Synthetic Environment Representational Semantics Using the Web Ontology Language

Mehul Bhatt[1], Wenny Rahayu[1], and Gerald Sterling[2]

[1] Department of Computer Science
La Trobe University
Melbourne, Australia 3086
+61-3-94791280
{mbhatt,wenny}@cs.latrobe.edu.au
[2] Air-Operations Division, DSTO
PO Box 4331 Melbourne
Australia 3001
+61-3-96267728
Gerald.Sterling@dsto.defence.gov.au

Abstract. The application of Ontologies for the definition and interoperability of complementary taxonomies has been well-recognised within the Modelling & Simulation (M&S) community. Our research pertaining to the specification of *Synthetic Environment* (SE) representational semantics has proposed the use of an ***S**ynthetic **E**nvironment **D**ata Representation **Onto**logy* (***sed*Onto**), which is modeled using W3C's ***W**eb **O**ntology **L**anguage*(OWL). The vocabulary specified in *sed*Onto is based the SEDRIS Data Representation Model (DRM), which is a technological framework for SE data interchange and interoperability.
In this paper, we present STOWL – ***SEDRIS To OWL** Transform* that automates the transformation of a SEDRIS based SE to a Web-Ontology based representation scheme in the OWL language. The target representation scheme, which shall be based on *sed*Onto, is in actuality an instantiation of the SE data representation terminology as specified by *sedOnto*. Such a transformation has many perceived advantages: It enhances SE interoperability by utilizing a Web-Ontology based approach for the specification of SE representation data, is consistent with existing industry based SE representation standards, namely SEDRIS, and that the representation scheme facilitates ontological reasoning over SE objects; a facility that is not directly supported by the SEDRIS DRM.

1 Introduction

The application of Ontologies for solving interoperability problems has been widely recognised across multiple domains. Ontologies, by virtue of the shared conceptualization that they provide, may be communicated between people and application systems thereby facilitating interchange, interoperability and common understanding. An ontology typically consists of a hierarchical description of important concepts in a domain, along with descriptions of the properties of each concept. The degree of formality employed in capturing these descriptions can be quite variable, ranging from natural language to logical formalisms, but increased formality and regularity clearly facilitates

M. Gallagher, J. Hogan, and F. Maire (Eds.): IDEAL 2005, LNCS 3578, pp. 9–16, 2005.

machine understanding [1]. Ontologies are increasingly being applied in the Modelling & Simulation (M&S) domain, with the e***X***tensible ***M***odelling and ***S***imulation initiative (XMSF) recommending the use of ontologies to allow the definition and approval of complementary taxonomies that can be applied across multiple XMSF application domains. As specified in the XMSF charter, this would involve the use of such XML based technologies such as XML Schema, RDF, OWL etc. [2]

In this paper, we propose the use of Ontological formalisms as the basis of Synthetic Environment (SE) representational semantics. The work reported herein is in continuum with our previous research pertaining to the construction of a SE representation ontology called *sed*Onto [3]. *sed*Onto is based on a ISO/IEC standard, namely SEDRIS, which is a technological framework for the successful *representation* and *interchange* of environmental data sets. In this paper, we propose and implement a necessary extension to *sed*Onto called ***STOWL*** – ***SEDRIS TO OWL*** *Transform*, which is the automation of the transformation of a SEDRIS based SE to a OWL ontology based form. More precisely, the resulting OWL representation scheme shall be based on *sed*Onto and will consist of instance data relevant to the vocabulary defined in it. Such a transformation has many perceived advantages: (a) Utilisation of the OWL/RDF (XML) serialisation syntax enables web-based sharing of SE data semantics thereby contributing toward the XMSF goal of web-enabled simulation systems. (b) Since the representation scheme is based on a ISO/IEC standard, it is practically applicable in industrial settings such as Defence and/or Environmental simulation systems where SEDRIS is mostly used. (c) Most importantly and in line with our envisaged application, existing OWL based reasoners may be applied so as to perform ontological reasoning in the SE domain.

