European Network of
Transmission System Operators
for Electricity

RDF-SYNTAX USER GUIDE

2024-04-04

VERSION 1.1.0
CIM WG APPROVED

Copyright notice:

This document is maintained by the ENTSO-E CIM WG. Comments or remarks are to be provided at cim@entsoe.eu

NOTE CONCERNING WORDING USED IN THIS DOCUMENT

The force of the following words is modified by the requirement level of the document in which they are used.

- SHALL: This word, or the terms “REQUIRED” or “MUST”, means that the definition is an absolute requirement of the specification.
- SHALL NOT: This phrase, or the phrase “MUST NOT”, means that the definition is an absolute prohibition of the specification.
- SHOULD: This word, or the adjective “RECOMMENDED”, means that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
- SHOULD NOT: This phrase, or the phrase “NOT RECOMMENDED”, means that there may exist valid reasons in particular circumstances when the particular behaviour is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behaviour described with this label.
- MAY: This word, or the adjective “OPTIONAL”, means that an item is truly optional.

33

Revision History

Version	Date	Paragraph	Comments
0.0.0	2023-10-01		CGMES SG's first draft of the document.
0.0.1	2023-11-08		CIM WG agreement to continue the document and separate the JSON-LD specialisation.
0.0.2	2023-11-15		CGMES SG revision of the document and reset the numbering to 0.0.2. Name of the document changed from *RDF-syntax Data Exchange Specification* to *RDF-SyntaxUsageGuidelines*.
0.0.3	2023-12-18		Integrating additional feedback from CGMES SG
1.0.0	2024-01-17		Integrating additional feedback from CIM WG
1.1.0-alpha	2024-03-20		For CIM WG review. Section 8 was updated, Section 8.4 was added.
1.1.0	2024-04-04		CIM WG approved.

CONTENTS

List of figures

No table of figures entries found.

List of tables

1. Introduction

This document aims at providing technical information and guidance for software developers and power system engineers that are implementing RDF based data exchange using standards such as IEC 61970-600-1:2021, IEC 61970-600-2:2021 (CGMES) or ENTSO-E specifications such as Network Code (NC) Data Exchange Specification.

The document intends to decrease the learning curve for people that are new to software implementations based on RDF technology or are looking for some necessary technical details to explain the reasoning of directions taken.

The information provided in the document that relates to CIMXML and RDFS does not replace or amend requirements and/or statements provided in other approved and published documents and it should be treated as technical guidance only. The disclaimer below which relates to application profiles should be noted.

Disclaimer

The test configurations (models), documents and application profiles are owned by ENTSO-E and are provided by ENTSO-E "as it is". To the fullest extent permitted by law, ENTSO-E shall not be liable for any damages of any kind arising out of the use of the test configurations (models), documents and application profiles (including any of their subsequent modifications).

ENTSO-E neither warrants, nor represents that the use of the test configurations (models), documents and application profiles will not infringe the rights of third parties. Any use of the test configurations (models), documents and application profiles shall include a reference to ENTSO-E. ENTSO-E web site is the only official source of information related to these test configurations (models), documents and application profiles.

2. Provided application profiles

The application profiles are provided to facilitate the implementation of the CGMES profiles and related constraints as defined in IEC 61970-600-1:2021, IEC 61970-600-2:2021, IEC 61970-301 and other related 61970-45x series of profiles.

Note that the application profile serialization based on RDFS and RDF XML syntax is defined in IEC 61970-501:2006 (Ed1) and CIM XML serialization is defined in IEC 61970-552:2016. However, current implementations deviate from these standards due to various reasons addressed in this document.

For CGMES v3.0 the machine-understandable application profiles include the following packaged:

- RDFS2020 e.g., IEC61970-600-2_CGMES_3_0_0_RDFS2020 for CGMES v3.0

A RDFS 2020 update export (see details on the update in the section "Different CIM RDFS versions") of the RDFS augmented version that is based on IEC 61970-501:2006 (Ed1) and used for exporting the RDFS for CGMES v2.4. The only difference (compared with RDFS2019 variant) is resolving export technical issues and the information from the abstract version class that is instantiated as part of the header of the RDFS instead as a version class with all details. No functional changes were made in RDFS2020 compared with RDFS2019. The notation "2020" does not refer to the year of generation, but it is the version of the augmented RDFS export by CimSyntaxGen.

