Draft IEC 61970: Energy Management System Application Program Interface (EMS-API) —

Part 301: Common Information Model (CIM) Base

Revision 028

Based on release startUmlAttribute.IEC61970CIMVersion.version.endUml

CONTENTS

FOREWORD 4

INTRODUCTION 6

1 Scope 8

2 Normative references 8

3 Terms and definitions 9

3.1 Terms and definitions overview 9

4 CIM specification 9

4.1 CIM modeling notation 9

4.2 CIM packages 9

4.3 CIM classes and relationships 10

4.3.1 Classes 10

4.3.2 Generalization 11

4.3.3 Simple association 12

4.3.4 Aggregation 12

4.4 CIM model concepts and examples 12

4.4.1 Concepts 12

4.4.2 Containment, equipment hierarchies and naming 12

4.4.3 Names model 13

4.4.4 Connectivity model 14

4.4.5 Inheritance hierarchy 16

4.4.6 Transformer model 16

4.4.7 Transformer tap modeling 17

4.4.8 Phase wire modeling 20

4.4.9 Cuts, clamps and jumpers model 21

4.4.10 Measurements and controls 23

4.4.11 Regulating control models 27

4.5 Modeling guidelines 27

4.5.1 Modeling for change 27

4.5.2 Process for amendments to the CIM 27

4.5.3 Changes to the CIM UML model 28

4.5.4 Changes to the CIM standards documents 28

4.5.5 CIM profiles 28

4.6 Modeling tools 28

4.7 User implementation conventions 29

4.7.1 Conventions beyond UML 29

4.7.2 Number of Terminals for ConductingEquipment objects 29

4.8 CIM modeling examples 29

5 Detailed model 29

5.1 Overview 29

5.2 Context 29

6 Package architecture (normative) 31

6.1 startUmlPackage.IEC61970.endUml 31

Bibliography 33

Figure 1 Working group package dependencies 10

Figure 2 – CIM IEC 61970-301 package diagram 10

Figure 3 – Example of generalization 12

Figure 4 – Example of simple association 12

Figure 5 – Example of aggregation 12

Figure 6 – Equipment containers 13

Figure 7 - Names 14

Figure 8 – Connectivity model 14

Figure 9 – Simple network example 15

Figure 10 – Simple network connectivity modeled with CIM Topology 15

Figure 11 – Equipment inheritance hierarchy 16

Figure 12 - Transformer and Tank model 17

Figure 13 – Transformer model 17

Figure 20 – Navigating from PSR to MeasurementValue 24

Figure 21 – Measurement placement 27

Figure 22 – Regulating control models 27

Figure 23 – CIM top level packages 30

Table 1 – NameType class naming conventions 13

Table 2 – measurementType naming conventions 25

Table 3 – MeasurementValueSource naming conventions 26

Attributes 30

Association ends 30

Enums 30

INTERNATIONAL ELECTROTECHNICAL COMMISSION

\_\_\_\_\_\_\_\_\_\_\_\_\_

ENERGY MANAGEMENT SYSTEM APPLICATION

PROGRAM INTERFACE (EMS-API) –

Part 301: Common information model (CIM) base

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 itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.

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.

International Standard IEC 61970-301 has been prepared by IEC technical committee 57: Power systems management and associated information exchange.

This fourth edition cancels and replaces the third edition, published in 2010. This fourth edition constitutes a technical revision.

Major changes from the fourth edition include the following.

* Transformer models modified to be consistent for use by distribution and transmission purposes. Additionally the tap changer model was updated to more clearly reflect the intended usage without relying upon rules for which attributes are appropriate in which situations.
* A more general and clear naming approach was added and several ambiguous attributes related to naming were dropped. The approach allows for users to define new name domains give them their own unique description.
* Dynamics models were added to support time domain simulations such as transient stability. This is incorporation from a prior effort sponsored by EPRI.
* Phase component wires models enhanced to describe internal phase specific attributes and connections. WG 14 was instrumental in these additions and cleanly integrating these into the shared Wires package.
* New datatypes added for Decimal, and cleanup of date and time types.
* New Compound datatypes added to the Domain package.
* Various editorial changes to cleanup UML model.

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

|  |  |
| --- | --- |
| FDIS | Report on voting |
| 57/XX/FDIS | 57/XX/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.

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

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.

The committee has decided that the contents of this publication will remain unchanged until the maintenance result date[[1]](#footnote-1)) 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 which define an application program interface (API) for an energy management system (EMS). This standard was originally based upon the work of the EPRI Control Center API (CCAPI) research project (RP-3654-1). The principle objectives of the EPRI CCAPI project were to:

* reduce the cost and time needed to add new applications to an EMS;
* protect the investment of existing applications or systems that are working effectively with an EMS.

The principal objective of the IEC 61970 series of standards is to produce standards which facilitate the integration of EMS applications developed independently by different vendors, between entire EMS systems developed independently, or between an EMS system and other systems concerned with different aspects of power system operations, such as generation or distribution management systems (DMS). This is accomplished by defining application program interfaces to enable these applications or systems access to public data and exchange information independent of how such information is represented internally.

The Common Information Model (CIM) specifies the semantics for this API. The Component Interface Specifications (CIS), which are contained in other parts of the IEC 61970 standards, specify the content of the messages exchanged.

The CIM is an abstract model that represents all the major objects in an electric utility enterprise typically needed to model the operational aspects of a utility. This model includes public classes and attributes for these objects, as well as the relationships between them.

The objects represented in the CIM are abstract in nature and may be used in a wide variety of applications. The use of the CIM goes far beyond its application in an EMS. This standard should be understood as a tool to enable integration in any domain where a common power system model is needed to facilitate interoperability and plug compatibility between applications and systems independent of any particular implementation.

This part of the standard, IEC 61970-301, defines the CIM Base set of packages which provide a logical view of the functional aspects of an Energy Management System including SCADA. Other functional areas are standardized in separate IEC documents that augment and reference this base CIM standard. For example, IEC 61968-11 addresses distribution models and references this base CIM standard. While there are multiple IEC standards dealing with different parts of the CIM, there is a single, unified information model comprising the CIM behind all these individual standards documents.

The International Electrotechnical Commission (IEC) draws attention to the fact that it is claimed that compliance with this document may involve the use of a patent concerning a computer-based implementation of an object-oriented power system model in a relational database. As such, it does not conflict with the development of any logical power system model including the Common Information Model (CIM), where implementation of the model is not defined.

The IEC takes no position concerning the evidence, validity and scope of this patent right.

The holder of this patent right has assured the IEC that he/she is willing to negotiate licences either free of charge or under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with IEC. Information may be obtained from:

ICL

Wenlock Way

West Gorton

Manchester

M12 5DR

United Kingdom (U.K.)

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified above. IEC shall not be held responsible for identifying any or all such patent rights.

ISO ([www.iso.org/patents](http://www.iso.org/patents)) and IEC (<http://www.iec.ch/tctools/patent_decl.htm>) maintain on-line data bases of patents relevant to their standards. Users are encouraged to consult the data bases for the most up to date information concerning patents.