2 *sed*Onto: A Synthetic Environment Data Representation Ontology

sed*Onto** – ***S*ynthetic* ***E****nvironment* ***D****ata Representation* ***Onto****logy* [3] is an ontology to be used within the M&S domain for the representation of data pertaining to a SE. We leverage existing standards for SE representation by '*web-enabling*' the SEDRIS Data Representation Model (DRM), which is widely adopted within the M&S community for the representation of SE data. The DRM is an object-oriented model, and provides a unified method for describing all data elements, and their logical relationships, needed to express environmental data in a seamless manner across all environmental domains. *sed*Onto is represented using the the ***W****eb* ***O****ntology* ***L****anguage* [4]. More specifically, we utilize the OWL DL subclass of the OWL language for the representation of *sed*Onto; driven by the fact that tool builders have already developed powerful reasoning systems that support ontologies constrained by the restrictions required by OWL DL, the best example here being RACER [5]. It must be emphasized that *sed*Onto formalizes the same terminology for SE representation as is specified in the SEDRIS DRM. Whereas the SEDRIS DRM is a UML based specification of the various SE representation classes (and their relationships), *sed*Onto is a mapping of the same in the OWL language.

Fig. 1, an extract from sedOnto, consists of OWL statements necessary for the definition of the DRM class *Model*. Models within the DRM are used to represent some generic environmental entity that can be referenced many times in a *transmittal* (a SE

```
<owl:Class rdf:about="#Model">
 <owl:Restriction>
      <owl:onProperty>
        <owl:FunctionalProperty rdf:ID="hasDynamicModelProcessing"/>
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
 </rdfs:subClassOf>
<owl:Class rdf:about="#SEDRIS_DRM_CLASS"/>
 </rdfs:subClassOf>
 <rdfs:comment rdf:datatype ="http://www.w3.org/2001/XMLSchema#string"
 >SEDRIS DRM Ref: Sheet 2</rdfs:comment>
 <rdfs:subClassOf>
   <owl:Restriction>
     <owl:maxCardinality rdf:datatype ="http://www.w3.org/2001/XMLSchema#int"
     >1</owl:maxCardinality>
     <owl:onProperty>
       <owl:ObjectProperty rdf:about="#hasClassificationData"/>
     </owl:onProperty>
   </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>
```

Fig. 1. Definition For *Class Model* in *sed*Onto

database) to create many instances of representations of similar environmental entities [6]. The class definition in Fig. 1 makes a number of important assertions such as: (a) *Model* has a certain attribute (*datatype property*), (b) *Model* is a subclass of another class (*subsumption relationship*), (c) *Model* aggregates objects of other classes (*aggregation relationship*) etc. Note that not all properties, both datatype or object, have been specified in the Model class definition in Fig. 1. The actual definition in *sed*Onto for a *Model* is too large to be included in its entirety in Fig. 1. For an in depth coverage of *sed*Onto, we direct interested readers to [3], which presents the *sed*Onto construction methodology along with potential applications of our proposed approach, namely – Terminological reasoning over SE objects and Web-based Sharing of SE transmittal semantics.

3 *STOWL*: Sedris to OWL Transform

In this section, we present the design and implementation of ***STOWL*** – ***S***EDRIS ***To OWL*** Transform, which is the automation of the transformation of a SEDRIS based SE or SEDRIS transmittal to a Web-Ontology based form. The resulting OWL based representing scheme will be based on *sed*Onto and in actuality shall be an instantiation of it. Specifically, *sed*Onto represents the 'Terminology' or TBOX whereas the automatically transformed SEDRIS transmittal represents the 'Assertions' or ABOX[1]. To make things clear, the precise situation is illustrated in Fig. 2. Note that although in different forms, the shaded boxes in Fig. 2 represent the same terminology for SE representation.

