57/1262/CDV

COMMITTEE DRAFT FOR VOTE (CDV)
PROJET DE COMITÉ POUR VOTE (CDV)

	Project number Numéro de projet	IEC 61970-552 Ed.1
	IEC/TC or SC: **57** CEI/CE ou SC:	Secretariat / Secrétariat **Germany**
☒ Submitted for parallel voting in CENELEC Soumis au vote parallèle au CENELEC	Date of circulation Date de diffusion **2012-06-29**	Closing date for voting (Voting mandatory for P-members) Date de clôture du vote (Vote obligatoire pour les membres (P)) **2012-11-30**
Also of interest to the following committees Intéresse également les comités suivants -		Supersedes document Remplace le document 57/1185/CD - 57/1251/CC

Proposed horizontal standard
Norme horizontale suggérée

☐ Other TC/SCs are requested to indicate their interest, if any, in this CDV to the TC/SC secretary
Les autres CE/SC sont requis d'indiquer leur intérêt, si nécessaire, dans ce CDV à l'intention du secrétaire du CE/SC

Functions concerned
Fonctions concernées

☐ Safety / Sécurité ☐ EMC / CEM ☐ Environment / Environnement ☐ Quality assurance / Assurance qualité

Titre: CEI 61970-552:

Interface de programmation d'application pour système de gestion d'énergie (EMS-API) –

Partie 552: Format d'échange de modèle CIM XML

Title: IEC 61970-552:

Energy Management System Application Program Interface (EMS-API) –

Part 552: CIM XML Model Exchange Format

Note d'introduction

-

Introductory note

-

ATTENTION
VOTE PARALLÈLE
CEI – CENELEC

L'attention des Comités nationaux de la CEI, membres du CENELEC, est attirée sur le fait que ce projet de comité pour vote (CDV) de Norme internationale est soumis au vote parallèle.
Les membres du CENELEC sont invités à voter via le système de vote en ligne du CENELEC.

ATTENTION
IEC – CENELEC
PARALLEL VOTING

The attention of IEC National Committees, members of CENELEC, is drawn to the fact that this Committee Draft for Vote (CDV) for an International Standard is submitted for parallel voting.
The CENELEC members are invited to vote through the CENELEC online voting system.

Table of Contents

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

Energy Management System Application Program Interface (EMS-API) –

Part 552: CIM XML Model Exchange Format

FOREWORD

1) The IEC (International Electrotechnical Commission) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of the 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, the IEC publishes International Standards. 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. The 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 the 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 National Committees.

3) The documents produced have the form of recommendations for international use and are published in the form of standards, technical specifications, technical reports or guides and they are accepted by the National Committees in that sense.

4) In order to promote international unification, IEC National Committees undertake to apply IEC International Standards transparently to the maximum extent possible in their national and regional standards. Any divergence between the IEC Standard and the corresponding national or regional standard shall be clearly indicated in the latter.

5) The IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with one of its standards.

6) Attention is drawn to the possibility that some of the elements of this International Standard may be the subject of patent rights. The IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 61970-552 has been prepared by subcommittee 57, of IEC technical committee: Power System Management and Associated Information Exchange:

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

FDIS	Report on voting
XX/XX/FDIS	XX/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.

The committee has decided that the contents of this publication will remain unchanged until _____. At this date, the publication will be

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

INTRODUCTION

This standard is part of the IEC 61970 series that define an application program interface (API) for an energy management system (EMS).

IEC 61970-301 specifies a Common Information Model (CIM): a logical view of the physical aspects of an electric utility operations. The CIM is described using the Unified Modeling Language (UML), a language used to specify, visualize, and document systems in an object-oriented manner. UML is an analysis and design language; it is not a programming language. For software programs to use the CIM it must be transformed into a schema form that supports a programmable interface.

IEC 61970-501 describes the translation of the CIM in UML form into a machine readable format as expressed in the Extensible Markup Language (XML) representation of that schema using the Resource Description Framework (RDF) Schema specification language.

IEC 61970-552 specifies how the CIM RDF schema specified in 61970-501 is used to exchange power system models using XML (referred to as CIM XML) defined in the 61970-45x series of profile standards, such as the draft IEC 61970-452 CIM Transmission Network Model Exchange Profile.

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

EMS-API –

Part 552: CIMXML Model Exchange Format

1 Scope

This International Standard specifies a Component Interface Specification (CIS) for Energy Management Systems Application Program Interfaces. This part specifies the format and rules for exchanging modeling information based upon the CIM. It uses the CIM RDF Schema presented in IEC 61970-501 as the meta-model framework for constructing XML documents of power system modelling information. The style of these documents is called CIM XML format.

Model exchange by file transfer serves many useful purposes. Profile documents as draft IEC 61970-452 and other profiles in the 61970-45x series of standards explain the requirements and use cases that set the context for this work. Though the format can be used for general CIM-based information exchange, specific profiles (or subsets) of the CIM are identified in order to address particular exchange requirements. The initial requirement driving the solidification of this specification is the exchange of transmission network modeling information for power system security coordination.

This standard supports a mechanism for software from independent suppliers to produce and consume CIM described modeling information based on a common format. The proposed solution:

- Is both machine readable and human readable, although primarily intended for programmatic access,
- Can be accessed using any tool that supports the Document Object Model (DOM) and other standard XML application program interfaces,
- Is self-describing,
- Takes advantage of current World Wide Web Consortium (W3C) recommendations.

This document is the Level 2 Component Interface Specification document that describes in narrative terms (with text and examples based on the CIM) the detailed definition of the CIM XML format.

2 Normative references

The following normative documents contain provisions, which, through reference in this text, constitute provisions of this International Standard. At the time of publication, the editions indicated were valid. All normative documents are subject to revision, and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

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 61970-301 Ed 2: Energy management system application program interface (EMS-API) – Part 301: Common Information Model (CIM) Base

IEC 61968-11 Common Information Model (CIM) Extensions for Distribution

IEC 61970-501 Ed.1: Energy management system application program interface (EMS-API) - Part 501: Common information model resource description framework (CIM RDF) Schema

IEC 60050 series: International Electrotechnical Vocabulary

W3C: RDF/XML Syntax Specification

W3C: Extensible Markup Language (XML) 1.0

W3C: XSL Transformations (XSLT)

W3C: Document Object Model (DOM)

3 Definitions

Refer to International Electrotechnical Vocabulary, IEC 60050, for general glossary definitions and IEC 61970-2, for EMS-API glossary definitions. For the purposes of this International Standard, the following apply.

Application Program Interface (API)
The set of public functions provided by an executable application component for use by other executable application components.

CIM - Common Information Model
The CIM is an abstract model that represents all the major objects in an electric utility enterprise typically contained in an EMS information model. 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 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.

Document Object Model (DOM)
The Document Object Model is a platform- and language-neutral interface defined by the World Wide Web Consortium (W3C) that allows programs and scripts to dynamically access and exchange the content, structure and style of documents.

DTD – Document Type Definition: a standard for describing the vocabulary and syntax associated with an XML document. XML Schema and RDF are other forms that can be used.

Energy Management System (EMS)
A 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.

HTML – Hypertext Markup Language
HTML is a markup language used to format and present information on the Web.

Model
The term Model, when used by itself, refers to a collection of data describing objects or entities real or computed. In power system analysis, a model is a set of static data describing the power system. That interpretation is included here; however, the term is used more broadly to include any instance data governed by a Profile. Examples of Models include the Static Network Model,

the Topology Solution, and the Network Solution produced by a powerflow or state estimator application.

Profile
A Profile is schema that defines the structure and semantics of a model that may be exchanged. A Profile is a restricted subset of the more general CIM.

Profile Group
A collection of profiles intended to be used together for a particular business purpose. A Profile Group governs a set models.

RDF - Resource Description Framework
RDF is a language recommended by the W3C for expressing metadata that machines can process simply. RDF uses XML as its encoding syntax.

