OWL/RDF Representation of SysML-V2

V 0.2, 9/26/2024

This is a summary of the choices made for the OWL & OSLC-RDF representation of SysML-V2. This represents the “rough consensus” agreed to in the sysml-v2-rdf meeting and the OSLC meeting, both on 9/26/2024. These choices will be finalized in the meeting on 10/4/2024 – all comments should be submitted to the Slack channel prior to that meeting.

# OWL Baseline

The OWL will be generated from the SysML-V2 metamodel XMI. Initial implementation will be done using OpenCeasar but other implementations of the mapping are possible.

As a baseline, we reference the preliminary OWL generation done by Magad in OpenCeasar: <https://github.com/opencaesar/sandbox/blob/master/sysml2/owl/www.omg.org/spec/SysML.owl>

We will use this baseline to define the mapping pattern with three changes defined below.

# Changes to baseline

## Map UML data types to OWL-XSD data types

The reference to UML will be removed and all datatypes will be mapped to the XSD equivalent. Specific map to be provided by Magad.

## Ordering of multi-valued properties.

SysML specifies ordering of properties with a multiplicity>1 using the UML:isOrdered property. RDF triples are unordered by definition so some pattern needs to be used to capture ordering.

The REST API uses JSON, which is naturally ordered. This ordering will be used to specify the ordering of the RDF properties when using the API.

Several patterns were considered:

* Use of RDF list using rfd:first & rdf:next
* Use of RDF collections using a {predicate}\_index pattern
* Use of triple reification with an ordering property, this is the pattern chosen

We will define a tiny foundation ontology “Ordering” with a property and an annotation.

* Ordering:isOrdered, range Boolean, an annotation of “owl:class” with semantics as defined for UML isOrdered
* Ordering:order, range Integer, with a domain of Statement, that specifies the index of each reified triple

For each ordered property value, each triple, there will be an rdf:Statement that has a unique URI (not a blank node). The rdf:Statement will reify the list triple and have a Ordering:order property*. It is the responsibility of any software that updates a SysML model to assert these triples as a part of the graph*. The implementation of the API must retain ordering for get and put.

The method to create the URI for these statements is not specified and should not be used for any computation.

**Example:**

:r a SysML:Reltionship .

:e a SysML:Element .

#This is the actual triple for the SysML:source property of a relationship

:r SysML:source :e .

# The following orders it with the addition of an "order" property.

:s\_1 a rdf:Statement ;

rdf:subject :r ;

rdf:predicate SysML:Element ;

rdf:object :e ;

Ordering:order 1 .

**Advantages of this pattern include:**

* The OWL and RDFS definitions of SysML can use “normal” domains and ranges
* It is easy to query
* A read-user of a model may choose to respect or ignore ordering
* The pattern is compatible with OSLC and can be implemented for SysML, a specific extension to provide for ordering in general will be considered by the OASIS team.

# Pattern for property name conflicts

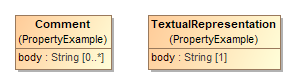
UML & MOF define property names scoped by a class whereas RDF & OWL scope property names by the ontology. Due to this, there has to be some algorithmic way to handle properties with the same name in the same ontology.

Analysis of the 59 name conflicts in SysML indicates that there is a consistent general concept associated with each – that is, the same term is not used for fundamentally different concepts. This indicates that some unification is possible. A unification to simple property names would make the terms consistent with the API JSON, simplifying the mapping between OWL and the API.

The patterns considered were:

* Prepend the class name to each property name. This makes them unique but odd from an OWL/RDF perspective. E.g. each query and update would have to know which variant of the property URI to use.
  + E.g. the “body” property would become “comment\_body & textualRepresentation\_body”
* Create a property using just the simple property label (e.g. “body”) without a domain and range that is restricted (in UML) or constrained (in OSLC shapes) within the context of each class that uses it. Note this is the pattern used by FIBO.
* Create a property using just the simple property label that uses a owl:Union for each case where the domain or range is different. Use restrictions (or OSLC constraints) to tighten range or cardinality as required. This is the pattern implemented by Magicdraw-CCM and the choice made.

Model example:



Note the cardinality was made different to illustrate the concept. The following is the OWL (generated by CCM):

<!-- https://www.omg.org/spec/SysML#body -->

<owl:DatatypeProperty rdf:about="https://www.omg.org/spec/SysML#body">

<rdfs:domain>

<owl:Class>

<owl:unionOf rdf:parseType="Collection">

<rdf:Description rdf:about="https://www.omg.org/spec/SysML#Comment"/>

<rdf:Description rdf:about="https://www.omg.org/spec/SysML#TextualRepresentation"/>

</owl:unionOf>

</owl:Class>

</rdfs:domain>

<rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>

<terms:description rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Definition of body in comment</terms:description>

<terms:description rdf:datatype="http://www.w3.org/2001/XMLSchema#string">definition of body in textual representation</terms:description>

<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">body</rdfs:label>

</owl:DatatypeProperty>

<!-- https://www.omg.org/spec/SysML#Comment -->

<owl:Class rdf:about="https://www.omg.org/spec/SysML#Comment">

<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Comment</rdfs:label>

</owl:Class>

<!-- https://www.omg.org/spec/SysML#TextualRepresentation -->

<owl:Class rdf:about="https://www.omg.org/spec/SysML#TextualRepresentation">

<rdfs:subClassOf>

<owl:Restriction>

<owl:onProperty rdf:resource="https://www.omg.org/spec/SysML#body"/>

<owl:qualifiedCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:qualifiedCardinality>

<owl:onDataRange rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>

</owl:Restriction>

</rdfs:subClassOf>

<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">TextualRepresentation</rdfs:label>

</owl:Class>

</rdf:RDF>

# OSLC Specification

An OSLC specification contains multiple artifacts; two of particular interest are the “vocabulary” and “shapes.” The vocabulary is specified in RDFS using OSLC-specific rules; these rules were not followed for the current artifacts, and as such, they are unusable.

The updated OSLC artifacts will provide a different (RDFS) representation of the OWL Model using the same URI, but using the OSLC-RDFS syntax. The Ordering:isOrdered property will be added to RDFS classes which will assert that the ordered triple reification pattern must be utilized and asserted by the OSLC implementation on a put. A OSLC get will return the ordering triples.

## OSLC Specification publication

The current SysML OSLC artifacts do not comply with OSLC rules .Either these artifacts will be deprecated in favor of ones generated using the above patterns and published by Oasis or will be updated based on the OSLC generation. In either case, the full OSLC server specification will be published by Oasis. *The specifications can not be in conflict*. A candidate generation will be provided by Jim Amsden. As the OSLC issues are better understood, this issue may be revisited by the FTF.

## OSLC URI Considerations

It is highly desirable for all RDF representations of SysML (OWL, RDFS & OSLC) to use the same URI for the same model element. As a condition for OSLC to use the same URI as OWL, some constraints on SysML are required:

* The Vocabulary URI should not be version-specific (As is done in the draft OWL)
* The fundamental semantics of a Vocabulary URI should not change (but may be restricted)
* A vocabulary URI may be deprecated but not deleted
* The URI should be dereferenceable

If these conditions can not be agreed to by the FTF & OMG, the URI will be different and a owl:equivelentProperty or owl:equivelentClass will be used to relate the two URI.

# Please submit any comments so we can finalize the OWL and OSLC mappings.