ENERGY MANAGEMENT SYSTEM APPLICATION

PROGRAM INTERFACE (EMS-API) –

Part 301: Common information model (CIM) base

# Scope

The common information model (CIM) is an abstract model that represents all the major objects in an electric utility enterprise typically involved in utility operations. By providing a standard way of representing power system resources as object classes and attributes, along with their relationships, the CIM facilitates the integration of Energy Management System (EMS) applications developed independently by different vendors, between entire EMS systems developed independently, or between an EMS system and other systems concerned with different aspects of power system operations, such as generation or distribution management. SCADA is modeled to the extent necessary to support power system simulation and inter-control center communication. The CIM facilitates integration by defining a common language (i.e. semantics) based on the CIM to enable these applications or systems to access public data and exchange information independent of how such information is represented internally.

The object classes represented in the CIM are abstract in nature and may be used in a wide variety of applications. The use of the CIM goes far beyond its application in an EMS. This standard should be understood as a tool to enable integration in any domain where a common power system model is needed to facilitate interoperability and plug compatibility between applications and systems independent of any particular implementation.

Due to the size of the complete CIM, the object classes contained in the CIM are grouped into a number of logical Packages, each of which represents a certain part of the overall power system being modeled. Collections of these Packages are progressed as separate International Standards. This particular International Standard specifies a Base set of packages which provide a logical view of the functional aspects of Energy Management System (EMS) information within the electric utility enterprise that is shared between all applications. Other standards specify more specific parts of the model that are needed by only certain applications. Subclause 4.2 below provides the current grouping of packages into standards documents.

# 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 60050 series, *International Electrotechnical Vocabulary*

IEC 61850-7-4, *Basic communication structure – Compatible logical node classes and data object classes, , First edition, 2003-05*

IEC 61850 (all parts), *Communication Networks and Systems in Substations*

IEC 61968 (all parts), *Standards for distribution management system interfaces*

IEC 61970-2, *Energy Management System Application Program Interface (EMS-API) – Glossary*

IEC 60870-6 (all parts), *Telecontrol equipment and systems – Telecontrol protocols compatible with ISO standards and ITU-T recommendations*

Object Management Group: UML 2.0 Specification – http://www.omg.org

# Terms and definitions

## Terms and definitions overview

For the purposes of this document, the terms and definitions of IEC 61970-2 apply, as well as the following.

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

energy management system   
EMS

computer system comprising a software platform providing basic support services and a set of applications providing the functionality needed for the effective operation of electrical generation and transmission facilities so as to assure adequate security of energy supply at minimum cost

application program interface   
API

the set of public functions provided by an executable application component for use by other executable application components

unified modeling language   
UML

formal and comprehensive descriptive language with diagramming techniques used to represent software systems, from requirements analysis, through design and implementation, to documentation

UML has evolved from a collection of methods contributed by different practitioners, into an Inernational Standard. The CIM relies on UML for defining the model, and automated tools generate the documentation, schemas, and other artifacts directly from the UML. A basic understanding of UML is necessary to understand the CIM.

profile

subset of DCIM classes, associations and attributes needed to accomplish a specific type of interface

It may be expressed in XSD, RDF, and/or OWL files. A profile can be tested between applications. A profile is necessary in order to “use” the DCIM. Several profiles are defined in other parts of the IEC 61968 family of standards.

# CIM specification

## CIM modeling notation

The CIM is defined using object-oriented modeling techniques. Specifically, the CIM specification uses the Unified Modeling Language (UML) notation, which defines the CIM as a group of packages.

Each package in the CIM contains one or more class diagrams showing graphically all the classes in that package and their relationships. Each class is then defined in text in terms of its attributes and relationships to other classes.

The UML notation is described in Object Management Group (OMG) documents and several published textbooks.

## CIM packages

The CIM is partitioned into a set of packages. A package is a general purpose means of grouping related model elements.The packages have been chosen to make the model easier to design, understand and review. The Common Information Model consists of the complete set of packages. Entities may have associations that cross many package boundaries. Each application will use information represented in several packages.

The comprehensive CIM is partitioned into groups of packages for convenience in managing and maintaining them. These groups are assigned to different working groups within TC57.

WG13 packages are included in IEC 61970-301 (this document) and contained under the IEC61970 package. The IEC61970 packages as a set are self contained and do not depend upon other outside packages. As shown in Figure 1 Working group package dependencies, the IEC61970 package and its subpackages are used as a core or base model for other CIM packages. The dashed lines indicate dependency relationships, with the arrowhead pointing from the dependent package to the package on which it has a dependency. WG14 packages in the IEC 61968 series of standards describe additional parts of the CIM that deal with other logical views of utility operations including include assets, locations, activities, consumers, documentation, work management, and metering. WG16 packages in the IEC 62325 series of standards describe electric energy markets.

startUmlDiagram.PackageDependencies.PackageDependencies.endUml

Figure 1 Working group package dependencies

Note that the package boundaries do not imply application boundaries. An application may use CIM entities from several packages. It is also anticipated CIM packages outside of this document will have dependencies upon some of the packages described in this document, and particularly the Domain and Core packages, though other dependencies will also exist.

Figure 2 shows the packages defined for IEC 61970-301 CIM Base and their dependency relationships.

startUmlDiagram.IEC61970.IEC61970Dependencies.endUml

Figure 2 – CIM IEC 61970-301 package diagram

Clause 6 contains the specification for each of the CIM packages.

NOTE  The contents of the CIM defined in this specification were auto-generated from the CIM UML electronic model release startUmlAttribute.IEC61970CIMVersion.version.endUml, which is available through the CIM Users Group.

## CIM classes and relationships

### Classes

The class diagram(s) for each CIM package shows all the classes in the package and their relationships. Where relationships exist with classes in other packages, those classes are also shown.

Classes and objects model what is in a power system that needs to be represented in a common way to EMS applications. A class is a description of an object found in the real world, such as a power transformer, generator, or load that needs to be represented as part of the overall power system model in an EMS. Other types of objects include things such as schedules and measurements that EMS applications also need to process, analyze, and store. Such objects need a common representation to achieve the purposes of the EMS-API standard for plug-compatibility and interoperability. A particular object in a power system with a unique identity is modeled as an instance of the class to which it belongs.

It should also be noted that the CIM is defined to facilitate data exchange. As defined in this document, CIM entities have no behavior. For a specific interface, a profile is defined consisting of a subsets of CIM classes, attributes, and associations. A profile defines the message payload for an interface.

Classes have attributes that describe the characteristics of the objects. Each class in the CIM contains the attributes that describe and identify a specific instance of the class. Only the attributes that are of public interest to EMS applications are included in the class descriptions.