- RDFSEd2Beta, e.g., IEC61970-600-2_CGMES_3_0_0_RDFSEd2Beta for CGMES v3.0

This is a beta version of application profile based on RDFS specified in the draft IEC 61970-501:Ed2. The purpose of inclusion of the beta version in the distribution is to enable review process. Please use these files only for information on the direction where RDFS will evolve in that standard and provide feedback that will be discussed in the standardization process. Namely, the RDFS contains the vocabulary only, while the constraints (cardinalities, datatypes, etc.) are expressed by SHACL based constraints.

- SHACL, e.g., IEC61970-600-2_CGMES_3_0_0_SHACL for CGMES v3.0

This is a package of all SHACL shapes/constraints applicable for CGMES v3.0. These are constraints for cardinalities and datatypes derived from the RDFS, constraints defined in the descriptions of the classes and attributes, constraints defined in IEC 61970-600-1:2021, IEC 61970-600-2:2021, IEC 61970-301 and other related 61970-45x series of profiles and expressed there in plain English text. Note that SHACL based constraints in this folder are serialized in two RDF formats, Turtle and RDF XML plain (no nesting). Originally the constraints were developed in Turtle using Notepad++ as an editor and then converted to RDF XML using CimPal app. Because many constraints rely on SHACL SPARQL method, which is not covered in the draft IEC 61970-501:Ed2, the RDF XML may not represent the desired way of serialization. However, the resulted RDF XML version was not used or validated in terms of content and should be used with a caution.

The recommended serialization of SHACL constraints is Turtle as that was the primary serialization and it is well tested. There is a question in the standardization community if RDF XML will need to be supported as amended or new development will be done in JSON-LD. The tendency is that the JSON-LD will become the main serialization that sematic web tool vendors must support while the other (e.g. Turtle and RDF XML) becomes optional.

- OCL, e.g. IEC61970-600-2_CGMES_3_0_0_OCL for CGMES v3.0

[deprecated, obsolete] This is a package of OCL based constraints that cover CGMES v3.0 in a similar way as SHACL shapes cover necessary validation scope. The RDFS Extracted subfolder contains OCL constraints derived from the RDFS. The XLSX Extracted subfolder contains constraints defined in the descriptions of the classes and attributes, constraints defined in IEC 61970-600-1:2021, IEC 61970-600-2:2021, IEC 61970-301 and other related 61970-45x series of profiles and expressed there in plain English text. However, please note that this package was developed in Nov 2020 and may have deviations compared to the published version of CGMES v3.0. OCL is no longer maintained. Only SHACL constraints will be maintained.

Packaging for other profiles is different:

- CGMES v2.4 - do not include SHACL constraints; RDFS exports are done with earlier versions i.e., not RDFS2020; OCL constraints are provided
- NC profiles – follow the setup as in CGMES v3.0: SHACL constraints are provided both for derived from RDFS constraints and custom SHACL constraints; OCL is not provided; RDFS2020 is exported.

3. Combining different CIM versions

Historically every version of CIM canonical model[1] and related profiles[2] had different URI for the namespace. In addition, some implementations rely on namespace prefix, not on actual namespace URI. This makes it impossible to combine or support mix of versions / provenance in the instance file, which should technology-wise not be a problem (namespace concept serves this). These are reasons that software applications have difficulties in handling or combining data from different CIM version. Consequently, if there is data exchanged that is governed by different CIM versions – each version in own dataset – larger or smaller amount of custom

[1] Canonical model is published in standards like IEC 61970-301 and IEC 61970-302

[2] Profiles are published in standards like IEC 61970-452, 61970-453, 61970-456, 61970-600-1, 61970-600-2, etc.

pre/post-processing would be required where the different versions are compatible with each other, to fit the data for handling with off-the-shelf tools.

In order to support implementations, starting with CIM18, the CIM international standard development community agreed to keep the URI of the canonical CIM stable between different versions of CIM. This means that if a class is defined in CIM vocabulary its URI will not change. Semantic versioning is applied on profile level and different packages in CIM in order to be able to describe and explain CIM evolution.

Starting from CIM18, the following setup is planned:

- Namespace URI is stable
- Each package of the CIM canonical model is versioned with URI. The URI changes only when the package is modified. This allows tracing changes. This also enables a process in which a standard that defined a profile does not need to be updated if CIM version changes and if the profile is depended on canonical packages that have not changed compared to previous version
- Each profile has version URI that changes every time the profile changes.
- Semantic versioning is applied to all canonical model, profiles and all machine-readable artifacts, i.e., if RDFS and SHACL constrains can be updated independently of IEC standard if the standard document is not impacted.