RDF Schema
RDF Schema is a schema specification language expressed using RDF to describe resources and their properties, including how resources are related to other resources, which is used to specify an application-specific schema.

Real-World Object
Real-world objects are objects that belong to the real world problem domain as distinguished from interface objects and controller objects within the implementation. The real-world objects for the EMS domain are defined as classes in IEC 61970-301 Common Information Model.

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 PowerTransformer, GeneratingUnit, 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.

SGML – Standard Generalized Markup Language
SGML is an international standard for the definition of device-independent, system-independent methods of representing texts in electronic form. HTML and XML are derived from SGML.

UML® - Unified Modeling Language™
UML is an object-oriented modeling language and methodology for specifying, visualizing, constructing, and documenting the artifacts of a system-intensive process.

URI - Uniform Resource Identifier:
a Web standard syntax and semantic for identifying (referencing) resources (things, such as files, documents, images).

XML - Extensible Markup Language
XML is a subset of Standard Generalized Markup Language (SGML), ISO 8879, for putting structured data in a text file. It is an endorsed recommendation from the W3C. It is license-free, platform-independent and well-supported by many readily available software tools.

XSL – eXtensible Stylesheet Language
XSL is a language for expressing stylesheets for XML documents.

4 Model exchange header

4.1 General

Model exchange typically involves the exchange of a collection of documents, each of which contains instance data (referred to as a model) and a header. The structure and semantics of each model is described by a profile, which is not included in the exchanged data. The overall exchange is governed by a collection of profiles called a Profile Group.

A header describes the content of the model contained in a document e.g. the date the model was created, description etc. The header may also identify other models and their relationship to the present model. Such information is important when the models are part of a work flow where, for example, the models have relations to each other, e.g. a model succeeds and/or depends on another.

The following subsections define the model with header data and work flow it is designed to support.

4.2 CIMXML documents and headers

A CIMXML document is described by a single header. Multiple headers in a CIMXML document are not allowed. Hence instance data in a CIMXML document adhere to a single profile.

In case multiple and possibly related CIMXML documents need to be kept together they shall be collected in an archive, e.g. zip.

4.3 Model and header data description

A description of a model is attached as header data to the model. Figure 1 describes the model with header information.

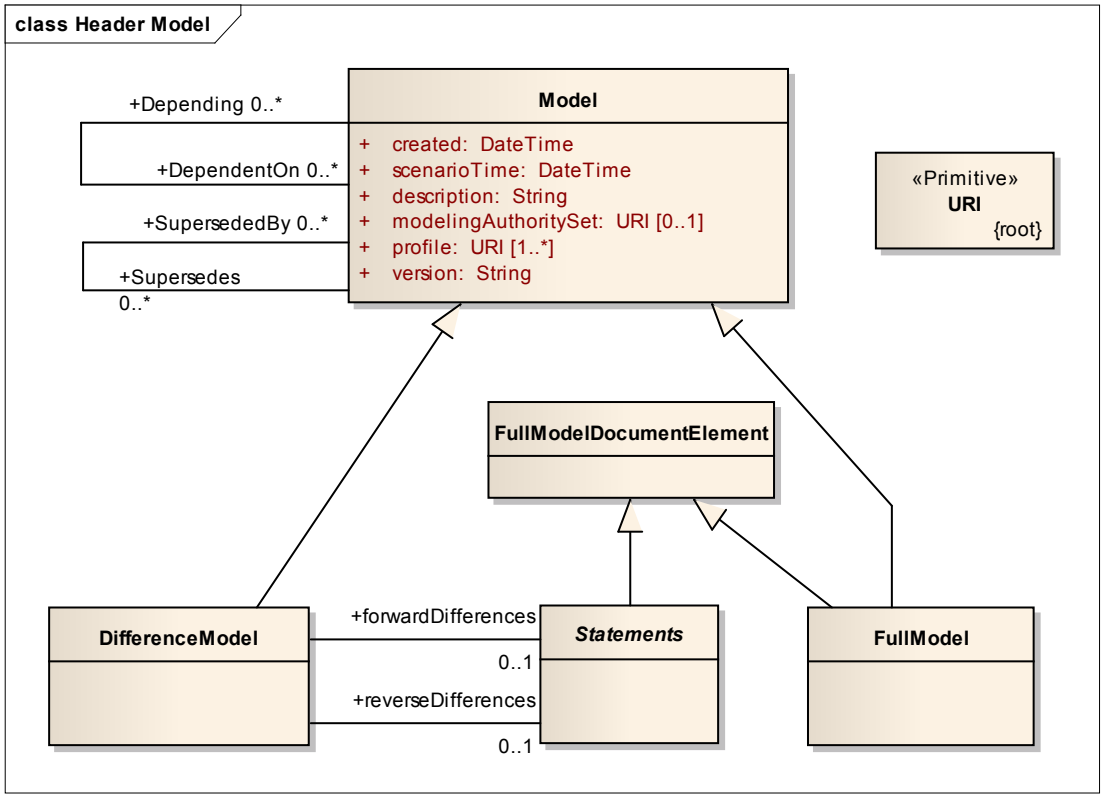


Figure 1 Model with header

In Figure 1 the classes FullModel, DifferenceModel and Statements describe the model data while the header is described by the classes Model and Description. The following is a bottom up description of these classes:

- The FullModelDocumentElement class represents any of the elements that may appear in a full model document. It has the two sub types Statements or FullModel, both are further described below. A full model document typically contains one FullModel element and one set of Definition elements.

- The Statements class represent a set of Definition (refer to section 5.2.3.4) and/or Description (refer to section 5.2.3.5) elements.

- The FullModel (refer to section 5.2.3.3) class represent the full model header and its contents is described by the Model class.

- The DifferenceModel (refer to section 5.2.4.6) class represents the difference model header. The content is described by the Model class, the association role forwardDifferences and association role reverseDifferences. Both association roles may have one set of Statements.

- The Model class describe the header content that is the same for the FullModel and the DifferenceModel. A Model is identified by a rdf:about attribute. The rdf:about attribute uniquely describe the model and not the document where the header exists. Hence multiple documents created from the same unchanged data model will have the same rdf:about.

The Model class attributes are described in Table 1.

Table 1 Header attributes

Class	Attribute	Description
Model	created	The date when the model was created (note this is typically not when the CIMXML document was created which is after this time).
Model	scenarioTime	The date and time that the model represents, e.g. the current time for an operational model, a historical model or a future planned model.
Model	description	A description of the model, .e.g. the name of person that created the model and for what purpose.
Model	modelingAuthoritySet	A uri describing the equipment model sourcing the data in a CIMXML document, e.g. a model for the whole or a part of a country.
Model	profile	A uri describing the Profiles that governs this model. It uniquely identifies the Profile and its version.
Model	version	A description of the version of the model sourcing the data in a CIMXML document. Examples are - Variations of the equipment model for the ModelingAuthoritySet - Different study cases resulting in different solutions.
Model	DependentOn	A reference to CIMXML documents describing the models that the model described by this document depends on, e.g. - A load flow solution depends on the topology model it was computed from - A topology model computed by a topology processor depends on the network model it was computed from.
Model	Depending	All models depending on this model. This role is not intended to be included in any document.
Model	Supersedes	When a model is updated the resulting model supersedes the models that were used as basis for the update. Hence this is a reference to CIMXML documents describing the updated models.
Model	SupersededBy	All models superseding this model. This role is not intended to be included in any document.

The profile attribute is an uri having the following format

http://iec.ch/<committee>/<year>/<standard>-<part>/<profileGroup>/<profile>/<version>

e.g. http://iec.ch/TC57/2011/61970-452/CPSM/Equipment/2

•

The UML in Figure 1 translates into CIMXML elements as follows

1. A leaf class in Figure 1 (DifferenceModel, Statements and FullModel) appears as class elements under the document element (section 6.2.3.3).
2. Statement elements appear as Definition (section 6.2.3.5) or Description elements (section 6.2.3.6).
3. Literal attributes, e.g. Model.created, appears as literal property elements (section 6.2.3.8).
4. Roles appear, e.g. Model.Supersedes, as resource property elements (section 6.2.3.10).
5. Inherited attributes and roles appear directly as elements under the leaf class following the rules 3, 4 and 5 above.