Each attribute has a type, which identifies what kind of attribute it is. Typical attributes are of type integer, float, boolean, string, date, and decimal, which are called primitive types. However, many additional types are defined as part of the CIM specification. For example, ShuntCompensator has a "maxU" attribute of type Voltage. The definition of many shared types is contained in the Domain Package described in Clause 6. The UML stereotypes of Primitive, enumeration, CIMDatatype, and Compound are added to classes used as types. The CIMDatatype stereotype is used with a specific CIM semantics for a triple of attributes {value, unit, multiplier}, which implies custom mapping to serialization artifacts such as RDFS, OWL, and XSD. Classes with these stereotypes do not participate in generalization or association relationships and are simply used as types for attributes. The enumeration stereotype is used to describe an attribute with an enumerated list of choices. The Compound stereotype is used to describe sets of related attributes that are commonly reused. Compound classes may consist of Primitive, enumeration, CIMDatatype or other Compound classes as long as the Compound classes do not recurse.

All CIM attributes are implicitly optional in the sense that profiles using the CIM may eliminate any attributes.

Relationships between classes reveal how they are structured in terms of each other. CIM classes are related in a variety of ways, as described in the subclauses below.

### Generalization

A generalization is a relationship between a more general and a more specific class. The more specific class can contain only additional information. For example, a PowerTransformer is a specific type of ConductingEquipment. Generalization provides for the specific class to inherit attributes and relationships from all the more general classes above it.

Figure 3 is an example of generalization. In this example taken from the Wires package, a Breaker is a more specific type of ProtectedSwitch, which in turn is a more specific type of Switch, which is a more specific type of ConductingEquipment, etc. A PowerTransformer is another more specific type of Equipment. Note that PowerSystemResource inherits from class IdentifiedObject which is not on the diagram so IdentifiedObject is shown in italic type in the upper right corner of class PowerSystemResource.

startUmlDiagram.DocIEC61970.DocumentationExampleInheritance.endUml

Figure 3 – Example of generalization

### Simple association

An association is a conceptual connection between classes. Each association has two “association ends”. The “association ends” were called “roles” prior to the UML 2.0 specification. Each association end describes the role the target class (i.e., the class the association end goes *to*) has in relation to the source class (i.e., the class the association end goes *from*). Association ends are usually given the name of the target class with or without a verb phrase. Each association end also has multiplicity/cardinality, which is an indication of how many objects may participate in the given relationship. In the CIM, associations are not named, only association ends are named. For example, in the CIM there is an association between a BaseVoltage and a VoltageLevel (See Figure 4 which is taken from the Wires package). Multiplicity is shown at both ends of the association. In this example, a VoltageLevel object may reference 1 BaseVoltage, and a BaseVoltage may be referenced by 0 or more VoltageLevel objects.

startUmlDiagram.DocIEC61970.DocumentationExampleAssociation.endUml

Figure 4 – Example of simple association

### Aggregation

Aggregation is a special case of association. Aggregation indicates that the relationship between the classes is some sort of whole-part relationship, where the whole class “consists of” or “contains” the part class, and the part class is “part of” the whole class. The part class does not inherit from the whole class as in generalization. Figure 5 illustrates an aggregation between the EquipmentContainer class and the Equipment class, which is taken from the Core package. As shown, an Equipment can be a member of zero or one EquipmentContainer objects, but an EquipmentContainer object can contain any number of Equipment objects. In the context of using CIM as an information model, aggregation does not have a precise or formal interpretation beyond a simple association and is intended to visually assist in representing normal usage.

startUmlDiagram.DocIEC61970.DocumentationExampleAggregation.endUml

Figure 5 – Example of aggregation

## CIM model concepts and examples

### Concepts

The CIM classes, attributes, types, and relationships are specified in Clauses 5 and 6. Clause 6 comprises a complete description of the IEC 61970-301 CIM Base model. To help understand how to interpret the CIM, some key model concepts used in the CIM are introduced and described in the following subclauses.

### Containment, equipment hierarchies and naming

#### Containment structure

Figure 5 shows the concept of equipment containers to form hierarchies in the CIM. Equipment containers represent ways of organizing and naming equipment typically found within a substation. As may be seen, there is some flexibility provided in which containers are used in a specific application of the CIM in order to accommodate different international practices as well as differences typically found between transmission and distribution substations. Bay, VoltageLevel, Substation, Line, and Plant are all types of EquipmentContainers. In general, a Bay is contained within a specific VoltageLevel, which in turn is contained within a Substation. Substations and Lines may be contained within SubGeographicRegions and GeographicRegions.

One containment hierarchy is used with the IdentifiedObject class to create hierarchical naming intended for human consumption. One hierarchy is specifically used to name equipment according to its function in the power system. This is the functional naming hierarchy. Other common identifications beside functional names are asset serial numbers. The functional name is different from a serial number in that it relates to the function of a particular equipment position or location in the power system. Regardless of what specific piece of physical equipment is placed at a location, the functional name is the same but the serial number varies depending on the physical equipment currently used.

#### IdentifiedObject class

The IdentifiedObject class contained in the Core package is inherited by all PowerSystemResource and many other classes. This class has attributes and associations to be used for naming all CIM objects. The mRID attribute of the IdentifiedObject class provides a straight forward and rigorous means of identity for CIM objects.

The following are definitions and conventions for how to use the IdentifiedObject attributes when naming PowerSystemResource objects (for more details, refer to documentation for IdentifiedObject and its attributes in Clause 6):

* mRID (Master Resource ID): A globally unique machine-readable identifier for an object instance.
* name: The name is any free human readable and possibly non unique text naming the object.
* aliasName: This attribute is deprecated and the Name classes of clause 4.4.3 provide a better alternative. The aliasName is free text human readable name of the object alternative to IdentifiedObject.name. It may be non unique and may not correlate to a naming hierarchy. The attribute aliasName is retained because of backwards compatibility between CIM relases. It is however recommended to replace aliasName with the Name class as aliasName is planned for retirement at a future time.

startUmlDiagram.Wires.NamingHierarchyPart1.endUml

Figure 6 – Equipment containers

The diagrams NamingHierarchyPart1 and NamingHierarchyPart2 in the Wires package show the functional naming hierarchy (refer to the Wires package documentation in Clause 6 for the details).

### Names model

A flexible and extensible naming model is provided to specify alternate names for objects. Figure 7 - Names shows the class diagram for alternate names and how they can be associated to a specific user defined name type. This model allows for specific definition of names within a particular domain and enforces no specific naming rules. The concepts of alias names, path names, and local names can all be accommodated by this model. The CIM models do not enforce or promote specific naming conventions, but allow for such naming conventions to be exchanged in a clear context.

startUmlDiagram.Core.Names.endUml

Table 1 – NameType class naming conventions

|  |  |
| --- | --- |
| name | description |
| ICCP | Reserved for ICCP (TASE-2) names. Used to describe ICCP point names for the MeasurementValue class and ICCP source names for the MeasurementValueSource class. |

Table 1 describes various name conventions for name types. The meaning of the columns in Table 1 are as follows:

name : The name attribute value.

description : The use of the name type.

Note that Table 1 is not an exhaustive list of values that can be used for the name attribute of the class Name.