[1] The formal semantics of OWL are based on Description Logic (DL), which distinguishes between an ontology (the TBox) and instance data (the ABox) relevant to the ontology

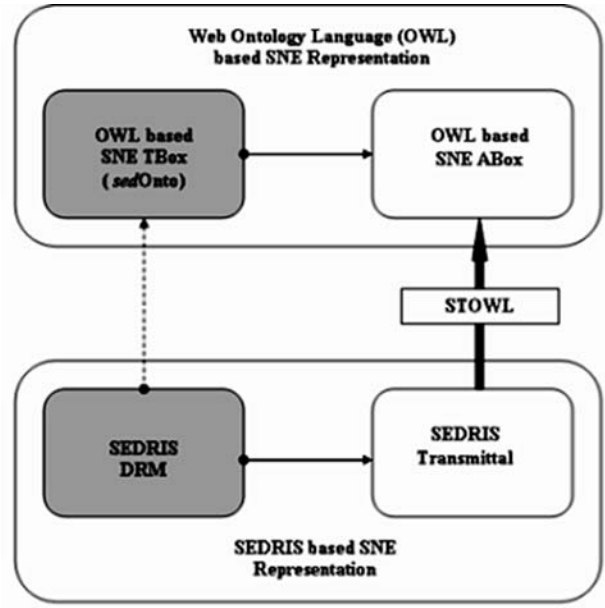

Fig. 2. *sed*Onto & *STOWL* – A Unified View

3.1 Restricted Views from the DRM

The DRM is massive in that it encompasses every structural element likely to be used for the representation of a SE pertaining to any domain. Indeed, applications with differing requirements would be interested in different aspects of a SE transmittal. For instance, an application with a task to reason about the topology or connectedness of the various objects present in the transmittal would be interested in the *FeatureTopologyHierarchy* present in it whereas one concerned with visualisation of those objects in the *GeometryHierarchy*.

STOWL uses the concept of a *Restricted View* so as to extract and transform the relevant information of interest. This is achieved by the specification of a DRM class factory that maintains a repository of the various DRM classes currently within the required view. With this setup, the actual transformer simply performs a depth-first traversal of the DRM class hierarchy whilst delegating object extraction and initialisation to the DRM class factory, which conditionally performs the necessary extraction and initialisation.

3.2 A Transformation Walkthrough

Providing a Web-Ontology based view of a SEDRIS DRM based transmittal is the essence of STOWL. In Fig. 2, it can be seen that the input to STOWL consists of a SEDRIS transmittal, which is semantically coupled to the SEDRIS DRM, whereas its output is a OWL document consisting of instance data for the terminology defined in *sed*Onto. In this section, we present a illustrative walkthrough of the transformation

process for sample transmittal data. We utilize the definition for *class Model* from sedOnto, previously discussed in section 2 (see Fig. 1).

STOWL Input. Various SE transmittals freely available from [6] have been used for testing STOWL. For the purposes of this walkthrough, we use one such transmittal, namely *anywhere_ruby.stf*. Whilst the details being unimportant here, it must be added *anywhere_ruby.stf* is a fictitious Model for a town square that could exist anywhere.

STOWL Output Extract. The output of the transformation process is a valid OWL/RDF document expressed using the XML serialization syntax. It basically consists of two inseparable parts – the *Instance Ontology Template*, which is the document preamble consisting of the ontology definition and *Instance Data*, which consist of the actual instance data for the terminology present in *sed*Onto. Fig. 3 consist of an extract from the output generated by STOWL for input *anywhere_ruby.stf*.

Instance Ontology Template. All output instance data is associated to a OWL ontology model. The instance ontology template is the generic specification for such a ontology. Loosely speaking, the Instance Ontology Template consists of the standard namespace declarations required in any OWL ontology and a *OWL:imports* statement asserting the fact that the instance ontology in question imports the vocabulary defined in *sed*Onto. The namespace declarations and the ontology itself is required to be embedded inside a *RDF element*[2]. Other instance data (such as the one in Fig. 3) would follow the namespace declarations and *import directive*.

