An Alternative CIM Modeling Approach using JSON-LD

Authors:
Milan Gavrić, Kosa Nenadić

Presented by:
Milan Gavrić, Ph.D.
milan.gavric@schneider-electric-dms.com
Head of Systems Integration Department
Content

- Typical CIM Modeling Approach
- RDF Serialization Formats
- JSON-LD
- Proposed CIM Modeling Approach
- ENTSO-E IOP Use Case
Integrations Need Standards

- „Standards are great. Everyone should have one.“ – Bob Metcalfe

- Research project, in cooperation with the Faculty of Technical Sciences, Novi Sad, Serbia
Integrating Utility Systems

Message Oriented Middleware
CIM compliant messages and models
Two Main Aspects of CIM Applicability

- Network Model Exchange
  - RDF based modeling approach
  - Standard-based, well defined, IOP tests performed on yearly basis
  - Different serialization approaches, mostly CIMXML
  - Introducing of CIM specifics in RDFS
  - Limited validation capabilities
  - Siloed approach – model parts
  - Limited standardized reasoning capabilities
  - Focus on Full Models

- Message Exchange
  - SOAP based messages
  - Standard-based, well defined, but issues with IOP tests
  - XML payload with defined XSDs
Typical CIM Modeling Approach

- RDFS - schema specification language expressed using RDF used to govern information exchanged within a specific business exchange context.

- CIMXML serialization format (subset of RDF)
  - Both machine and human readable, primarily intended for programmatic access
  - Can be accessed by any tool that supports the DOM
  - Self-describing
  - Takes advantage of current W3C recommendations.

- CIM Datasets (Instance Files) that are Model Part content normally contain ‘dangling references’.

- RDFS maintains CIM and Extension namespaces

---

Typical CIM Modeling Approach
ENTSO-E CIM Modeling Approach

- Assembled Power System Model = Combined instance files from different models
  - MAS files EQ, SSH, TP, DY, GL grouped in .zip
  - Boundary MAS files
- OCL used for handling defined constraints

![Diagram showing the relationship between ENTSo-E Extensions and CIM profiles with references using Model.profile #URI and Model.DependentOn #URI for EQ and SSH models. MAS files are zipped.]
**ENTSO-E CIM Modeling Approach**

- Profiles enable definitions of
  - different cardinalities
  - different model extensions
- Model Parts assume referenced object existence
- Enriching the knowledge about the referenced instances

<table>
<thead>
<tr>
<th>IdentifiedObject</th>
<th>EQ profile</th>
<th>TP profile</th>
<th>SSH profile</th>
<th>SV profile</th>
<th>DL profile</th>
<th>GL profile</th>
<th>DY profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>.name</td>
<td>R</td>
<td>O</td>
<td>O</td>
<td>R</td>
<td>R</td>
<td>x</td>
<td>O</td>
</tr>
<tr>
<td>.description</td>
<td>O</td>
<td>O</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>O</td>
</tr>
<tr>
<td>.energyIdentCodeEic</td>
<td>O</td>
<td>O</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>.shortName</td>
<td>O</td>
<td>O</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>
Dangling Reference Types

- There are two types of dangling references:
  - An object of one class may reference an object of another class, as allowed by associations defined in the CIM UML.
  - An object may exist in one Model Part, and another Model Part may provide additional information about the object.
Independent CIM Profiles Modeling

- Each CIM Profile has its own set of classes and properties, e.g. each CIM Profile has defined a separate IdentifiedObject class in the EA.
## Mapping CIM RDFS to OWL

<table>
<thead>
<tr>
<th>CIM RDF Schema</th>
<th>CIM OWL 2 Ontology</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIM RDF Schema</td>
<td>owl:Ontology</td>
</tr>
<tr>
<td>cims:profile;</td>
<td>rdfs:isDefinedBy</td>
</tr>
<tr>
<td>rdfs:Class with Primitive stereotype</td>
<td>XSD type</td>
</tr>
<tr>
<td>rdfs:Class - CIMDatatype stereotype</td>
<td>XSD type, owl:Class</td>
</tr>
<tr>
<td>rdfs:Class enumeration stereotype</td>
<td>owl:Class</td>
</tr>
<tr>
<td>Enumeration Element</td>
<td>owl:NamedIndividual</td>
</tr>
<tr>
<td>rdfs:Class</td>
<td>owl:Class</td>
</tr>
<tr>
<td>rdfs:Property</td>
<td>owl:DatatypeProperty, owl:ObjectProperty</td>
</tr>
<tr>
<td>cims:multiplicity</td>
<td>owl:qualifiedCardinality,</td>
</tr>
<tr>
<td></td>
<td>owl:maxQualifiedCardinality,</td>
</tr>
<tr>
<td></td>
<td>owl:minQualifiedCardinality</td>
</tr>
<tr>
<td>cims:inverseRoleName</td>
<td>owl:inverseOf</td>
</tr>
<tr>
<td>cims:datatype</td>
<td>rdfs:range of owl:DatatypeProperty or</td>
</tr>
<tr>
<td></td>
<td>owl:ObjectProperty</td>
</tr>
<tr>
<td>cims:isAggregate</td>
<td>meta:isAggregate (owl:AnnotationProperty);</td>
</tr>
<tr>
<td>cims:stereotype</td>
<td>meta:stereotype (owl:AnnotationProperty);</td>
</tr>
<tr>
<td>cims:ClassCategory</td>
<td>owl:Class (Meta ontology)</td>
</tr>
<tr>
<td>cims:belongsToCategory</td>
<td>meta:belongsToCategory (owl:AnnotationProperty);</td>
</tr>
<tr>
<td>cims:isFixed*</td>
<td>meta:isFixed (owl:AnnotationProperty);</td>
</tr>
<tr>
<td>cims:associationUsed*</td>
<td>meta:associationUsed (owl:AnnotationProperty);</td>
</tr>
</tbody>
</table>
RDF1.1 Serialization Formats