6. A CIMXML model document is identified by a Model rdf:about attribute (implicit in the UML). Hence the roles DependentOn and Supersedes are references to the Model rdf:about attribute.
7. A full model document may be regenerated multiple times from the same source data. Full model documents regenerated from unchanged source data keep the model identification (Model rdf:about) unchanged from the original full model document.
8. When generating a full model document superseding a differential the new full model document will have the same model identification (Model rdf:about) as the differential if the model is unchanged since the differential was created. Hence it is an alternate to the differential.

4.4 Work flow

A work flow is described by a sequence of exchange events. The model description in section 4.2 supports work flow events related in time with the Model.Supersedes attribute and events related to profiles with the Model.DependentOn attribute. An example of this is shown in Figure 2.

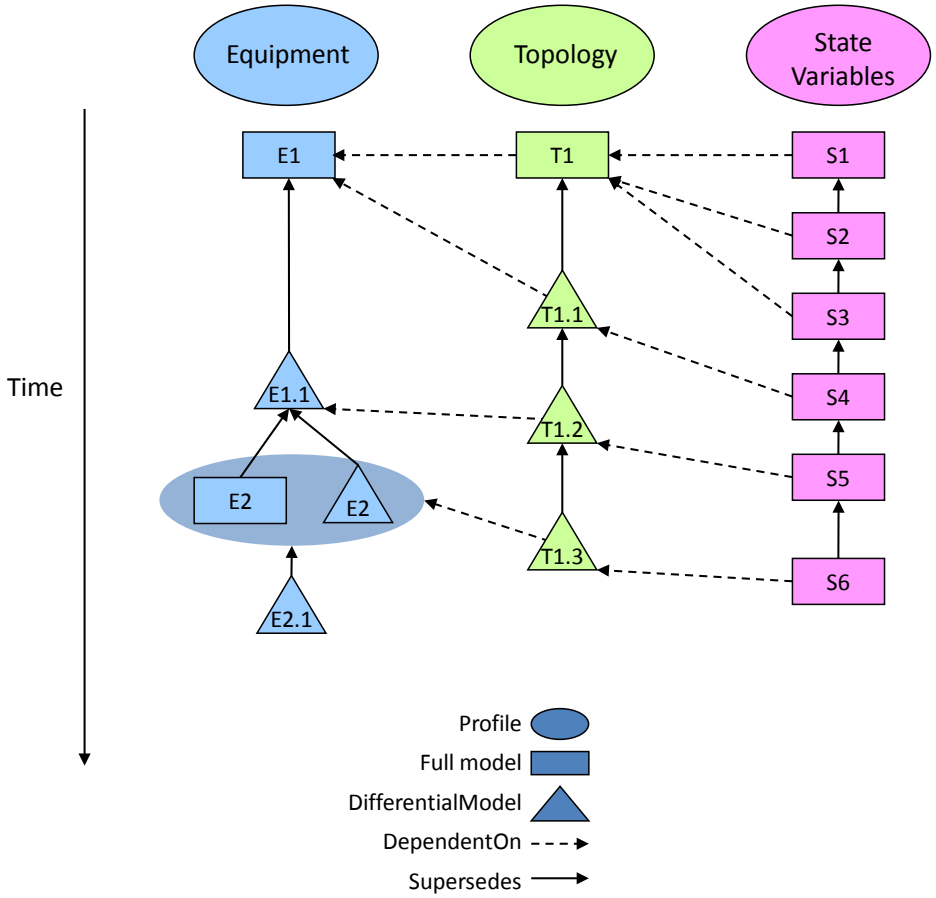


Figure 2 Example work flow events

In this example, a solved network model is exchanged as a collection of models governed by a Profile Group comprising Equipment , Topology, and State Variables documents. The left time line in Figure 2 represents how the Equipment model document is exchanged over time. The center time line shows how new Topology results are exchanged over time and the Equipment models on which each depends. The right most time line shows how multiple State Variable documents are

exchanged and the Topology files on which they depend. Also note that the equipment model E2 is represented both by a full and an incremental document. The situation in Figure 2 represents a simple case. A more complex situation is shown in Figure 3.

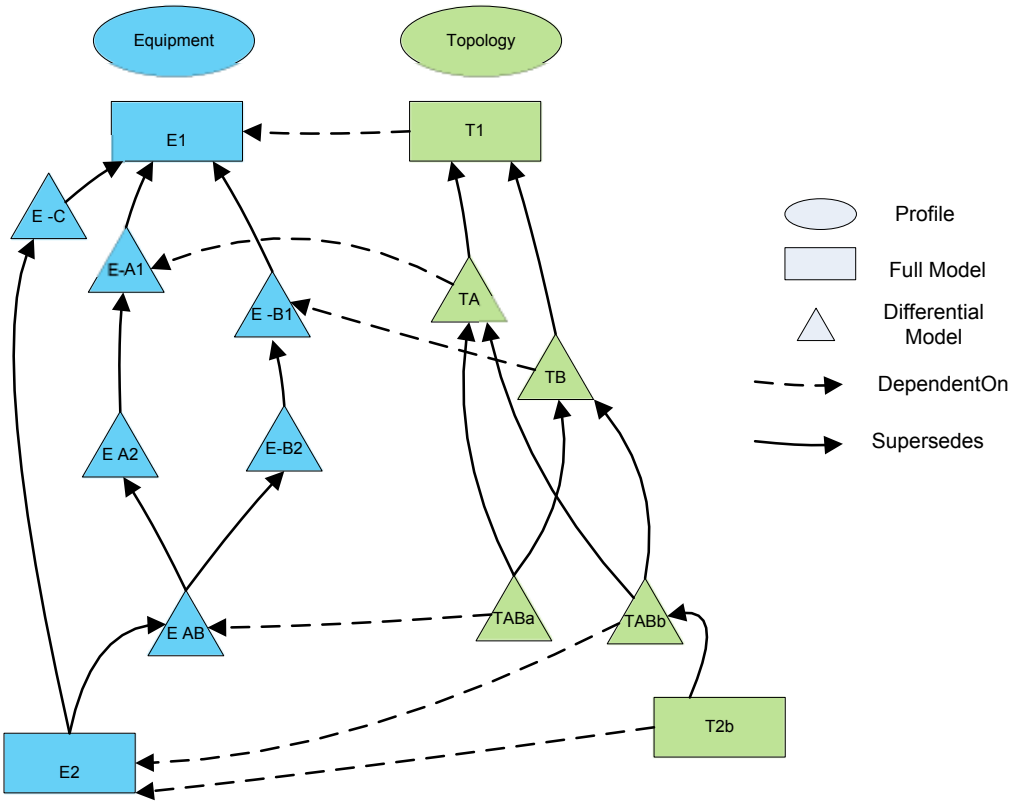


Figure 3 Example work flow events with more dependencies

The CIMXML documents in Figure 3 may be created from a data modeller environment where multiple change tracks of a model appear in parallel, e.g. the equipment model has three tracks E-Ax, E-Bx and A-C that eventually merge into the full model E2 superseding the equipment model tracks.

A receiver of the CIMXML documents may use any of the topology documents TA, TB, TABa or T2b with the equipment model from E2. As the sender (the data modeller in this example) only verified T2b with E2 this is the only combination that is supposed to fit together. Concerning T2b the receiver may choose to apply TB and TABb to T1 instead of using T2b.

5 Object identification

5.1 URIs as identifiers

CIM XML elements are identified by a URI. A URI can have two forms

- URL
- URN

The URL and URN forms have fundamentally different structures, i.e.

- URL form; *protocol*://*authority*/*path*?*query*#*fragment* where the *protocol* in CIM XML is http
- URN form: urn:*namespace*:*specification* where the *namespace* in CIM XML is uuid (for more details refer to section 6.4)

A CIM XML document may use either or.