Figure 7 - Names

### Connectivity model

#### Connectivity description

Figure 6 shows the Topology class diagram which models connectivity between different types of ConductingEquipment. Also included is a portion of the Meas package class diagram dealing with measurements to illustrate how measurements are associated with conducting equipment.

startUmlDiagram.Topology.Main.endUml

Figure 8 – Connectivity model

To model connectivity, Terminal and ConnectivityNode classes are defined. A Terminal belongs to one ConductingEquipment, although ConductingEquipment may have any number of Terminals. Each Terminal may be connected to a ConnectivityNode, which is a point where terminals of conducting equipment are connected together with zero impedance. A ConnectivityNode may have any number of terminals connected, and may be a member of a TopologicalNode (i.e., a bus), which is in turn a member of a TopologicalIsland. TopologicalNodes and TopologicalIslands are created as a result of a topology processor evaluating the “as built” topology and the actual Switch positions.

It is possible to exchange messages directly involving TopologicalNode and bypassing the ConnectivityNode details by using the TopologicalNode to Terminal association. This is often useful for exchanges involving bus/branch models which do not typically contain switch detail. When the TopologicalNode model is exchanges it is useful to also exchange the “connected” attribute on the Terminal class. This allows for the topological structure of the bus/branch model to be retained, yet indicate a terminal is not completely connected. Normally this disconnection is the result of one or more switches opening in the detailed model which is not represented in a bus/branch model exchange.

EquipmentContainers, which are a specialization of a ConnectivityNodeContainer, may contain zero or more ConnectivityNodes. The associations, ConductingEquipment – Terminal and Terminal – ConnectivityNode, capture the as-built topology of an actual power system network. For each Terminal connected to a ConnectivityNode, the associations of the other Terminal(s) connected to the same ConectivityNode identify the ConductingEqupment object(s) that are electrically connected. Similar connectivity at the “bus/branch” level of detail can be expressed using the TopologicalNode instead of the ConnectivityNode. The BusNameMarker class, which has an association to ConnectivityNode, is used to carry user bus names to be applied to TopologicalNodes. TopologicalNodes are created from ConnectivityNodes as a result of topology processing.

To model the analog values such as voltage and power, each Terminal has an association with a Measurement class from the Meas package. Although not shown in Figure 6, a Measurement object is associated with at least one MeasurementValue object. Each MeasurementValue object is an instance of a measurement from a specific source, for example, a telemetered measurement.

Clause 6 contains a complete description of each class in Figure 8 along with the definition of all the attributes and relationships supported in each class.

#### Connectivity and containment example

To illustrate how the connectivity model and containment model would appear as objects, a small example is presented in Figure 9. The example shows a transmission line with a T-junction spanning two substations and a substation having two voltage levels with a transformer between them. The transmission line consists of two different cables. One of the voltage levels is shown with a busbar section having a single busbar and two very simple switchgear bays connecting to the busbar.



Figure 9 – Simple network example

Figure 10 shows how connectivity is modeled in the CIM as well as one way (but not necessarily the only way) containment is modeled for the diagram in Figure 10. The shaded square boxes represent EquipmentContainers, and the white square boxes represent ConductingEquipment. Darker shading indicates the EquipmentContainer is higher up in the containment hierarchy (i.e., Substation is highest, VoltageLevel next, etc.). White circles represent ConnectivityNodes, and black small circles represent Terminals. A Terminal belongs to a ConductingEquipment, and a ConnectivityNode belongs to an EquipmentContainer. This means that the borders (or contact points) between ConductingEquipment are their Terminals interconnected via ConnectivityNodes.



Figure 10 – Simple network connectivity modeled with CIM Topology

The Line SS1-SS2 contains three ACLineSegments (Cable1, Cable2, and Cable3) and associated ConnectivityNode (CN2) to model a T junction, which provides a connection to SS4. This represents just one way that this configuration could be modeled. Each ACLineSegment has two Terminals. Cable1 is connected to CN3 and CN2 via these Terminals. CN3 is contained by the VoltageLevel 400KV. The breaker BR1 has two terminals of which one is connected to CN3.

Measurements are represented by square callouts where the arrow points to a Terminal. P1 is connected to the right Terminal belonging to Breaker BR1. Note that P1 is drawn inside the box representing BR1. This is because a Measurement may belong to a PowerSystemResource (PSR), as is the case with BR1. P2 is drawn inside the VoltageLevel 400KV, which means it belongs to the 400KV VoltageLevel instead of BR3.

### Inheritance hierarchy

Figure 11 shows an overview of the inheritance hierarchy modeled in the CIM. This overview, which is included as one of the Wires package diagrams, spans several of the CIM packages.

startUmlDiagram.Wires.InheritanceHierarchy.endUml

Figure 11 – Equipment inheritance hierarchy

### Transformer model

Figure 13 shows a portion of the Wires package in a class diagram which models a PowerTransformer device.

The transformer model has been revised from previous versions to allow for use of the same instance model in both balanced and unbalanced models. Additionally the PowerTransformer itself is not a ConductingEquipment with potentially multiple terminals to more directly model transformers with one, two, three, or more terminals. As typical of ConductingEquipment, when in service, the PowerTransformer conducts power from one of its terminals to another.

As shown in Figure 12 - Transformer and Tank model, the PowerTransformer is also able to optionally model tank details, which can be used to describe in detail the transformer internal winding phase connections and imbalances. In all cases a PowerTransformer models a group of physical devices acting together to transform power among terminals and in one physical location. Often for transmission systems three physical, single phase devices are represented by one PowerTransformer instance. If detail of the individual single phase devices were required, the TransformerTank objects should additionally be modeled.



Figure 12 - Transformer and Tank model

Both a PowerTransformer and a TransformerTank can have impedance described in either an asumed star connection with implied center connection or with mesh impedance form by using the optional TransformerMeshImpedance class. If using the mesh impedance form, you would specify a TransformerMeshImpedance class for each possible terminal to terminal connection, thus one instance for two terminals, three for three terminals, and six for four terminals and so on.

As shown, a PowerTransformer is a specialized class of ConductingEquipment, which is a specialized class of Equipment. This is shown by the use of the generalization-type of relationship, which uses an arrow to point to the general class. The ConductingEquipment is shown to be a specialized class of Equipment by use of the italic “*Equipment*” class name in the upper right corner. The inheritance permits the PowerTransformer to inherit attributes from ConductingEquipment, Equipment, and all generalizations of Equipment that are not shown in this diagram.

A PowerTransformer also has relationships to PowerTransformerEnd and TransformerTank which are modeled with an association-type of relationships. As shown, a PowerTransformer may associate zero or more PowerTransformerEnds, but a PowerTransformerEnd may associate to only zero or one PowerTransformer. Simlarly, the PowerTransformer may associate to zero or more TransformerTank objects, but a TransformerTank may associate to only zero or one PowerTransformer object.

The PowerTransformerEnd specializes the TransformerEnd class which has other relationships as well:

* a generalization relationship with IdentifiedObject;
* two association relationships with the TransformerMeshImpedance class, such that a TransformerEnd object may be “From” 0, 1, or more TransformerMeshImpedance objects and To 0, 1, or more TransformerMeshImpedance objects;
* aggregation relationships with the PhaseTapChanger and RatioTapChanger classes, such that a TransformerWinding object may have 0 or 1 PhaseTapChanger objects and 0 or 1 RatioTapChanger objects associated with it.

