system, a client can make a request, but the server is under no obligation to actually carry out that request. IEC 61970 GDA apply_updates returns success or failure that only indicates whether the apply_updates has been successfully received, but it is up to the server to decide whether to honor the request or not. Other clients find out about a new state estimator result document because the state estimator sends out a "state estimator result" created message.

Since all component interaction is only via a shared knowledge of the CIM, there is a very limited number of services or verbs to standardize. If one only considers data, then only a limited number of operations are possible: synchronous read/query, update, as well as publish and subscribe. Consequently, the data focused services of the GID only consist of GDA, GES, HSDA, and TSDA. IEC 61970 and IEC 61968 do not require additional standardization of a service for every component in the system. The approach of IEC 61970 and IEC 61968 is to essentially push the system specialization into the data layer. That is, adding an object to the CIM called "state estimate result" does not require the creating of new IEC 61970-4xx and IEC 61970-5xx series service and technology mapping specifications but only a relatively easy extension of the CIM (which may not necessarily need to be standardized). Furthermore, the GID provides a full set of services for discovery of the data model in use (typically the CIM with custom extensions). Component interfaces do not have to be rebuilt for every deployment scenario and CIM customization. In this way, component coupling is limited to a shared knowledge of the CIM while still providing complete discovery.

Note that from the point of view of a client overhead and the ability to create workflow scripts, there is no difference between the method called "run state estimator" and a generic method called "create" which takes "state estimator result" as a parameter. All scripting languages have the ability to pass parameters to a method. Passing a parameter to a generic method is not more difficult than calling a specialized method without any parameters. That is both IEC 61970 and IEC 61968 specify the use of generic methods instead of specialized methods.

Thus, it is the strict use of data driven generic services/verbs that ensures that the coupling between components is loose. The use of generic services allows service providers to be easily aggregated. Once aggregated, a consumer of information has no way of knowing what actual system component is acting as the master for any CIM object. A GID aggregator can in effect become a façade that decouples all components.

Bibliography

IEC 60060-50, *International Electrotechnical Vocabulary*

IEC 61850-7-2, *Communication networks and systems in substations – Part 7-2: Basic communication structure for substation and feeder equipment – Abstract communication service interface (ACSI)*

IEC 61968 (all parts), *Application integration at electric utilities – System interfaces for distribution management*

IEC 61968-1, *Application integration at electric utilities – System interfaces for distribution management – Part 1: Interface architecture and general requirements*

IEC 61968-3, *Application integration at electric utilities – System interfaces for distribution management – Part 3: Interface for network operations*

IEC 61968-11, *Application integration at electric utilities – System interfaces for distribution management – Part 11: Common Information Model (CIM) extensions for distribution*

IEC 61970-301, *Energy management system application program interface (EMS-API) – Part 301: Common Information Model (CIM) base*

OMG, Data Access From Industrial Systems, document dtc/01-07-03

OPC Foundation, OPC Data Access Custom Interface Specification

OPC Foundation, OPC Alarms and Events Custom Interface Specification

OPC Foundation, OPC Historical Data Access Custom Interface Specification

OPC Foundation, OPC XML Specification

OPC Foundation, OPC Batch Custom Interface Specification

Control Center Application Program Interface (CCAPI) Project: API Standard Proposal Requirements for Generic Interface Definition (GID), EPRI, Palo Alto, CA: 2001. 1001975.

Sun Microsystems, Java 2 Enterprise Edition, J2EE Connector Architecture Specification

Sun Microsystems, Java Messaging Service Specification

Open Applications Group, Open Applications Middleware API Specification

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

3, rue de Varembé
PO Box 131
CH-1211 Geneva 20
Switzerland

Tel: + 41 22 919 02 11
Fax: + 41 22 919 03 00
info@iec.ch
www.iec.ch