4. Specifics on RDF/CIM-XML-syntax serialization: General differences between CIM XML (552) and RDF XML (W3C)

The CIM XML is defined in the IEC 61970-552:2016. This version of the standard is based on a much earlier edition in which some serialization assumptions were made. Important: When the initial version of IEC 61970-552 was developed, the W3C recommendations on RDF XML were not released. Therefore, there was a growing gap during the last two decades. The latest RDF XML was standardized by W3C in 2014 (RDF 1.1 XML Syntax (w3.org)) and IEC 61970-552 did not align with this due to existing implementations objecting changes in CIM XML.

Many experts complain that RDF is difficult to read due to the references and the flat structure of the file, i.e., no nesting as present in XSD[3]-governed XML. Such complaints should not be addressed to RDF in general, but rather to IEC 61970-552 CIM XML. The W3C RDF XML can be serialized in different forms and many open libraries as Apache Jena support these natively. For instance, the abbreviated version of RDF XML is very much mirroring nested structure of XSD-governed XML.

Another important point to note is that RDF as a general framework is not bound to a given serialization. RDF based dataset can be serialized in different forms – CIM XML (IEC 61970-552), RDF XML (W3C), Turtle (W3C), JSON-LD (W3C), N-Triples (W3C), etc. Each of these different serializations have their advantages and disadvantages. In general Turtle, due to its human readability, is a preferred serialization to provide example datasets when explaining concepts. JSON-LD, which is not the JSON, but a special JSON for linked data, is targeted serialization for the future.

CIM XML, used for CIM based data exchanges, is based on RDF XML serialization specification and restricts it to simplify processing of instance data. However, it also introduces some changes or special assumptions, which have evolved with time and have been difficult to change (due to current implementations for CIM based data exchanges), but which require special pre/post processing when using off-the-shelf RDF tools. Known differences between CIM XML and the RDF XML are listed below:

- CIM XML defined in IEC 61970-552 is unclear regarding the exchange of the datatypes (float, integer, etc.). The approach taken by the community is that datatypes are not exchanged with the instance data assuming receiving party is aware of expected datatypes. Therefore, when consuming data one needs to know the expected datatype from the information in the Schema

[3] W3C XML Schema, used to validate XML instance data. W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures

(RDFS), e.g., in the RDFS we have information that an attribute has ActivePower as a datatype and that value is float, unit W and multiplier M. Note that there are differences in profiling approaches and for example if the profile is generated using CIMTool, a separate profile (separate schema) needs to be prepared for the Domain package.

The example below is from RDFS of EQ profile and illustrates how ActivePower datatype is defined. The property cims:isFixed[4] is used to define that the values for multiplier and unit. The ActivePower.value attribute has cims:dataType property which defines that the datatype is the primitive Float which maps to xsd:float.

```
      <rdf:Description rdf:about="#ActivePower">
    <rdfs:label xml:lang="en">ActivePower</rdfs:label>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Product of RMS value
of the voltage and the RMS value of the in-phase component of the
current.</rdfs:comment>
<cims:stereotype>CIMDatatype</cims:stereotype>
    <cims:belongsToCategory
rdf:resource="http://iec.ch/TC57/ns/CIM/CoreEquipment-
EU#Package_CoreEquipmentProfile"/>
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
     </rdf:Description>
     <rdf:Description rdf:about="#ActivePower.value">
    <cims:stereotype
rdf:resource="http://iec.ch/TC57/NonStandard/UML#attribute"/>
    <rdfs:label xml:lang="en">value</rdfs:label>
    <rdfs:domain rdf:resource="#ActivePower"/>
    <cims:dataType rdf:resource="#Float"/>
    <cims:multiplicity    rdf:resource="http://iec.ch/TC57/1999/rdf-schema-
extensions-19990926#M:0..1" />
    <rdf:type        rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-
ns#Property"/>
     </rdf:Description>
     <rdf:Description rdf:about="#ActivePower.multiplier">
    <cims:stereotype
rdf:resource="http://iec.ch/TC57/NonStandard/UML#attribute"/>
    <rdfs:label xml:lang="en">multiplier</rdfs:label>
    <rdfs:domain rdf:resource="#ActivePower"/>
    <rdfs:range rdf:resource="#UnitMultiplier"/>
    <cims:multiplicity    rdf:resource="http://iec.ch/TC57/1999/rdf-schema-
extensions-19990926#M:0..1" />
    <cims:isFixed
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">M</cims:isFixed>
    <rdf:type        rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-
ns#Property"/>
     </rdf:Description>
     <rdf:Description rdf:about="#ActivePower.unit">
    <cims:stereotype
rdf:resource="http://iec.ch/TC57/NonStandard/UML#attribute"/>
    <rdfs:label xml:lang="en">unit</rdfs:label>
    <rdfs:domain rdf:resource="#ActivePower"/>
    <rdfs:range rdf:resource="#UnitSymbol"/>
    <cims:multiplicity    rdf:resource="http://iec.ch/TC57/1999/rdf-schema-
extensions-19990926#M:0..1" />
    <cims:isFixed
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">W</cims:isFixed>
    <rdf:type        rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-
ns#Property"/>
     </rdf:Description>
```