5.2 About rdf:ID and rdf:about

A CIM XML element can be identified by two different RDF constructs

- rdf:ID
- rdf:about

The use of rdf:ID and rdf:about has a specific meaning that does not align with their definition in RDF. The meanings are

- an rdf:ID specify the life of object globally, i.e. it's creation or deletion.
- an rdf:about is a reference to an existing object.

5.3 CIM XML element identification

CIM XML files contain XML elements describing CIM objects (ACLineSegments, Substations etc.). The CIM has lots of association roles that show up as references in the XML elements (typically as rdf:resource or rdf:about attributes). CIM data show up in different CIM XML files that depend on each other as described in section 4. Some references then cross CIM XML file boundaries. A consequence of this is that the identification of a CIM object must be stable during its life time. Otherwise referencing objects across file boundaries will break.

A common practice in object oriented systems is to assume all objects has an identifier that is unique in space and time which means

- Different objects are assigned different identifiers.
- Identifiers once assigned are never reused even if the original object having it is gone.

Object identification is so central that it is implicitly modelled i.e. it is not explicit in an object model as a base class attribute (e.g. the class Object with the attribute mrid).

The different CIM XML identifications that are possible

- rdf:ID in the URL form.
- rdf:ID in the URN from.
- rdf:about in the URL.
- rdf:about in the URNform.

Note that the xml:base attribute is not required.

Some examples

- rdf:ID="_26CC8D713B7E4CF88C938D9D557A4846" where a UUID is used
- rdf:ID="ACLineSegment1" where a type and a number is used
- rdf:about="#_26CC8D713B7E4CF88C938D9D557A4846" where a UUID is used
- rdf:about="#ACLineSegment1" where a type and a number is used

345 • rdf:about=”urn:uuid:26CC8D71-3B7E-4CF8-8C93-8D9D557A4846” where a UUID is used

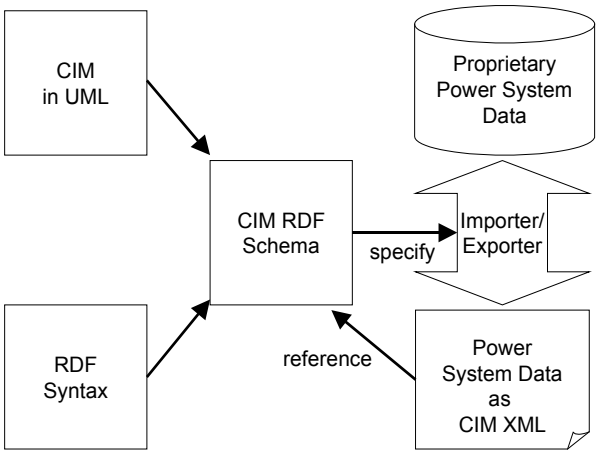
346

347 # 6 CIM XML format rules and conventions

348 ## 6.1 General

349 Given the CIM RDF Schema described in IEC 61970-501, a power system model can be converted
350 for export as an XML document (see Figure 3). This document is referred to as a CIM XML
351 document. All of the tags (resource descriptions) used in the CIM XML document are supplied by
352 the CIM RDF schema. The resulting CIM XML model exchange document can be parsed and the
353 information imported into a foreign system.

354



355

356 **Figure 3 CIM XML-based power system model exchange mechanism**

357 ## 6.2 Simplified RDF syntax

358 ### 6.2.1 General

359 RDF syntax provides many ways to represent the same set of data. For example, an association
360 between two resources can be written with a resource attribute or by nesting one element within
361 another. This could make it difficult to use some XML tools, such as XSL processors, with the CIM
362 XML document.

363 Therefore, only a subset of the RDF Syntax is to be applied in creating CIM XML documents. This
364 syntax simplifies the work of implementers to construct model serialization and de-serialization
365 software, as well as to improve the effectiveness of general XML tools when used with CIM XML
366 documents. The reduced syntax is a proper subset of the standard RDF syntax; thus, it can be
367 read by available RDF de-serialization software.

368 The following subsections define a subset of the RDF Syntax. This simplified syntax is for
369 exchanging power system models between utilities. The aim of the specification is to make it
370 easier for implementers to construct de-serialization software for RDF data, to simplify their
371 choices when serializing RDF data, and to improve the effectiveness of general XML tools such as
372 XSLT processors when used with the serialized RDF data.

The reduced syntax is a proper subset of the standard RDF syntax. Thus, it can be read by RDF de-serialization software such as SirPAC [8]. In this, it differs from other proposals for a simplified syntax, such as [9], [10].

The reduced syntax does not sacrifice any of the power of the RDF data model. That is, any RDF data can be exchanged using this syntax. Moreover, features of RDF such as the ability to extend a model defined in one document with statements in second document are preserved.

6.2.2 Notation

The simplified syntax is defined in the following section. Each kind of element is defined in a subsection beginning with a model of the element, followed by some defining text, and a reference to the RDF grammar. The semantics of the element are not detailed (refer to the RDF recommendation [3] for that information). The notation for the element model is as follows:

1. A symbol in italics in the position of an element type, attribute name or attribute value indicates the type of name or value required. The symbol will be defined in the text.
2. The symbol *rdf* stands for whatever namespace prefix is chosen by the implementation for the RDF namespace. Similarly the symbol *cim* stands for the chosen CIM namespace prefix.
3. A comment within the element model indicates the allowed content. A symbol in italics stands for a kind of element or other content defined in the text. A construction *(*a | b) indicates that *a* and *b* are alternatives. A construction a* indicates zero or more repetitions of a.
4. All other text in the model is literal.

6.2.3 Syntax definition (Normative)

6.2.3.1 General

The syntax definition is enriched with examples. The examples should help to get a better understanding of the formal syntax definition. The same example is used for several syntax definitions. The syntax focused in the example is indicated in bold.

6.2.3.2 Name space URIs defined in this specification

The following name spaces are defined in this specification

- `cim-model-description_uri` described by `xmlns:md`
- `difference-model-namespace-uri` described by `xmlns:dm`

Their values are defined as

- `xmlns:md=”http://iec.ch/TC57/61970-552/ModelDescription/1#”`
- `xmlns:dm=”http://iec.ch/TC57/61970-552/DifferenceModel/1#”`

6.2.3.3 Document element

```
<rdf:RDF xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
            xmlns:cim=”cim-namespace-uri”
            xmlns:md=”cim-model-description_uri”>
      <!-- Content: full-model (definition|description)* -->
</rdf:RDF>

<rdf:RDF xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
            xmlns:cim=”cim-namespace-uri”
            xmlns:md=”cim-model-description_uri”
            xml:base=”power-system-base-uri“>
```

```
      <!-- Content: full-model (definition|description)* -->
</rdf:RDF>
```

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:cim="http://iec.ch/TC57/2004/CIM-schema-cim10#"
   xmlns:md=" http://iec.ch/TC57/61970-552/ModelDescription/1#">

</rdf:RDF>
```

1. The element type is rdf:RDF.
2. The RDF namespace must be declared as http://www.w3.org/1999/02/22-rdf-syntax-ns#
3. The CIM namespace must be declared. With newer versions of the CIM schema the version needs to be adjusted in the CIM name space. Parties exchanging documents have to agree on the used version.
4. Other namespaces may be declared.
5. The xml:base attribute shall be present when the rdf:about URL form is used, refer to section 5.3.

The RDF grammar clause: 6.1

6.2.3.4 FullModel element

```
<md:FullModel rdf:about=model-uri>
      <!-- Content: (literal-property|resource-property
                                          |compound-property)* -->
</md:FullModel >
```

Example:

```
<md:FullModel rdf:about="http://polarenergy.com/2008/case1234/solution">
  <md:Model.created>2008-12-24</md:Model.created>
    <md:Model.Supersedes rdf:resource="http://polarenergy.com/2008/case1233/solution"/>
    <md:Model.DependentOn rdf:resource="http://polarenergy.com/2008/case1234/topology"/>
    <md:Model.version>V32</md:Model.version>
  <md:Model.modelingAuthoritySet>http://polarenergy.com/2008/NorthPoleTSO</md:Model.modelingAuthoritySet>
    <md:Model.description>Santa Claus made a study case peak load summer base topology solution</md:Model.description>
    <md:Model.profile>http://iec.ch/TC57/61970-456/StateVariables/1</md:Model.profile>
</md:FullModel>
```

1. The full model element introduces a new model.
2. The value of the about attribute, model-uri, is a name chosen by the implementation. The model-uri uniquely identifies a document and is the name referenced by other documents, e.g. by Supersedes or DependentOn, as indicated in Figure 2. Hence it must be unique among all documents being exchanged by a number of parties.

6.2.3.5 Definition element

```
<classname rdf:ID=identity>
   <!-- Content:
   (literal-property|resource-property|compound-property)*
   -->
```

```
</classname>


<classname rdf:about=resource-uri>
   <!-- Content:
   (literal-property|resource-property|compound-property)*
   -->
</classname>
```

Example:

```
<cim:SynchronousMachine rdf:about="#_31DFC4296BFB4E2EB299642491B3ABC1">
   <cim:IdentifiedObject.name>IN-2</cim:IdentifiedObject.name>
   <cim:SynchronousMachine.minimumMVAr>-9999</cim:SynchronousMachine.minimumMVAr>
   <cim:SynchronousMachine.operatingMode rdf:resource="http://iec.ch/TC57/2001/CIM-schema-
cim10#SynchronousMachineOperatingMode.generator"/>
   <cim:RegulatingCondEq.RegulationSchedule rdf:resource="#_CA32746FA0024C2BBCF47BC97430BF87"/>
   <cim:Equipment.EquipmentContainer rdf:resource="#_6CB8701A12F14DE99E68125D95073A75"/>
</cim:SynchronousMachine>
```

1. The definition element introduces a new resource and gives its type. There are two forms: the first as an rdf:ID attribute and the second an rdf:about attribute. The rdf:ID attribute has a limited scope as described in section 5.3 and shall not be used.
2. The element type, *classnam*e, is the XML qualified name of a class from the CIM schema or other schema declared as a namespace in the document element.
3. The value of the id attribute, *identit*y, is a XML name symbol chosen by the implementation. It must be unique in the document. (It is not necessarily related to the power system resource name.)
4. The value of the about attribute, *resource-uri,* is a URI-reference that identifies the subject resource. (This can be reduced to the fragment part for identifying resources defined in the present base namespace.) This may also be a UUID URI, refer to 6.4.

6.2.3.6 Description element

```
<rdf:Description rdf:about=resource-uri>
   <!-- Content:
   (literal-property|resource-property|compound-property)*
   -->
</rdf:Description >
```

Example:

```
<rdf:Descriptionrdf:about="#Substation-2">
   <cim:IdentifiedObject.name>TROY</cim:IdentifiedObject.name>
</rdf:Description>
```

1. The description element adds information about a resource introduced elsewhere in this or another document.
2. The *resource-uri* is a URI-reference that identifies the subject resource. (This can be reduced to the fragment part for identifying resources defined in the present base namespace.) This may also be a UUID URI.
3. The Description element is used only in difference models (refer to section 6.2.4). It is never used in full models.

6.2.3.7 Compound element

```
<classname>
   <!-- Content:
   (literal-property|resource-property|compound-property)*
   -->
</classname>
```

Example:

```
<md:Description>
  <md:Description.name>Santa Claus</md:Description.name>
</md:Description>
```

1. The compound element introduces a structured value. The value does not represent a resource nor have any *identity*. It can only appear as the object of a property.
2. The element type, *classnam*e, is the XML qualified name of a compound class.

6.2.3.8 Literal-Property element

```
<propname>
      <!-- Content: text -->
</propname>
```

Example:

```
<cim:SynchronousMachine rdf:ID="_31DFC4296BFB4E2EB299642491B3ABC1">
   <cim:IdentifiedObject.name>IN-2</cim:IdentifiedObject.name>
   <cim:SynchronousMachine.minimumMVAr>-9999</cim:SynchronousMachine.minimumMVAr>
   <cim:SynchronousMachine.operatingMode rdf:resource="http://iec.ch/TC57/2001/CIM-schema-
cim10#SynchronousMachineOperatingMode.generator"/>
   <cim:RegulatingCondEq.RegulationSchedule rdf:resource="#_CA32746FA0024C2BBCF47BC97430BF87"/>
   <cim:Equipment.EquipmentContainer rdf:resource="#_6CB8701A12F14DE99E68125D95073A75"/>
</cim:SynchronousMachine>
```

1. The literal-property element introduces a property and a literal value applying to the enclosing resource.
2. The element type, *propnam*e, is the XML qualified name of a property from the CIM schema or other schema declared as a namespace in the document element.
3. The content *text* is any XML text with <, >, and & escaped representing the value of the property.

6.2.3.9 Compound-Property element

```
<propname>
      <!-- Content: (compound) -->
</propname>
```

Example:

```
<md:Model.Creator>
  <md:Description>
    <md:Description.name>Santa Claus</md:Description.name>
  </md:Description>
</md:Model.Creator>
```

6.2.3.10 Resource-Property element

*<propname rdf:*resource=*resource-uri/>*

1. The resource-property element introduces a property and a resource as its value applying to the enclosing resource.
2. The element type, *propnam*e, is the XML qualified name of a property from the CIM schema or other schema declared as a namespace in the document element.
3. The *resource-uri* is an URI-reference that identifies a resource.

Example 1 - URI-Reference:

The example contains two references one for an RegulationSchedule and the other to the parent represented as EquipmentContainer.

```
<cim:SynchronousMachine rdf:ID="_31DFC4296BFB4E2EB299642491B3ABC1">
   <cim:IdentifiedObject.name>IN-2</cim:IdentifiedObject.name>
   <cim:SynchronousMachine.minimumMVAr>-9999</cim:SynchronousMachine.minimumMVAr>
   <cim:SynchronousMachine.operatingMode rdf:resource="http://iec.ch/TC57/2001/CIM-schema-
cim10#SynchronousMachineOperatingMode.generator"/>
   <cim:RegulatingCondEq.RegulationSchedule rdf:resource="#_CA32746FA0024C2BBCF47BC97430BF87"/>
   <cim:Equipment.EquipmentContainer rdf:resource="#_6CB8701A12F14DE99E68125D95073A75"/>
</cim:SynchronousMachine>
```

Example 2 - Enumeration:

The example defines the attribute value of SynchonousMachine.operatingMode as "generator". The operatingMode is specified in the CIM schema as the enumeration SynchronousMachineOperatingMode.