Clause 6 contains a complete description of each class in Figure 13 along with the definition of all the attributes and relationships supported in each class.

startUmlDiagram.Wires.Transformer.endUml

Figure 13 – Transformer model

### Transformer tap modeling

#### Generalized tap modeling

Multiple types of transformers tap models are supported by the CIM model. Depending on the physical construction, the effects of tap movement will be defined using either the RatioTapChanger class or the PhaseTapChanger class and its various specializations. The PhaseTapChanger can be specialized into a number of different types representing the physical construction and hence behaviour. The attributes of TapChanger are applicable for all the RatioTapChanger specializations.

Multiple tap changers can be combined within one power transformer, but it is expected that only one phase shifting and one ratio changing tap changer would be modeled on any given terminal. We document the ratio or phase shift change relative to the other terminal(s) assuming those terminals are at the nominal (1 PU) ratio and nominal (zero) phase angle shift. It is permitted to combine both a phase shifting and ratio tap changer on one terminal, though this practice is discouraged if the asymetrical transformer model is an appropriate representation.

Tabular represenations for both ratio and phase shifting transformers can be used where the available CIM models do not appropriatIely reflect the physical behaviour. The tabular models are optionally added to the the other CIM tap models. It is recommended to use the linear models when adding tabular models.

#### Voltage ratio transformer modeling

The RatioTapChanger class is used to define a transformer tap that changes the voltage magnitude ratio without changing the phase angle across the transformer. A RatioTapChanger may be used in cases the where the transformer inheritently introduces a fixed phase shift, such as from a wye-delta winding connection.

#### Voltage ratio transformer tabular modeling

A RatioTapChangerTabular may be optionally associated to any RatioTapChanger and represents a non-linear relationship between tap number, tap ratio, and impedances. Normally, if associated, the tabular ratio model takes precedence.

#### Linear phase shifter transformer modeling

A phase shifting transformer is linear if the phase shift increment per tap step is uniform over the full range of the tap changer. A linear phase shifter is modeled using the PhaseTapChangerLinear class.

#### Symmetrical phase shifter modeling

A symmetrical phase shifter construction is modelled with a PhaseTapChangerSymmetrical class and changes to voltage magnitude and phase over the full range of tap steps for a symmetrical phase shifter is shown in Figure 14 below.

Symmetrical_Phase_Shifter

Figure 14 – Symmetrical Phase Shifter

The phase shift angle can be represented as:



*ρ* = 1

Where α = phase shift angle, local terminal relative to other nominal terminal

*n’* = current tap position

Δ*U* = quadrature voltage change

*Ue* = voltage at neutral tap

δ*U* = voltage change per tap position

*ρ* = voltage magnitude transformation ratio, local terminal to other nominal terminal

*UL* = local terminal voltage

*US* = system or other terminal voltage

The sign of the n’δU term is used to indicate the relative phase shift at the associated terminal. If n’δU is positive, then the phase angle of the associated terminal is increased. If δU is negative then the phase angle of the associated terminal is decreased.

#### Asymmetrical phase shifter modeling

An asymmetrical phase shifter construction causes the voltage ratio to change as the phase shift changes. The changes to voltage magnitude and phase over the full range of tap steps for an asymmetrical phase shifter is shown in Figure 15 below.

Asymmetrical_Phase_Shifter

Figure 15 – Asymmetrical Phase Shifter

The phase shift angle can be represented as:



Where α = phase shift angle, local terminal relative to other nominal terminal

*n’* = current tap position

*δU* = voltage change per tap position

*Θ*= winding connection angle

*ρ* = voltage magnitude transformation ratio, local terminal to other nominal terminal

*UL* = local terminal voltage

*US* = system or other terminal voltage

An asymmetrical phase shifter is modeled with the PhaseTapChangerAsymetrical class. The PhaseTapChanger.windingConnectionAngle must be a multiple of 30 degrees: -90, -60, -30, 30, 60, 90 …). An asymmetrical phase shifter with a ninety degree winding connection angle is not the same as a symmetrical phase shifter.

#### Phase shifter transformer tabular modeling

A PhaseTapChangerTabular may be optionally associated to any PhaseTapChanger and represents a non-linear relationship between tap number, phase shift, and impedances. Normally, if associated, the tabular phase shifter model takes precedence.

### Phase wire modeling

The CIM models provide a means to specify equivalent single-line representation of balanced three phase networks or to additionally and optionally specify individual phase detail. In single phase network there is a direct correspondence between Terminal objects and physical wire connections. In the single-line representation of balanced three phase networks we use one Terminal to represent the group of coordinating phase connections. Thus one Terminal can be used to specify the one or more coordinated phase connections using the Terminal.phases attribute which is of enumeration type PhaseCode. PhaseCode has the possible phase connection combinations of ABC, ABCN, AB, BC, CA, A, B, C and so forth. The specification of phases at each terminal allows for modeling jumpers or switches that connect different phases. This phase connectivity specification is intended as a nominal or normal phase specification. The result of energization source is not reflected in this specification. This phase specification provides a means to locally connect like phases.

Figure 16 – Phase connectivity provides an example for specifying phase connections. This example shows a ConnectivityNode instance on the left with A, B, and C phases implied. Note that ConnectivityNode does not specify its phase content but rather implicitly connects like phases of terminals associated to the ConnectivityNode. The example shows a two phase jumper connecting on the left (Terminal.squenceNumber=1) phases A and C and on the right (Terminal.sequenceNumber=2) phases A and B. The internals of the jumper model SwitchPhase objects which specify in detail how the phase connections route. The Terminal.phases attribute only specifies the external connection of a device. Continuing to the right, two line segments are connected. The upper line segment is a two phase line on phases A and B. The lower line segment is single phase on phase B. The dark lines indicate CIM associations while the lighter arrows indicate the represented electrical connections at the phase level.



Figure 16 – Phase connectivity

As a clarification, the SwitchPhase attribute named phaseSideA specifies any single phase (such as A, B, C, or N) and refers to the Terminal with sequenceNumber equal to 1. The SwitchPhase attribute named phaseSideB refers to the Terminal with sequenceNumber equal to 2.

The switch model allows for cross phase connections to be modeled in general within the device. The transformer model also has explicit internal device modeling of phase connectivity. At this time all other ConductingEquipment branch specializations such as ACLineSegment assume like phases connect at both terminals. The CIM model is open for future extensions to introduce internal device phase connectivity.