[4] *cims* is the prefix for namespace <https://iec.ch/TC57/1999/rdf-schema-extensions-19990926#> which is the CIM-specific extension to RDFS.

In accordance with current versions of standards the information of the datatypes is not exchanged and in the instance data, the following will be serialized

```
  <cim:SynchronousMachine rdf:about="#_3a3b27be-b18b-4385-b557-6735d733baf0">
      …
    <cim:RotatingMachine.p>-90</cim:RotatingMachine.p>
   …
  </cim:SynchronousMachine>
```

Looking at the instance data it is not possible to say if the value is kW or MW. It is also not possible to validate if the value is float as a common parser will parse the value as a string. This is why the parser needs to use the information from the RDF in order to assign the expected datatype, e.g. xsd:float and then SHACL validator can validate if the value conforms to the declared datatype for this property.

If the datatype were instantiated it would look like this below. However, this would increase information in the instance file.

```
  <cim:SynchronousMachine rdf:about="#_3a3b27be-b18b-4385-b557-6735d733baf0">
      …
    <cim:RotatingMachine.p
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">-
90</cim:RotatingMachine.p>
   …
  </cim:SynchronousMachine>
```

Still this does not solve the problem with multipliers. Currently the validation of this is part of the conformity process related to CGMES. This is yet another reason why conformity is important.

Starting from CIM18, there is an agreement to change the multipliers in all profiles to “none”. The result of this will be that the value for active power will be exchanged in W. Engineering notation is used to help serializing values with the right precision as shown below.

```
  <cim:SynchronousMachine rdf:about="#_3a3b27be-b18b-4385-b557-6735d733baf0">
      …
    <cim:RotatingMachine.p>-90E6</cim:RotatingMachine.p>
   …
  </cim:SynchronousMachine>
```

- In CIM XML, there is special treatment of rdf:ID vs. rdf:about. In CIM XML:
  - rdf:ID is used for all objects that are serialized first time in the instance data (semantic of “create”), while
  - rdf:about is used when an object (the instance of the class) is updated. This should not be confused with an update of a value of a property. For example, in the equipment (EQ) profile all classes have rdf:ID as the EQ is the base profile, but in Steady State Hypothesis (SSH) all objects are with rdf:about as only attributes are added to existing classes already exchanged in the EQ. In CGMES profiling style such classes are marked with stereotype “Description”.

Rule MVAL5 in IEC 61970-600-1:2021 provides some example of this.

- In CIM XML, there is no declaration of xml:base, which means that parsed data will get local URI if the parser do not impose specific xml:base at the time of the parsing.

If the CIM XML the following and xml:base is not declared:
<cim:ACLineSegment rdf:ID="_ffbabc27-1ccd-4fdc-b037-e341706c8d29">
For example, Apache Jena library will produce this, something comparable is to expect for other standard RDF parsing tools. The identifier of the object is file:\\C:Temp\test.xml#_ffbabc27-1ccd-4fdc-b037-e341706c8d29 if the instance file was located in temp folder in C drive.

If xml:base[5] is declared are the time of parsing and if the base is http://iec.ch/TC57/CIM100 the result is
http://iec.ch/TC57/CIM100#_ffbabc27-1ccd-4fdc-b037-e341706c8d29
Different implementations can import with different xml:base if xml:base is not declared but defined by the implementation at the time of the parsing. Implementations can eventually ignore xml:base declaration, but this is not defined for specific reason and certain base is required in an exchange.