```
<cim:SynchronousMachine rdf:ID="_31DFC4296BFB4E2EB299642491B3ABC1">
   <cim:IdentifiedObject.name>IN-2</cim:IdentifiedObject.name>
   <cim:SynchronousMachine.minimumMVAr>-9999</cim:SynchronousMachine.minimumMVAr>
   <cim:SynchronousMachine.operatingMode rdf:resource="http://iec.ch/TC57/2001/CIM-schema-
cim10#SynchronousMachineOperatingMode.generator"/>
   <cim:RegulatingCondEq.RegulationSchedule rdf:resource="#_CA32746FA0024C2BBCF47BC97430BF87"/>
   <cim:Equipment.EquipmentContainer rdf:resource="#_6CB8701A12F14DE99E68125D95073A75"/>
</cim:SynchronousMachine>
```

6.2.4 Syntax extension for difference model

6.2.4.1 General

The general syntax definition in the first part of this clause is used for partial and full model data exchange. Once the initial complete set of model data is exchanged, only updates are required to maintain the model as changes occur. In general, those changes can be specified as a set of differences between two models. The difference file is itself an RDF model (a collection of RDF statements) and therefore can be processed by an RDF infrastructure.

6.2.4.2 Example use case (informative)

To illustrate the difference file approach to handling incremental model updates, an example use case is provided. In this example, the participants are Regional Energy Co. and Network Power Co:

- Each participant has a copy of a power system model, B1.
- Regional Energy Co. updates B1, to reflect forthcoming power system modifications, producing B2.
- Regional Energy Co. sends the differences between B1 and B2 to Network Power Co. as a difference model.
- Network Power Co. reviews and validates the difference model.
- Network Power Co. merges the difference model with its copy of model B1, to produce B2.

An alternative would have been for Regional Energy Co. to simply send Network Power Co. a copy of B2. However, B2 is a very large model and it is not feasible to validate it in any reasonable period of time. Validation is not entirely automated, but involves analysis by experts. Indeed, the best validation strategy for B2 may be to compare it to the previously validated B1. This brings us back to the need for a difference model.

A more complicated use case would involve more than two participants. Several peers of Regional Energy Co. would contribute difference models to Network Power Co. This use case would introduce issues of parallel model changes and concurrency conflict.

6.2.4.3 Requirements (informative)

Given two RDF models, B1 and B2, called base models, the requirement is for a difference model that:

- Represents the differences between the two base models
- Is itself an RDF model (a collection of RDF statements) and therefore can be processed by RDF infrastructure.
- Efficiently represents a small difference between two large base models.
- When an object is deleted, the system applying the differences is responsible for performing the "cascading deletions" ,i.e, finding and deleting all other contained objects. Associations with deleted objects should also be deleted.
- Remove operations are not reversible (at least, not from the information in the difference model).
- May contain information about itself such as authorship, purpose and date.
- May contain information to protect against conflicts arising when two difference models are created concurrently from the same base model.

The requirement to treat each difference file as a database commit operation is outside the scope of this service (i.e., a roll back functionality, if desired, is the responsibility of the receiving application, not the sending application). This is in recognition of the fact that the sending application may not be aware of changes made in the B2 model files by other agents since the last update to B1.

6.2.4.4 Structure of difference file

Given two base RDF models, B1 and B2, the difference model is made up of four groups of statements, each encoded as a sequence of resource description structures:

- Header statements, comprising statements about the difference model itself.
- Forward difference statements, comprising statements found in B2, but not in B1.

- Reverse difference statements, comprising statements found in B1, but not in B2.
- Precondition statements, comprising statements found in both B1 and B2 and considered to be dependencies of the difference model in an application defined sense.

Any or all of the four groups can be empty.

The difference model itself is represented by a resource of type dm:DifferenceModel. It is conventional to use the URI of the document itself for this resource.

The following properties apply to the difference model resource:

- dm:forwardDifferences is a property of the difference model whose value is a collection of statements (i.e., resources of type rdf:Statement) representing the forward difference statements.
- dm:reverseDifferences is a property of the difference model whose value is the collection of reverse difference statements.
- dm:preconditions is a property of the difference model whose value is the collection of precondition statements.

Header properties also apply to the difference model resource. These may indicate authorship, date and purpose. These properties can be drawn from the Dublin Core vocabulary or any other convenient schema.

The namespace for the difference model vocabulary, represented by the prefix dm: in the foregoing, is: **http://iec.ch/TC57/61970-552/DifferenceModel/1#**

6.2.4.5 Preconditions and concurrency

The precondition statements are a subset of both B1 and B2 and carry no difference information. In simple, sequential model revision scenarios they can be omitted.

For a large shared model, sequential revision is not always feasible. Revisions are likely to be constructed concurrently by different participants, without reference to each other. Concurrency issues must be handled, but the conventional database-oriented approach of using locks to detect incompatible concurrent transactions is not feasible on a web-scale.

The precondition statements are an alternative to locks. Informally, they represent the information that would have been read-locked in an equivalent database transaction. Software agents that process difference models can check that the preconditions hold and, if not, warn of incompatible model revisions.

The choice of statements to include as preconditions is application-specific (as is the choice of which information to lock in a database transaction). Preconditions should include statements that would the affect decisions of the agent that produced the model revision.

6.2.4.6 Difference model template

The following is a template for the conventional syntax of a difference model.

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
       xmlns:cim="cim-namespace-uri"
        xmlns:md="cim-model-description_uri"
       xmlns:dm="difference-model-namespace-uri"
       xml:base="power-system-base-uri">
       <dm:DifferenceModel rdf:about=model-uri>
```

```
            <!-- Content: (literal-property|resource-property|compound-property)* --
      >
            <dm:preconditions parseType="Statements"
                  xml:base="power-system-base-uri">
                  <!-- Content: (definition|description)* -->
            </dm:preconditions>
            <dm:forwardDifferences parseType="Statements">
                  xml:base="power-system-base-uri">
                  <!-- Content: (definition|description)* -->
            </dm:forwardDifferences>
            <dm:reverseDifferences parseType="Statements">
                  xml:base="power-system-base-uri">
                  <!-- Content: (definition|description)* -->
            </dm:reverseDifferences>
      </dm:DifferenceModel>
</rdf:RDF>
```

Just for clarification with the namespace "dm" new statements are introduced that are valid extensions to the standard RDF syntax through the new property rdf:parseType, which is called Statements.

```
<property parseType="Statements">
      <!-- Content: (definition|description)* -->
</property>
```

The content model of an element with rdf:parseType="Statements" is the same as the content model of the rdf:RDF element.

The content generates the same RDF statements as if it appeared in an rdf:RDF element.

6.2.4.7 Difference model usage

6.2.4.7.1 General

The following cases explain the usage of the difference model.

6.2.4.7.2 Add resource

The difference model contains for a given resource only a ***forward difference*** statement if the particular is resource is added.

Example:

The following example adds two new ACLineSegments each with its adjacent Terminals. The Terminals are linked to new ConnectivityNodes. Those ConnectivityNodes are assigned to a new VoltageLevel in an existing Substation.

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cim="cim-namespace-uri"
xmlns:md="cim-model-description_uri"
xmlns:dm="difference-model-namespace-uri"
xml:base="http://example.com/example-power-system#">
  <dm:DifferenceModel rdf:about="http://polarenergy.com/2008/change22">
<md:Model.created>2008-12-24</md:Model.created>
    <md:Model.Supersedes rdf:resource="http://polarenergy.com/2008/case1234"/>
    <md:Model.DependentOn rdf:resource=" http://polarenergy.com/2008/equip/v3"/>
    <md:Model.version>V32</md:Model.version>
```