### Cuts, clamps and jumpers model

The UML model for cuts and jumpers on a line segment and the additional connections is described in Figure 17 – Cuts, clamps, and jumpers UML model. The Jumper class is a specialization of the more general Switch class and provides for phase crossings and phase connectivity as was illustrated in Figure 16 – Phase connectivity. The Jumpers therefore can connect Terminals in the normal manner. The CIM classes Cut and Clamp provide a means to effectively add Terminals to an ACLineSegment at a specified distance down the line by associating new CIM objects to an ACLineSegment object. Each Clamp instance introduces one new terminal connected at the specified distance down the line and without interruption of the line. Each Cut instance introduces two new terminals which are effectively the ends of the cut. By convention, each cut’s terminal is oriented toward the ACLineSegment terminal with the same sequenceNumber. If the cut is open, no flow passes through the ACLineSegment across the point of the cut, but flow can pass through either side of the ACLineSegment and through the appropriate cut terminal. Commonly, jumpers are connected to the ends of cuts, but any type of conducting equipment can be connected in the normal manner using ConnectivityNode or TopologicalNode.

startUmlDiagram.Wires.CutsAndJumpers.endUml

Figure 17 – Cuts, clamps, and jumpers UML model

An example of the addition of cuts and clamps to an existing CIM model is illustrated in Figure 18 – Example before cuts and jumpers and Figure 19 – Example after cuts and jumpers. Figure 18 shows two physically parallel line segments named “FRED” and “LEROY”. In this example the system is energized from the right, the switch on the left is closed, and the loads name “ONE” and “TWO” are being supplied. Then the three faults are applied at the specified distances from Terminal 1 of the line respective line segments segments. In this example the phase detail is not specified for simplicity, but phase detail can follow the same approach outlined in Clause 4.4.8.



Figure 18 – Example before cuts and jumpers applied

Figure 19 shows how the pure addition of Cut, ConnectivityNode, Jumper, Clamp and potentially other Conducting Equipment can represent model changes reflecting typical temporary work. These same models can also be used for permanent model changes. This shows how cuts applied at the specified distances can isolate the faults.

Figure 19 shows how this example leaves the faulted ends of cuts dangling, while the unfaulted network is connected to supply the loads. This example also demonstrates how multiple cuts can be applied to one line segment. This example shows open cuts, but if a cut were closed, flow could continue through the ACLineSegment with part of the flow potentially redirected to the terminals of the cut, much like the case for a clamp.



Figure 19 – Example after cuts and jumpers applied

**Although not shown in the example above, a cut provides two terminals and connections could be made to either or both terminals. An example case might involve introducing a cut and connecting both ends with a jumper that crosses phases.**

### Measurements and controls

#### Measurement overview

Measurements are used to represent the state variables that can be found in industrial processes. Each industrial process has its specific types of measurements. A power system typically has power flows, voltages, positions (e.g., breakers, isolators), fault indications (air pressure, oil pressure over temperature, etc.), counters (e.g., energy), etc.

The name "Measurement" would seem to indicate that all the state variables are measured. This is not always strictly the case as many measurements are calculated by SCADA or EMS/DMS functions, such as State Estimator or Power Flow calculations. As a consequence, a measurement may have a number of alternate values (e.g., manually supplied, telemetered, state estimated, optimized etc.). This is supported by the Measurement and MeasurementValue models in the Meas package. The classes in the StateVariable package have been added to the CIM model specifically for the purpose of supporting exchange of values calculated by functions such as State Estimator and Power Flow.

The measurement model now allows for optionally specifying the specific phases of a measurement.

#### Control overview

Controls are used to represent control variables. Power system control variables typically are set points, raise lower commands, select before execute commands and on/off commands etc. The Meas package supports control variables with the Control model.

#### Use of measurement-related classes

A PowerSystemResource (PSR) may have zero to many measurements associated with it by containing one or more measurements. Each measurement may have one or more measurement values. Observing the following guidelines will enable applications to navigate and find the required measurement values in a consistent way (see Figure 20):

a) Measurements of a PowerSystemResource are classified by the attribute measurementType. The values to be used for Measurement.measurementType are specified in Table 1.

b) MeasurementValues of a Measurement are classified by MeasurementValueSource.

c) MeasurementValueSource inherits from IdentifiedObject. The values to be used for MeasurementValueSource.name and MeasurementValueSource.description are given in Table 2. This table provides a number of source names to be used where possible. However, the exact names to be used for specific applications are defined in related IEC 61970 Component Interface Specifications (CIS).

d) The tables may be extended for proprietary needs. The names added must start with a unique name (e.g. the company name) and an underscore. Example: xyz\_AverageTemperature.

e) The ValueAliasSet is used for discrete measurements and describes mappings from values to symbolic names, such as enumeration literals having specific integer values. Different communication protocols (e.g. for RTUs and for control centers as ICCP or ELCOM) use different data encodings. A system may have a system wide mapping for all Discrete values or group the Discrete values and make a mapping per group. Creation of a single system wide mapping that covers existing communication protocols is outside the scope of this specification.

startUmlDiagram.Meas.Measurement.endUml

Figure 20 – Navigating from PSR to MeasurementValue

Table 2 – measurementType naming conventions

|  |  |  |
| --- | --- | --- |
| measurementType | 61850 Name | description |
| Current | Amp | Current (rms) of a non-three phase circuit |
| ThreePhaseCurrent | AvAmps | Total current (rms) in a three phase circuit |
| PhaseCurrent | A | Measured phase current. |
| Frequency | Hz | Frequency |
| PowerFactor | PwrFact | Power Factor not allocated to a phase |
| ThreePhasePowerFactor | TotPF | Average power factor in a three phase circuit |
| ThreePhaseApparentPower | TotVA | Total apparent power in a three phase circuit |
| ThreePhaseReactivePower | TotVAr | Total reactive power in a three phase circuit |
| ThreePhaseActivePower | TotW | Total real power in a three phase circuit. |
| ApparentPower | VoltAmp | Apparent power in a non-three phase circuit |
| ReactivePower | VoltAmpr | Reactive power in a non-three phase circuit |
| Voltage | Vol | Voltage (rms) not allocated to a phase |
| ActivePower | Watt | Real power in a non-three phase circuit |
| Pressure | Pres | Pressure |
| Temperature | Tmp | Temperature |
| Angle | Ang | Angle between voltage and current |
| ApparentEnergy | TotVAh | Apparent energy |
| ReactiveEnergy | TotVArh | Reactive energy |
| ActiveEnergy | TotWh | Real energy |
| Automatic | Auto | Automatic operation (not manual) |
| LocalOperation | Loc | Local operation (not remote) |
| SwitchPosition | Pos | Switch position  [2bits= intermediate,open,closed,ignore] |
| TapPosition | TapPos | Tap position of power transformer or phaseshifter |
| Operation Count | OperCnt | Operation count - typically for switches |
| LineToNeutralVoltage |  | Line to neutral voltage. |
| LineToGroundVoltage |  | Line to ground voltage. |

Table 2 describes various types of measurements also defined in IEC 61850. The meaning of the columns in Table 2 are as follows:

* measurementType (Measurement.measurementType) is the IEC 61970 measurement type name.
* 61850 Name is the name assigned to the data in IEC 61850. (Refer to Clause 6 of IEC 61850-7-4, data object name semantics).
* description of the data.

It shall be noted that Table 2 is a non exhaustive list and that the mapping between measurements as defined in a control center and a substation is non-trivial.