- In the IEC 61970-552 we have the header definition embedded with the serialization instructions, which makes it complex to transition between different versions of the header information. Both CGMES v2.4 and CGMES v3.0 refer to IEC 61970-552 which required to have md:FullModel class as a header. This is why when ENTSO-E had to cover additional requirements and align with W3C DCAT 3, it was necessary to just add W3C DCAT attributes to md:FullModel. In the future, it is expected that the serialization of the header and the rest of the instance file are decoupled, which will allow that instance files are using dcat:Dataset as a header class instead of md:FullModel. However, in order to realize this, CGMES standards need to be updated through the lengthy IEC standardization process.

5. Different CIM RDFS versions

Some information is already provided above in the section that explains the application profiles. Due to historical reasons, the definition of RDF Schema (RDFS) exported for each of the profiles also deviates from W3C. There is a strong influence of profiling techniques used by different CIM communities, e.g. IOP vendors discussions reflected in EA Add-ins, IEC WG13, IEC WG16, etc. There are also cims (CIM scheme) extensions, as defined in IEC 61970-501 standard (in 2006). Finally, and in addition to the above, the RDF Schema used for profiles generated since 2010 do not fully follow IEC 61970-501 either, but the implementation has been industry-driven. In 2019-2020 there was an effort to prepare draft for the next Edition of 61970-501, but due to lack of resources, this work has not been completed yet. Currently CimSyntaxGen[6] supports the following exports:

- RDFS2019 export of profiles relate to RDFS that was used by 2019 (this is industry driven implementation which is not documented in either specification or a standard);
- RDFS2020 export of profiles have change in the Schema header (in 2019 version, the version information is serialized as instances of the version class, while in 2020 version this information is translated to a header). This is also industry driven implementation – there is no approved specification.
- The beta version of RDFS edition 2 is still a work in progress. This version was a prototype of a draft version related to IEC 61970-501 Ed2, which was not finished. The main changes are separation of vocabulary description and constraints as well as aligning with W3C of RDF scheme definitions. The objective was to eliminate the usage of proprietary CIM namespace

[5] this is what in XSD-governed XML one would call namespace URI; “cim” prefix associated with this namespace URI would be the namespace prefix.

[6] The tool used to generate the RDFS of the profiles.

(cims) in the RDFS. This version is available for the purpose of collecting feedback that will be integrated in the final version.

RDFS2020 version is the recommended version to use. It is exported for CGMES v3 profiles, Network Code profiles and metadata and header profiles. CGMES v2.4 used RDFS version which was implemented before RDFS 2019

6. Datatypes and associated issues

In Canonical CIM, datatypes are specified with classes marked with stereotype “Primitive”, “CIMDatatype” and “Compound”. These classes are profiled and assigned to the attributes that use them. In the RDFS derived from profile definition, there is complete information on the datatypes and their multipliers[7]. However, in the CIM XML where data is serialized there is no information on the datatypes. When the data is parsed a standard parser would most probably assume all attributes’ values as strings and if this data is validated, non-string datatype will be reported as invalid. Therefore, when implementing data import, developers need to know this and apply some rules considering the information provided in the RDFS.

The same is valid for the units. In RDFS, there is information if, for example, active power value should be in MW. This information is not explicitly exchanged in the instance data and RDFS needs to be consulted when mapping/converting the data to the internal data model.

Detailed example is provided in Section 4 of this document.

7. Available Tools

There are multiple tools available either for free or under specific license conditions when used in production enterprise environment. Most of them come with some maintenance support.

Profiling tools:

- CimContextor, CimSyntaxgen
- CIMTool

W3C RDF Test Suites:

Two test suites published by the W3C, a W3C RDF Validation Tool Service (https://www.w3.org/RDF/Validator/) that the W3C School on RDF (https://www.w3schools.com/XML/xml_rdf.asp). These test suites are only recommended for non-confidential data for testing purposes.

- RDF 1.0 Test Suite https://www.w3.org/TR/rdf-testcases/
- RDF 1.1 Test Suite https://www.w3.org/TR/2014/NOTE-rdf11-testcases-20140225/#test-suites-and-implementation-reports

RDF Tools and Libraries for data processing and validation:

[7] Note that this is specific to the profiling technique and tooling used. For instance, this is not the case with CIMTool where there is reduction to just corresponding primitive type and info about Datatype is lost. The same is valid for the units.