```
    <md:Model.modelingAuthoritySet>http://polarenergy.com/2008/NorthPoleTSO</md:Model.modelingAuthoritySet>
    <md:Model.description>Santa Claus made a study case peak load summer base topology solution</md:Model.description>
    <md:Model.profile>http://iec.ch/TC57/61970-452/EquipmentModel/1</md:Model.profile>
  <dm:forwardDifferences rdf:parseType="Statements "
      xml:base="http://polarenergy.com/2008/ident/">
      <!-- Add ACLineSegment ACLine_New1 -->
      <cim:ACLineSegment rdf:ID="ACLine_New1">
        <cim:IdentifiedObject.name>New 1</cim:IdentifiedObject.name>
        <cim:Conductor.r>0.0646</cim:Conductor.r>
        <cim:Conductor.x>0.5961</cim:Conductor.x>
        <cim:Conductor.bch>0.4066</cim:Conductor.bch>
      </cim:ACLineSegment>
      <cim:Terminal rdf:ID="Term_New1">
        <cim:IdentifiedObject.name>T1</cim:IdentifiedObject.name>
        <cim:Terminal.ConnectivityNode rdf:resource="#_9190056CEA54055932C646A4739238C"/>
        <cim:Terminal.ConductingEquipment rdf:resource="#ACLine_New1"/>
      </cim:Terminal>
      <cim:Terminal rdf:ID="Term_New2">
        <cim:IdentifiedObject.name>T2</cim:IdentifiedObject.name>
        <cim:Terminal.ConnectivityNode rdf:resource="#_B3F193FA6E2240C9A603EF03122242D3"/>
        <cim:Terminal.ConductingEquipment rdf:resource="#ACLine_New1"/>
      </cim:Terminal>
      <!-- Add ACLineSegment ACLine_New2 -->
      <cim:ACLineSegment rdf:ID="ACLine_New2">
        <cim:IdentifiedObject.name>New 2</cim:IdentifiedObject.name>
        <cim:Conductor.r>0.0646</cim:Conductor.r>
        <cim:Conductor.x>0.5961</cim:Conductor.x>
        <cim:Conductor.bch>0.4066</cim:Conductor.bch>
      </cim:ACLineSegment>
      <cim:Terminal rdf:ID="Term_New3">
        <cim:IdentifiedObject.name>T1</cim:IdentifiedObject.name>
        <cim:Terminal.ConnectivityNode rdf:resource="#ND_New1"/>
        <cim:Terminal.ConductingEquipment rdf:resource="#ACLine_New2"/>
      </cim:Terminal>
      <cim:ConnectivityNode rdf:ID="ND_New1">
        <cim:IdentifiedObject.name>ND   New1</cim:IdentifiedObject.name>
        <cim:ConnectivityNode.EquipmentContainer rdf:resource="#Volt_New"/>
        <cim:ConnectivityNode.Terminals rdf:resource="#Term_New3"/>
      </cim:ConnectivityNode>
      <cim:Terminal rdf:ID="Term_New4">
        <cim:IdentifiedObject.name>T2</cim:IdentifiedObject.name>
        <cim:Terminal.ConnectivityNode rdf:resource="#ND_New2"/>
        <cim:Terminal.ConductingEquipment rdf:resource="#ACLine_New2"/>
      </cim:Terminal>
      <cim:ConnectivityNode rdf:ID="ND_New2">
        <cim:IdentifiedObject.name>ND   New2</cim:IdentifiedObject.name>
        <cim:ConnectivityNode.EquipmentContainer rdf:resource="#Volt_New"/>
        <cim:ConnectivityNode.Terminals rdf:resource="#Term_New4"/>
      </cim:ConnectivityNode>
      <cim:VoltageLevel rdf:ID="Volt_New">
        <cim:IdentifiedObject.name>230K</cim:IdentifiedObject.name>
        <cim:VoltageLevel.Substation rdf:resource="#_9F0A09.."/>
        <cim:VoltageLevel.BaseVoltage rdf:resource="#_CF8BD1450E264399891F7FE5653D0760"/>
      </cim:VoltageLevel>
    </dm:forwardDifferences>
  </dm:DifferenceModel>
</rdf:RDF>
```

6.2.4.7.3 Delete resource

The difference model contains for a given resource only a ***reverse difference*** statement if the particular resource is deleted.

Cascading deletes are deletes where an object and its child objects (if any) are deleted. In a cascading delete it would be possible to just include the root or parent object in a CIM XML file. The receiver then has to figure out what child objects to delete. To make clear what objects are included in a cascading delete the creator of the CIM XML file shall include all objects as elements in the cascade. Including only the root or parent object is not allowed.

The EquipmentContainer-Equipment relation is a parent-child relation where deletion of an EquipmentContainer shall also result in a deletion of its child Equipment. Other examples of such parent child relations are

- EquipmentContainers also has a parent child relation, e.g. Station-VoltageLevel.
- PowerTransformer and it's TransformerEnds
- ConductingEquipment and its Terminals

The CIM does not currently specify the containment relations. As this information is missing it is up to an implementer to decide which relation is regarded a containment relation. This spoils interoperability. This is the reason to include all objects in a cascaded delete to indicate the sending systems interpretation of containment.

Associations not considered a containment relation are cut from objects that that are in a cascading delete, e.g. if a ConnectivityNode isn't affected by a delete but a ConductingEquipment connected to it is, then the association Terminal-ConductingEquipment is cut. This means that if the reference to a cut object is from an object that stays the reference from that objects shall be removed which means an update of the object that stays.

Delete elements shall have all its property elements included. Reason is that this enables reversing the delete operation and re-creates the object.

Example:

The example below contains the deletion of a PowerTransformer with all resources that are hierarchically subordinated.

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cim="cim-namespace-uri"
xmlns:dm="difference-model-namespace-uri"
xml:base=" http://example.com/example-power-system#">
  <dm:DifferenceModel rdf:about=" http://polarenergy.com/2008/change23">
    <!-- header properties omitted for brevity -->
    <!-- Delete Transformer -->
    <dm:reverseDifferences rdf:parseType="Statements" xml:base=" http://polarenergy.com/2008/ident/">
      <cim:PowerTransformer rdf:ID="_41BB4445675643FA9E5A48B6CD71790E">
        ...all properties of the transformer follows here...
      </cim:PowerTransformer>
      ...all parts of the transformer follows here....
    </dm:reverseDifferences>
  </dm:DifferenceModel>
</rdf:RDF>
```

6.2.4.7.4 Update resource

The difference model contains for a given resource ***forward difference*** and ***reverse difference*** statements if the resource is changed.

Example:

The example below defines the move of the EnergyConsumer from 115k to 230k through the changed link from its Terminal to a different ConnectivityNode.

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:cim="cim-namespace-uri"
  xmlns:dm="difference-model-namespace-uri"
  xml:base="owner">
  <dm:DifferenceModel rdf:about="http://polarenergy.com/2008/change24">
    <!-- header properties omitted for brevity -->

    <!-- Move EnergyConsumer load from 115K to 230K -->
    <dm:forwardDifferences rdf:parseType="Statements" xml:base="" xml:base="http://polarenergy.com/2008/ident/">
      <rdf:Description rdf:about="#_39E4E3051C704DCCA42345E4812DCD07">
        <cim:Terminal.ConnectivityNode rdf:resource="#_612FA147902C4F88BE3F0302B3750B18"/>
      </rdf:Description>
    </dm:forwardDifferences>
    <dm:reverseDifferences rdf:parseType="Statements" xml:base="" xml:base="http://polarenergy.com/2008/ident/">
      <rdf:Description rdf:about="#_39E4E3051C704DCCA42345E4812DCD07">
        <cim:Terminal.ConnectivityNode rdf:resource="#_5D74FC6AB5184A3E9E724827EFD197CF"/>
      </rdf:Description>
    </dm:reverseDifferences>
  </dm:DifferenceModel>
</rdf:RDF>
```

6.3 CIM XML format style guide (Informative)

A useful feature of RDF syntax is that it allows an arbitrary subset of a power system model to be serialized in a document. This is a two edged sword, however. A document produced by one party may not be usable by a second party if it does not contain all the properties expected. Moreover, a document containing a partial model may not be usable if the resource URI's do not agree with other documents.

The following guidelines apply to the content of a CIM XML document and help maximize the range of applications that can use it.

1. Include the likely primary key properties of each resource at the point it is introduced. For example, the *cim:IdentifiedObject.name* and *cim:Equipment.EquipmentContainer* properties are likely to be required properties.

   Reason: a large class of applications will want to load a database with the model data. Many database schemas will require primary key values on insertion.

2. Include single-valued properties rather than their many-valued inverse. For example, use *cim:Equipment.EquipmentContainer* and not *cim:EquipmentContainer.Contains_Equipments*.

   Reason: Because these properties are inverses, a statement predicated on one implies the converse statement predicated on the other. It is less error prone to include only one side and makes editing or transforming the document easier.

3. When encountering many to many relationships, there is usually a primary direction of reference. Include the primary reference rather than their many-valued inverse. For example, use cim:SynchronousMachine.MVArCapabilityCurves and not cim:MVArCapabilityCurve.SynchronousMachines, since the primary relationship is from SynchronousMachine to MVArCapabilityCurve.

   Reason: Same reasons as for item 3 above.

4. When encountering a single-valued relationship with a single value inverse, include either one, but not both. Importing software needs to be designed to handle either direction of reference and infer the inverse.

Reason: Because these properties are inverses, a statement predicated on one implies the converse statement predicated on the other. This is less error prone, and arguably, makes editing or transforming the document easier.

5. Use the XML name conventions when specifying an `rdf:ID`. *That is, "*A **Name** is a token beginning with a letter or one of a few punctuation characters, and continuing with characters as defined in [7]"

   Reason: Follows XML conventions. Though such identifiers may be named from a numbering system, they should start with a letter.

6. Use stable URIs for resources when a model is spread among multiple documents, or when multiple versions of the model are in use. A stable identifier is one that is never reused for another purpose and never changes. A URI that is initially derived from a registered domain name and a random number is one way to minimize accidental reuse. For example: http://example.org/PSR#_2962 might be a stable URI that can appear as rdf:ID="_2962" or rdf:about=#_2962" assuming an XML base declaration of xml:base="http://example.org/PSR#". Another alternative is to use UUID's for identification (described in the next section.) However, identifiers based on property values (such as a name) and sequentially allocated identifiers are both subject to change.

   Reason: this avoids referential integrity errors between documents.

7. Many valued properties, if used, appear as repeated property elements having the same property name.

6.4 Universally Unique Identifiers

UUIDs (Universally Unique IDentifier), also known as GUIDs (Globally Unique IDentifier) can be used to identify resources in such a way that (a) the identifiers can be independently allocated by different authorities and (b) identifiers are stable over time and across documents.

If, in addition, the UUID is embedded in a Uniform Resource Name (URN) then the document can be simplified by the elimination of XML base namespace declarations (xml:base attributes). The URN is a concise, fixed-length, absolute URI.

The standard for an URN containing a UUID is defined by the Internet Engineering Task Force RFC 4122, [Ref]. The following is an example:

```
urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6
```

RFC 4122 specifies the syntax of the URN and how the UUID portion following the last colon is allocated. The algorithm is aligned with, and technically compatible with, IEC 9834-8:2004 Information Technology, "Procedures for the operation of OSI Registration Authorities: Generation and registration of Universally Unique Identifiers (UUIDs) and their use as ASN.1 Object Identifier components" ITU-T Rec. X.667, 2004.

When a URN is used in a Definition Element, section 6.2.3.4, the second template given there applies and the URN replaces the *resource-uri* construction. When used in the Description Element or Resource Property Element, sections 6.2.3.5 and 6.2.3.9, the URN replaces the resource-uri construction in those templates.

A URN can also be used to identify a model. In this case the URN replaces the *model-uri* construct in the FullModel Element, section 6.2.3.3and in the DifferenceModel element, section 6.2.4.6.

6.5 Representing new, deleted and changed objects as CIM XML elements (Normative)

With an object it is meant a system internal representation of the data in a CIM XML file.

The following cases exists for identification of elements and how they appear in full or differential models

- new objects are represented by the definition element (refer to section 6.2.3.4) identified by a rdf:ID attribute in full or differential models.
- deleted objects are represented by the definition element (refer to section 6.2.3.4) identified by a rdf:ID attribute in differential models.
- changed objects are represented by the description element (refer to section 6.2.3.5) identified by a rdf:about attribute in differential models.
- an added property (e.g. internally a null value is changed to a valid value) is a change that appears only in the forward section of a difference model.
- a removed property (e.g. internally a valid value is changed to a null value) is a change that appears only in the backwards section of a difference model.

6.6 CIM RDF schema generation with CIM profile

IEC 61970-501 of this series discusses the generation of CIM RDF Schema. A CIM XML model exchange document uses a subset of the CIM to address the model exchange needs of a specific use case (see Part 4XX, CIM XML Model Exchange Specification). A CIM profile defines that portion of the CIM that an importer and exporter of a CIM XML document should be expected to handle. The RDF Schema for a profile then contains only the classes and properties defined for that profile.

6.7 CIM extensions

The CIM RDF schema can be extended with new classes and attributes by providing a separate namespace. Because a separate namespace is used, the customized CIM XML documents clearly delineate what is CIM standard and what is custom. Several different custom extensions can exist and be clearly identified within the same XML document. When these customized documents are imported to information systems that know nothing about the extensions, the elements with the unknown tags can be simply ignored. The following declaration identifies an extended namespace, *bpa*.

xmlns:bpa="http://www.bpa.gov/schema/cim_extension/2001may"

For example, we can add a non-CIM attribute, *OriginalPO,* to the *breaker* class, as shown below. These customized tags for BPA can be simply ignored if a system import program is not interested in such extensions.

```
<cim:SynchronousMachine rdf:ID="_31DFC4296BFB4E2EB299642491B3ABC1">
   <cim:IdentifiedObject.name>IN-2</cim:IdentifiedObject.name>
   <cim:SynchronousMachine.minimumMVAr>-9999</cim:SynchronousMachine.minimumMVAr>
   <cim:SynchronousMachine.operatingMode rdf:resource="http://iec.ch/TC57/2001/CIM-schema-
cim10#SynchronousMachineOperatingMode.generator"/>
   <bpa:OriginalPO>PO1234378</bpa:OriginalPO>
   <cim:RegulatingCondEq.RegulationSchedule
rdf:resource="#_CA32746FA0024C2BBCF47BC97430BF87"/>
   <cim:Equipment.EquipmentContainer rdf:resource="#_6CB8701A12F14DE99E68125D95073A75"/>
</cim:SynchronousMachine>
```

The RDF schema corresponding to this extension can be added to a separate RDF schema document thereby keeping the CIM RDF schema clearly separate and allowing each to evolve independently.

6.8 RDF simplified syntax design rationale (informative)

The following points explain some of the choices made in the simplified syntax.

1. The literal properties could be represented by property attributes (RDF grammar clause 6.10). This would be more compact. However, property elements were chosen because they are easier to deal with in XSLT expressions. (For example, they can be sorted.) They also make it easier to represent multi-line text.
2. The syntax is flat, with a two-level resource/property structure. More deeply nested structures might be more compact. Moreover, a well-chosen nested structure might permit common queries to be more easily encoded in XSLT expressions. On the other hand, the flat structure was chosen because it is the simplest structure possible and is easy to produce and interpret. By avoiding any application dependency on the details of a nesting structure it should be a more portable syntax.
3. All resources are given a type at the time they are introduced (by the definition element). However, the RDF model allows a resource to be un-typed. In the present application, un-typed resources are not required.

Annex B
Bibliography

1. *XML for CIM Model Exchange*, IEEE PES PICA 2001 Conference Paper, May 2001, A. deVos, S. Widergren, J. Zhu.
2. *Simplified RDF Syntax for Power System Model Exchange*, CIM XML Interoperability Group Paper, 16 November 2000, A. deVos.
3. *Resource Description Framework (RDF) Model and Syntax Specification*, W3C Recommendation 22 February 1999 http://www.w3.org/TR/REC-rdf-syntax, Ora Lassila, Ralph R. Swick.
4. *Resource Description Framework (RDF) Schema Specification*, W3C Proposed Recommendation 03 March 1999 http://www.w3.org/TR/PR-rdf-schema, Dan Brickley, R.V. Guha, Netscape.
5. *Uniform Resource Identifiers (URI): Generic Syntax*; Berners-Lee, Fielding, Masinter, Internet Draft Standard August, 1998; RFC2396.
6. *Namespaces in XML*; Bray, Hollander, Layman eds, W3C Recommendation; http://www.w3.org/TR/1999/REC-xml-names-19990114
7. *Extensible Markup Language 1*.0 (Second Edition), W3C Recommendation 6 October 2000, http://www.w3.org/TR/REC-xml, Bray, Paoli Sperberg-McQueen, Maler.
8. "SiRPAC - Simple RDF Parser & Compiler", http://www.w3.org/RDF/Implementations/SiRPAC
9. "A Strawman Unstriped Syntax for RDF in XML", Tim Berners-Lee, November 1999, http://www.w3.org/DesignIssues/Syntax
10. "Simplified Syntax for RDF", Sergey Melnik, November 1999, http://www-db.stanford.edu/~melnik/rdf/syntax.html