Table 3 – MeasurementValueSource naming conventions

| name | description |
| --- | --- |
| SCADA | Telemetered values received from a local SCADA system |
| CCLink | Value received from a remote control center via TASE.2 or other control center protocol |
| Operator | Operator entered value (always manually maintained, PSR is not connected to an RTU) |
| Estimated | Value updated by a state estimator |
| PowerFlow | Value updated as result of a Powerflow |
| Forecasted | Value that is planned or forecasted. |
| Calculated | Calculated from other measurement values (e.g., a sum) |
| Allocated | Calculated by a load allocator |

Following these conventions:

* each Measurement instance represents a technological quantity of a PowerSystemResource;
* each MeasurementValue of a Measurement represents a value for the technological quantity, as supplied from a single source;
* the source attribute in MeasurementValueQuality then indicates whether the source actually provided the current value, or whether it had been substituted or defaulted.

Note that a new MeasurementValue identity is not normally created for each exchange of a measured value. It is expected that the same MeasurementValue instance could exchange new values in subsequent messages. Therefore the identity of a MeasurementValue instance can be established apriori to exchange of measured values. The modeling of a time series of values is not explicitly expressed in the CIM model, though the model allows for systems to internally build such time series models through a series of measurement value exchanges.

#### Attachment of measurements

As mentioned in the previous subclause and as shown in Figure 20, Measurements are contained by a PowerSystemResource. This is sufficient for Measurements that are not related to connectivity, e.g. temperature, weight, size.

To specify the location of a Measurement in the network, an association to Terminal is used. Examples include power flows, voltages, and currents. Voltages have no direction and can be attached wherever appropriate in relation to the sensor placement. Flows have direction and must be attached such the flow direction is evident from the placement.

Figure 21 shows two examples of the placement of Measurements.



Figure 21 – Measurement placement

P12 is a voltage Measurement that measures the voltage at the Junction J1. P12 is topologically related to the ConnectivityNode CN1 via the Terminal in Junction J1. P11 is a Measurement that measures the flow through Breaker BR10 at the side connected to the ConnectivityNode CN1. P11 is topologically related to the ConnectivityNode CN1 via the left Terminal in Breaker BR10. Temp is a Measurement that measures the Breaker temperature. As a temperature is not related to connectivity, it has no relation to a Terminal - it just belongs to the Breaker BR10.

#### ICCP measurements

ICCP (known officially as IEC 61870-6 TASE.2) data sourcs are specified using the MeasurementValue classes (and its specializations) and the MeasurementValueSource class. The MeasurementValueSource class is used to define the control center supplying the ICCP data. The MeasurementValueSource must be associated with an instance of Name where the attribute Name.name holds the name of the supplying control center. The instance of NameType associated with the control center Name must have the NameType.name attribute set to “ICCP”.

The MeasurementValue classes are used to specify the ICCP ID. The MeasurementValue must be associated with an instance of Name where the attribute Name.name holds the ICCP ID. The instance of NameType associated with the ICCP ID Name must have the NameType.name attribute set to “ICCP”. The MeasurementValue.name attribute holds the SCADA point name. Each MeasurementValue will be associated with one Measurement. Each MeasurementValue being supplied via ICCP must also have an association to a MeasurementValueSource.

### Regulating control models

Regulation control, such as automatic voltage control at generators or voltage tap control at transformers is modelled using a class named RegulationControl. The RegulatingControl class provides the capability to model multiple equipment participating in a regulation scheme. These regulation schemes may be physical or manually implemented in actual power system operation, but are reflected in the models used for power system analysis purposes. Figure 22 shows how RegulatingCondEq and TapChanger classes can participate in regulation.

startUmlDiagram.Wires.RegulatingEquipment.endUml

Figure 22 – Regulating control models

## Modeling guidelines

### Modeling for change

The following subclauses provide guidelines on how to maintain and extend the CIM.

The CIM is meant to contain classes and attributes that will be exchanged over public interfaces between major applications. The goal is to keep, as much as possible, only the generic features from which a detailed implementation may be derived. In general, it is easier to change the value or domain of an attribute than to change a class definition. This makes the model more robust because it is able to support a broader class of requirements, and more stable because new requirements may be able to be handled without requiring changes to the model.

### Process for amendments to the CIM

IIt may be desirable to amend the CIM to either revise the existing model or to extend the CIM to model additional elements of an electric utility power system. The recommended process for such amendments is as follows.

1. Prepare a Use Case(s) to describe the desired changes. This should include proposed changes to the appropriate class diagrams showing new/revised classes, attributes, and associations.
2. The Use Case(s) is then reviewed by the appropriate IEC Working Group to decide if the requested changes should be treated as revisions to the current CIM standard, or if they should be treated as private amendments, not requiring a change to the standard itself.
3. Proposed amendments accepted by the working group will be added to a list of outstanding issues, and at the appropriate time, a new version of the CIM model will be prepared and an update made to the appropriate IEC CIM specification.

### Changes to the CIM UML model

From a modeling perspective, when the CIM is extended, the approach is to start with the existing CIM UML model. The extensions may be added in any of several ways that are available in UML, but in all cases the approach is to inspect the current model and determine the best way to build off of the existing class diagrams. The extensions may take the form of any of the following, starting from the simplest to the most complex:

* adding additional attributes to existing classes;
* adding new classes that are specializations of existing classes;
* adding new classes via associations with existing classes.

The main objective is to reuse the existing CIM to the maximum extent possible. From a packaging point of view, extensions should be made to existing Packages where possible. If the extensions comprise a new domain of application, then consideration should be given to creating a new Package for the additions, but still creating the necessary associations to the existing package, keeping in mind that even though a new Package is being created, the CIM is still a single model.

### Changes to the CIM standards documents

From a documentation perspective, when the CIM is extended, a decision must be made whether the changes constitute updates to existing CIM standards documents, or whether a new Part 3xx specification is required. In either case, the extensions will then become part of the IEC standard CIM.

### CIM profiles

An implementation of the CIM need not include all classes, attributes, or associations in the standard CIM specification to be compliant with the CIM standard. Profiles may be defined to specify which elements must be included (i.e., mandatory elements) in a particular use of the CIM, as well as which are optional. These profiles will be defined in the Part 4xx series of standards.

An example is the profile for exchanging power system models. This profile specifies how the CIM is to be used for exchanging power system models in XML, and also specifies the mandatory and optional classes, attributes, and associations to be supported for this use of the CIM. Profiles can be maintained using a variety of tools such as the open source CIMTool, available at http://www.CIMTool.org.

## Modeling tools

This CIM release was constructed using Sparx Systems Enterprise Architect product. The entire CIM UML model exists as an Enterprise Architect project file and is viewable with that tool, including the class diagrams and descriptions of classes, attributes, types, and relationships. Viewing the CIM in this fashion provides a graphical navigation interface that permits all CIM specification data to be viewed via point-and-click from the class diagram in each package. A free viewer is available through Sparx Systems.