- Grouping RDF statements in multiple graphs
- Associating named graph with an IRI
- First introduced in SPARQL

![Diagram of Multiple Graphs Support]

- TriG
- N-Quads
- JSON-LD
- Turtle
- N-Triples
JavaScript Object Notation for Linked Data (JSON-LD)

- a lightweight Linked Data format that extends JSON:
  - easy for humans to read and write
  - provides a way to help JSON data interoperate at Web-scale
- JSON-LD 1.0 is a W3C Recommendation published at 16th Jan 2014
- Current version JSON-LD 1.1
- Defines algorithms for transformation of JSON-LD documents:
  - expansion,
  - compaction,
  - flattening,
  - and RDF serialization/deserialization.
Shapes Constraint Language (SHACL)

- SHACL is a W3C Recommendation published at 20\textsuperscript{th} July 2017
- Used for RDF graph validation
- SHACL is divided into two parts:
  - SHACL Core
  - SHACL-SPARQL
Proposed Modeling Approach

- CIM UML
- Extensions UML
  - restricts
- Profiles RDFS
  - uses
- CIMXML

- CIM OWL 2
  - uses
  - ontology mapping
- Extensions OWL 2
  - semantic mapping
- Profiles OWL 2
  - uses
- CIM JSON-LD
  - uses
  - validated/reshaped

- Meta Ontology OWL 2
  - uses
- Model Description Ontology OWL 2
  - uses
  - SHACL shape graphs
Proposed Modeling Approach

- A comprehensive common CIM ontology is created, including Model, Meta and Extension ontologies using RDFS and OWL2 concepts.
- Standard XML Schema datatypes are used.
- Each CIM Profile ontology is mapped to the common CIM ontology, and potentially to the Extension ontology using owl:equivalentClass and owl:equivalentProperty.
- Each CIM Profile ontology has its own namespace.
- Each ontology/namespace is dereferenceable.
- CIM data documents and contained resources are dereferenceable.
- JSON-LD syntax is utilized.
- The goal is formally and semantically interlinked data.
Proposed Approach on CGMIES

- CIM UML
  - extends
- CGMES UML
  - restricts
- CGMES Profiles RDFS
  - uses
  - validated by
- CGMES OCL
  - validated by
- CGMES Mini Grid CIMXML
- CGMES Profiles
  - uses
- CGMES Extensions OWL2
  - conversion and semantic mapping
  - uses
- CIM OWL2
  - uses
  - restricts
  - restricts
- Meta Ontology
  - uses
  - restricts
- Model Description Ontology
  - uses
  - restricts
- CGMES Profiles OWL2
  - uses
  - restricts
  - restricts
- CGMES Profiles OWL 2
  - uses
  - restricts
  - restricts
- CGMES SHACL shape graphs
  - validates/reshapes
- CGMES Mini Grid JSON-LD
  - uses
  - restricts
  - restricts
Proposed Modeling Approach
ENTSO-E DL and EQ Profiles
Proposed Modeling Approach
Example Profiles in JSON-LD

Test Configurations from the Conformity Assessment Scheme v2.0

CommonCIM.owl.jsonld

```
{  "@id": "co:IdentifiedObject",
  "@type": "owl:Class",
  "me:belongsToCategory": {  "@id": "me:Package_Core"
  },
  "me: stereotype": "abstract",
  "rdfs:comment": "This is a root class to provide common identification ...",
  "rdfs:isDefinedBy": { "@id": "http://www.utility.com/CommonCIM" },
  "rdfs:label": { "@language": "en", "@value": "IdentifiedObject" },
  "rdfs:subClassOf": [  { "@id": "_:N20498595194f4ebbb39689de7423c983" },
   { "@id": "_:Na4c4e690f46443fe0001b9af1dfedc" },
   { "@id": "_:Nca9e4426b4c4c22a0f1ee100f5392e35" },
   { "@id": "_:Nd47c895b16f4128811ab7053e879b0" }
  ]
}...
```