There are a number of libraries, editors, and databases that are fairly common. This list is not extensive or complete but does provide a cross section of tools that a knowledge engineer or developer might use in conjunction with CIM Profiling tools.

Table 1

Tool	Type	Comments
Protégé v5.5	Ontology editor	
Apache Jena 4.9.0	Java Library	
ValiMate (put link)	SHACL validation	Free GUI version available. Other versions provided under license conditions.
CimPal (open source) –	RDF conversion and some basic manipulations around RDF; generation of SHACL	Java based, using Apache Jena
Pypi rdflib v7.0	Python Library	Can process exactly 1 RDFS, 1 RDFXML and 1 SHACL at a time; not multiple, which is normally needed.
GraphDB Desktop v10.2.1 (Free)	Triplestore	It can be used for testing, but for enterprise use, license is needed.
Blazegraph v2.1.4	Triplestore	
Neptune Serverless 1.2.0.2	Triplestore	
Topbraid Composer 6.0.1	Ontology editor	Not maintained anymore; they moved to a cloud solution. Still, TopBraid SHACL validation code available from GitHub.
Easy RDF	Web service	For testing purpose.
W3C Validation Service	Web service	For testing purpose.
StarDog	Enterprise Knowledge Graph platform	

RDF4J	Java library	
SESAME	Open source RDF database	
PySHACL	Python library on SHACL	
Neo4J/NeoSemantics	Graph Database	

8. SHACL based constraints and validation

SHACL is a W3C recommendation: Shapes Constraint Language (SHACL) (w3.org)

SHACL is a language for validating RDF graphs against a set of conditions. These conditions are provided as shapes and other constructs expressed in the form of an RDF graph. RDF graphs that describe the constraints are called "shapes graphs" in SHACL and the RDF graphs that are validated against a shapes graph are called "data graphs". SHACL standardises the validation reports that are produced by a SHACL validation engine. In addition of data validation SHACL can be used for: interface building, data structure communication, code generation, data integration and rule-based inferencing. The vocabulary of SHACL is inspired by IBM Resource Shapes. The syntax and rules are inspired by SPIN (SPARQL Inferencing Notation) and ShEx (Shape Extensions). The “SHACL and OWL Compared”[8] provides information on the common features between SHACL and OWL.

The shapes graphs can be combined in different ways by using owl:imports to validate a collection of different data graphs e.g. representing instances of core equipment (IEC 61970-452) profile, topology and steady state hypothesis profiles (IEC 61970-456).

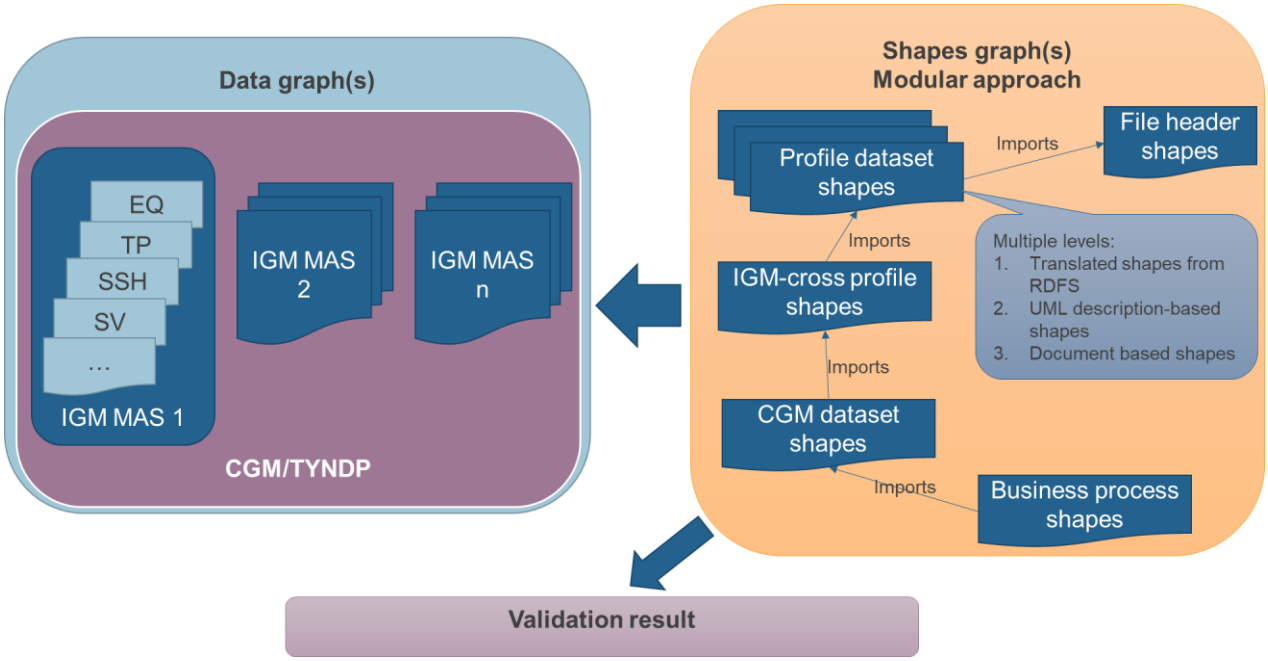
CIMXML, serialised using IEC 61970-552, does not contain information about the datatypes of different attributes. Therefore, the datatypes information shall be available during the instance data validation in order to be able to apply constraints related to validation of datatypes information.