Ideally, the CIM information model is independent of any specific UML tool, though experience has shown that exchanges between different tools are often less than perfect. Until tool interoperability is proven effective, future changes to the CIM specification, resulting in new versions of this standard, will be incorporated first into the Enterprise Architect project description to ensure a single source for the CIM model data.

Clause 6 of this document was auto-generated using the open source tool called jCleanCim which uses the automation interfaces of Microsoft Word and Enterprise Architect and using the standard IEC format and styles.

## User implementation conventions

### Conventions beyond UML

This following subclauses provide recommended user conventions when using the CIM in actual system implementations.

### Number of Terminals for ConductingEquipment objects

The following ConductingEquipment classes have two terminals: ACLineSegment, DCLineSegment, Jumper, Fuse, Breaker, Disconnector, LoadBreakSwitch, Cut, and SeriesCompensator. The PowerTransformer class typically has two terminals, but may also have one or more terminals. For example a zig-zag connected grounding transformer may have one terminal. Three terminal transformers are commonly used in transmission systems and in special cases transformers may have four, five, or more terminals. All other ConductingEquipment leaf classes (notably also including Clamp and BusbarSection) have a single terminal.

## CIM modeling examples

Power system models have been created from the CIM UML model in various ways. The first example is an RDF (Resource Description Framework) Schema version of the CIM, which uses XML (eXtensible Markup Language) to describe a power system network model. IEC 61970-501 and other profile standards in development are used to specify the model exchange format. RDF Schema versions of the CIM have been used to create XML model files of actual networks for purposes of interoperability testing. An RDF Schema version of the CIM is generated from the CIM UML model file using software tools based on the RDF Schema specification of the CIM.

It should be noted that an RDF schema version of the CIM is still metadata rather than an instantiation of an actual network. However, complete network model files with descriptions of all network elements and their electrical connectivity can be generated by system suppliers using proprietary export tools, and then imported by other systems via a similar import tool, which is used to populate a local network engineering tool database. Examples of such CIM XML models include the Siemens 100 bus model, the Alstom 60 bus model, and the ABB 40 bus model files[[2]](#footnote-2)) used for CIM XML interoperability testing.

# Detailed model

## Overview

The Common Information Model (CIM) represents a comprehensive logical view of Energy Management System information. This definition includes the public classes and attributes, as well as the relationships between them. The following subclauses describe how clause 6 is structured. Clause 6 is automatically generated from theCIM model maintained with the tools described in subclause 4.6.

## Context

The CIM is partioned into subpackages. The Domain package defines datatypes used by the other packages. The Generation package is subdivided into Production and GenerationDynamics packages. Classes within the packages are listed alphabetically. Native class attributes are listed first, followed by inherited attributes in order of depth of inheritance, then by attribute name. Native associations are listed first for each class, followed by inherited associations in order of depth of inheritance, then alphabetically by class name, then alphabetically by association end name. The associations are described according to the role of each class participating in the association. The association ends are listed under the class at each end of an association.

Figure 23 shows the top level packages included in this document.



Figure 23 – CIM top level packages

For each package, the model information for each class is fully described. Attribute and association end information for native and inherited attributes is listed as below. For any inherited attributes or association ends the “description” column will contain text indicating the attributes is inherited from a specific class. The description column for native attributes and association ends contains the actual description.

Attributes

|  |  |  |
| --- | --- | --- |
| name | type | description |
| native1 | Float | A floating point native attribute of the class is described here. |
| native2 | ActivePower | Documentation for another native attribute of type ActivePower. |
| Name | Float | Inherited from class IdentifiedObject |

In the Attributes table, in some cases, an attribute is a constant, in which case the phrase “(const)” is added in the name column of the attributes table. In such cases, the attribute normally has an initial value also which is preceded by an equal sign and appended to the attribute name.

Association ends

| [mult from] | [mult to] name | type | description |
| --- | --- | --- | --- |
| [0..\*] | [0..1] PSRType | PSRType | PSRType (custom classification) for this PowerSystemResource. |
| [0..1] | [0..\*] Measurements | Measurement | The Measurements that are included in the naming hierarchy where the PSR is the containing object |
| [1..1] | [0..\*] OperatingShare | OperatingShare | The linkage to any number of operating share objects. |
| [0..\*] | [0..\*] PsrLists | PsrList |  |
| [1..1] | [0..1] OutageSchedule | OutageSchedule | A power system resource may have an outage schedule |
| [0..\*] | [0..\*] ReportingGroup | ReportingGroup | Reporting groups to which this PSR belongs. |
| [0..1] | [0..\*] DiagramObjects | DiagramObject | inherited from: IdentifiedObject |
| [1..1] | [0..\*] Names | Name | inherited from: IdentifiedObject |

In the association ends table, the first column describes the multiplicity at this end of the association (i.e., how this class participates in association). Second column describes the other association end. Its multiplicity is included in brackets. The association end name is listed in plain text. The class at the other end of the association is listed in the “type” column. A multiplicity of zero indicates an optional association. A multiplicity of “\*” indicates any number is allowed. For example, a multiplicity of [1..\*] indicates a range from 1 to any larger number is allowed.

In the case that a class is an enumeration, the Attributes table is replaced by the Enums table as shown below, since enumeration literals have no type. There are no inherited enumeration literals for an enumeration class.

Enums

|  |  |
| --- | --- |
| literal | description |
| native1 | This is the first native enumeration value. |
| native2 | This is the second native enumeration value. |
| native3 | There are typically no inherited attributes for enumerations. |

# Package architecture (normative)

## startUmlPackage.IEC61970.endUml

Bibliography

IEC 61970-501, *Energy Management System Application Program Interface (EMS-API) – CIM RDF Schema*

RP-3654-1 EPRI Control Center API (CCAPI) research project.

CIMTool – Open source CIM profile management tool. <http://www.cimtool.org>

CIM Users Group – The community of CIM users. [http://cimug.ucaiug.org](http://cimug.ucaiug.org/)

DNP (Distributed Network Protocol): Series of standards. <http://www.dnp.org/>

RP570 (RTU Protocol based on IEC 57 part 5-1 (present IEC 870)). <http://library.abb.com/GLOBAL/SCOT/scot229.NSF/VerityDisplay/9A5C1896695487E6C2256A7200361578/$File/REC501RP570_EN_A.pdf>

IEC 60050 International Electrotechnical Vocabulary

IEC 61968-11 Ed.1: Application integration at electric utilities - System interfaces for distribution management - Part 11: Common Information Model (CIM) Extensions for Distribution

W3C: RDF/XML Syntax Specification <http://www.w3.org/RDF/>

W3C: Extensible Markup Language (XML) 1.0 <http://www.w3.org/XML/>

\_\_\_\_\_\_\_\_\_

1. ) The National Committees are requested to note that for this publication the maintenance result date is June 2011. [↑](#footnote-ref-1)
2. ) Siemens100 bus model, Alstom 60 bus model, and ABB 40 model are the trade names of products supplied by Siemens, Alstom Grid, and ABB. This information is given for the convenience of users of this document and does not constitute an endorsement by IEC of the product named. Equivalent products may be used if they can be shown to lead to the same results. [↑](#footnote-ref-2)