DiagramLayoutProfile.owl.jsonld

```
{  "@id": "dl:IdentifiedObject",
  "@type": "owl:Class",
  "me:belongsToCategory": { "@id": "me:Package_Core" },
  "rdfs:comment": "This is a root class to provide common identification ...",
  "rdfs:isDefinedBy": { "@id": "http://www.utility.com/DiagramLayoutProfile" },
  "rdfs:label": { "@language": "en", "@value": "IdentifiedObject" },
  "rdfs:subClassOf": [  { "@id": "_:Nc6f32b1d2db64552b08bc9001879ae" },  // dl:IdentifiedObject.DiagramObjects  { "@id": "_:N0a195b34e4e6942447e9a2a3796" },  // dl:IdentifiedObject.name  { "@id": "_N7f342f29e628486895349695db8986" }  // dl:IdentifiedObject.mRID
  ],
  "owl:equivalentClass": {  "@id": "co:IdentifiedObject"
  }
}...
```

Meta

```
{  "@id": "http://www.utility.com/meta#Package_LoadModel",
  "@type": [  "http://www.utility.com/meta#ClassCategory",
   "owl:NamedIndividual" ],
  "rdfs:comment": { "@language": "en", "@value": "Package LoadModel" },
  "rdfs:label": { "@language": "en", "@value": "LoadModel" }
}...
```

EquipmentProfileCoreShortCircuitOperation.owl.jsonld

```
{  "@id": "eq:IdentifiedObject",
  "@type": "owl:Class",
  "me:belongsToCategory": { "@id": "me:Package_Core" },
  "rdfs:comment": "This is a root class to provide common identification ...",
  "rdfs:isDefinedBy": { "@id": "http://www.utility.com/EquipmentProfileCoreShortCircuitOperation" },
  "rdfs:label": { "@language": "en", "@value": "IdentifiedObject" },
  "rdfs:subClassOf": [  { "@id": "_N5146c688556a08492b7981cecf19e52" },
   { "@id": "_Nf6be9c788e04470904d6d924305c2e" },
   { "@id": "_Nc6a76150a4e4f3b766ab0c53ee31e" },
   { "@id": "_N76a05a6617274960baaaf3c5a10306c" }
  ],
  "owl:equivalentClass": { "@id": "co:IdentifiedObject" }
}...
```
Proposed Modeling Approach
Example Instance Files in JSON-LD

Mini Grid Test Configuration

```
MiniGridTestConfiguration_BC_DL_v3.0.0.jsonld
...
{  
  "@id": "http://www.utility.com/model#_000a02c6-3b66-f8e5-2d87-381bd275fb1f",
  "@type": "dl:DiagramObject",
  "dl:DiagramObject.Diagram": {  
    "@id": "http://www.utility.com/model#_e410e806-f110-3450-b721-4a38b5d75314"
  },
  "dl:DiagramObject.IdentifiedObject": {  
    "@id": "http://www.utility.com/model#_bb2136b7-d699-4b34-9b30-d9047d534974"
  },
  "dl:IdentifiedObject.name": "DiagramObject114"
},
...
```

```
MiniGridTestConfiguration_BC_EQ_v3.0.0.jsonld
...
{  
  "@id": "http://www.utility.com/model#_01a240e9-5607-4844-9d53-5c8b08b5c9a8",
  "@type": "eq:Terminal",
  "eq:ACDCTerminal.sequenceNumber": {  
    "@type": "xsd:integer",
    "@value": "3"
  },
  "eq:IdentifiedObject.name": "T3_2",
  "eq:Terminal.ConnectingEquipment": {  
    "@id": "http://www.utility.com/model#_5d38b7ed-73fd-405a-9cdb-78425e003773"
  },
  "eq:Terminal.ConnectivityNode": {  
    "@id": "http://www.utility.com/model#_0ccf291a-eadf-4010-a26c-f691a135118f"
  },
  "eq:Terminal.phases": {  
    "@id": "eq:PhaseCode.ABC"
  }
},
...
```
Reaching 5-star Linked Data

★ CIM data are available on the utility Intranet and possibly partially available on the Internet for integration with 3rd parties

★★ CIM data are available as machine readable structured data – e.g. a utility can provide CIM data in a proprietary format understandable by internal applications

★★★ CIM data are available in non-proprietary format – standard RDF/XML and the proposed JSON-LD serialization

★★★★ W3C open standards are used to identify things – by using the proposed JSON-LD approach each CIM resource is identified with an IRI

★★★★★ CIM data are linked – JSON-LD is used to link related models and ontologies.
Proposed Approach - Pros and Cons

Pros

- Linked Data - solution aligned with W3C recommendations
- Standard tools are applicable, including SHACL validation
- OWL concepts are applied
  - Reasoning capabilities are increased
- Separate namespace for each profile
- Available standard algorithms – flattening, expanding, compacting
- Available framing capabilities
- Easier model maintenance – referencing CIM model

Cons

- Large JSON-LD files containing multiple instance files (model parts)
- Needed application for the conversion to JSON-LD
- Not proved in practice
- Needed additional research to investigate the capabilities for large files handling by the available applications
Thank You