When applying SHACL to CGMES profiles and IEC 61970 profiles in general it should be taken into account that these IEC standards are a paper copy. At the machine-readable side there are UML, XMI, RDFS “views” of the profiles. At the same time currently, there are multiple places where constraints are defined as follows:

- in the UML contains (in descriptions of classes, attributes, association roles) a lot of constraints which are only expressed in English text.
- The IEC 61970-301 and 302 have additional clarifications and constraints.
- Profile documents also contain a lot of constraints.
- IEC 61970-600-1&2 specify CGMES constraints.
- Business processes have a need to further restrict and define business process specific constraints.

Modular approach in validation can be applied by using SHACL. The following chart illustrates an option. SHACL allows use of owl:import which is used by the validation engines to collect necessary sets of shapes when executing a given validation task.

[8] https://spinrdf.org/shacl-and-owl.html



SHACL constraints that are published for CGMES v3.0 include the following types of constraints:

- Constraints derived from RDFS
  - Cardinality of associations and attributes
  - Datatypes
  - Association ends within profile and cross profile
- Constraints developed based on descriptions of classes, attributes and associations
- Constrains derived from the test in the standards such as IEC 61970-301, IEC 61970-452, etc.
- Constraints already numbered in standards like IEC 61970-600-1 and IEC 61970-600-2, etc.

The constraints derived from RDFS are created in an automated way using CimPal. All the rest are pretty much a manual effort. The constraints are maintained under Apache 2 license. There is still work to be done on ensuring the maintenance process is robust enough.

Work on specification to export SHACL constraints from EnterpriseArchitect is planned.

The main serialization for SHACL constraints is Turtle as this was tested in the initial development. Turtle is the most human readable RDF serialization. RDF XML and JSON-LD as possible serialization but RDF XML was not well tested for SHACL and JSON-LD is under development.

8.1. Validation of datasets

It is recommended that SHACL constraints are used for RDF data validation. In order to validate you need to have the instance data, the SHACL constraints and the validation engine. As explained above SHACL constraints already include constraints derived from the scheme and custom constraints. Therefore, the RDFS is not needed for the validation.

8.2. Validation of multiple dependent datasets

Validation of multiple datasets if necessary for most business processes. However, it is not efficient to exchange everything every time and to validate everything every time. It is

recommended that each business process describes a data validation strategy/framework in order to describe what is validated where. This approach provides direct input to the design of the SHACL constraints that can be applied on a portion of data. For example, for the CGMES conformity assessment scheme, DNV as an Assessment Body defined the necessary subset of constraints that are active when validating different use cases and test steps part of the test use cases.

8.3. Tooling used in to create and validate SHACL constraints

The following tooling was used to create the CGMES based constraints:

- For the generation from RDFS to SHACL as well as to generate some of the constraints from Excel to SHACL – CimPal – open-source app.
- For editing - Notepad, Excel
- For the validation and testing – ValiMate

8.4. Design SHACL files (.ttl) for each profile part of CGMES v3.0 based on the information from RDFS

8.4.1. Overview

- The section focuses on the CGMES v3.0 SHACL based constraints. However similar approach is used for preparation of SHACL based constraints for Network Codes Profiles.
- Based on the information from RDFS so called Shapes are produced and forming a Shapes graph. Shapes graphs are produced for all profiles that are part of the CGMES v3. The latest export of RDFS from CGMES UML (FDIS25-iec61970-600-CGMES-v3_0_0.eap) and the latest CimSyntaxGen are used.
- Having the Shapes graphs enables direct validation of instance files (Data graphs). The Shapes graphs can be combined in different ways to validate a collection of different Data graphs e.g. for EQ, TP and SSH. The combination task can be done using metadata specific to an application or owl[9] import approach can be used, which allow for machine readable configuration of the validation scope. The configuration of what should be validated and when is not covered in this document.
- All Shapes graph are validated against the machine-readable version of SHACL so called Shapes graph to validate the Shape graph, i.e. the SHACL constraints that validate SHACL constraints. However, an error was found in the reference Shape graph. W3C support fixed the Shape graph but this is not officially published yet. It is only in GitHub: https://github.com/w3c/data-shapes/files/4021871/shacl-shacl.ttl.txt
- As CIMXML, serialised using 61970-552, does not contain information about the datatypes of different attributes when an instance file is imported in an off the shelf RDF tool/API normally the default datatype assigned at the time of the import is a string. This prevents applying any meaningful Shapes to validate datatypes as there will be many false/inaccurate results of the validation. To overcome this problem, the application that imports CIMXML shall ensure that datatypes are properly mapped with the Data graph, i.e. the parser needs to enrich the data graph based on the information on datatypes provided in the profile (RDFS), see section 4.

8.4.2. List of constraints present in the Shapes graphs

The following is a general summary.

- For an attribute:
  - Shapes on multiplicity: in cases where the attribute has multiplicity 1..1. Note that for instance the file header uses 1..* or 0..* for some attributes and this is also reflected in the designed shapes
  - Shapes on datatypes:

[9] OWL ia W3C Web Ontology Language. Link: OWL 2 Web Ontology Language Document Overview (Second Edition) (w3.org)

- In cases where the datatype is a Primitive, the shape validates if the datatype of the instance data has the right type, e.g. Float, String, Boolean, etc. The shapes use the respective xsd datatypes. It also checks if the object of the triple is a literal.
- In cases where the datatype is a CIMDatatype, the shape validates if the datatype of the instance data has the right type which is defined as a datatype of .value attribute of the CIMDatatype. Other properties like multiplier or unit cannot be subject to validation as there is not enough information exchanged in the instance data. It also checks if the object of the triple is a literal.
- In cases where the datatype is an Enumeration, the shape validates if the datatype of the instance data is one of the attributes/enumerated values of the Enumeration. It also checks if the object of the triple is a IRI (Internationalized Resource Identifier).
- In cases where the datatype is a Compound, the shape validates if the datatype of the instance data is a Compound which is defined as a blank node in the triple. According to 61970-552 compounds do not have identity. Nested compounds are also validated including the multiplicity and datatypes of the attributes of the compound. The validation principles for the attributes of the compound follow the same pattern as for a “normal” attribute. A sequence path mechanism in SHACL is used for pointing to the location of the different attributes of the compound or nested compounds.

- For an association:
  - Shapes on multiplicity: both lower bound and upper bound are checked, i.e. shapes follow the if the multiplicity is 0..1 (i.e. max 1) or 1..* (i.e. min 1) or 1..1 (i.e. exactly 1) or 0..2 (i.e. max 2), etc. depending on what is specified in the profile.
  - Shapes on the value type: it validates if the value type is an IRI. By definition this is a requirement for all associations. It also checks the value type if it is an instance of the class that is defined in the profile among all possible classes considering the inheritance structure.

8.4.3. Applied conventions

As in SHACL different shapes need to be uniquely identified the following conventions are applied:

- The prefix of the shapes for a given profile is the short name (abbreviation) of the profile.
- The subject of a shape is the name of a class, attribute, association but in the namespace of the Shape graph, e.g. eq:ACLineSegment.
- If the shape is validating cardinality the subject is amended by “-cardinality”, e.g. eq:ACLineSegment.bch-cardinality or eq:Equipment.EquipmentContainer-cardinality for an association.
- If the shape is validating datatype the subject is amended by “-datatype”, e.g. eq:ACLineSegment.r-datatype
- If the shape is for validating association values type the subject is amended by “-valueType”, e.g. eq:DCTerminal.DCConductingEquipmentDCSwitch-valueType
- Different shapes are grouped. Three groups are defined:
  - CardinalityGroup – for all shapes related to cardinality validation.
  - DatatypesGroup – for all shapes related to datatypes validation.
  - AssociationsGroup– for all shapes related to associations validation.