Instance Data (ABox). The OWL extract in Fig. 3 consists of instance data for the *class Model* defined in *sed*Onto (see Fig. 1). The Model instance, which corresponds to one of the models present in the transmittal *anywhere_ruby.stf*, makes the following assertions: (a) **hasGeometryModel**: The Model has a *GeometryModel* instance (given by the relative URI *"#instance_GeometryModel_57404368"*)associated with it. Note that *hasGeometry* is a *subproperty* of a another object property called *hasComponent* thereby giving it the intended interpretation of a *aggregation relationship*, i.e., Model aggregates objects of class GeometryModel. (b) **hasDynamicModelProcessing**: This Model represents something that can move within the environment defined by the transmittal in which it is present. (c) **hasClassificationData**: This Model aggregates an instance of the class *ClassificationData* (given by the relative URI *"#instance_ClassificationData_57405136"*). Instances of this class are used within the source transmittal to provide *thing-level* semantics for the Models being represented. In this case, it can be seen in Fig. 3 that the ClassificationData associated to this Model has an attribute (given by the *hasEDCSClassification* relationship) that assign a EDCS[3] code of *145* to this Model. Within SEDRIS, this code has been defined to be a building; using the symbolic constant *ECC_BUILDING*.

[2] Since every valid OWL document has a valid RDF model

[3] The Environmental Data Coding Specification (EDCS) provides a mechanism to specify the environmental "*things*" that a particular data model construct (from the DRM) is intended to represent

```
<sedOnto:Model rdf:about="#instance_Model_57122976">
  <sedOnto:hasMovingParts rdf:datatype="XMLSchema#boolean"
  >false</sedOnto:hasMovingParts>
  <sedOnto:hasName rdf:datatype="XMLSchema#string"
  >/apartment_bldg</sedOnto:hasName>
  <sedOnto:hasGeometryModel>
    <sedOnto:GeometryModel
    rdf:about="#instance_GeometryModel_57404368"/>
  </sedOnto:hasGeometryModel>
  <sedOnto:hasUnits rdf:datatype="XMLSchema#boolean"
  >true</sedOnto:hasUnits>
  <sedOnto:hasDynamicModelProcessing rdf:datatype="XMLSchema#boolean"
  >false</sedOnto:hasDynamicModelProcessing>
  <sedOnto:hasClassificationData>
    <sedOnto:ClassificationData
    rdf:about="#instance_ClassificationData_57405136">
      <sedOnto:hasEDCSClassificationCode rdf:datatype="XMLSchema#int"
      >145</sedOnto:hasEDCSClassificationCode>
    </sedOnto:ClassificationData>
  </sedOnto:hasClassificationData>
  <sedOnto:hasModelReferenceType rdf:resource="#SE_MDL_REF_TYP_ROOT"/>
</sedOnto:Model>
```

Fig. 3. *Model* Extract: Apartment Building

Unique Instance Names. As can be seen in Fig. 3, the Model instance itself and every object related to Model through a object property has a resource name that is unique. This is necessary because most instance objects are related to many other objects through various relationships. By having unique instance names, such references can be resolved to the same instance object instead of having to reproduce instance data multiple times under different heads (ie., URI's). Such instance names are generated by concatenating the string *"instance_$DRM_CLASS_NAME_"* with a unique *Sort ID* that is maintained by the SEDRIS implementation for every object present in the transmittal. Uniqueness and absence of redundant data is therefore guaranteed.

Data Completeness and Validation. Within the scope of the *restricted view* that STOWL is working on, the resulting OWL instance data generated by it is *Complete*. This means that every *defining element* of a certain DRM *Class* – all its attributes and relationships with other classes in the DRM – is transformed into the target representation scheme. The transformation is complete so that a two way transform would in principle be possible. The only exception to this is a scenario in which one of the classes defining elements (say its attribute or another component object) lies outside of the restricted view. Validation here refers to the process of performing the following three types of tests on the transformed model (and *sed*Onto): (a) **Maintenance Tests**: Check whether or not facet (property) constraints are being maintained. (b) **OWL DL Tests**: Perform OWL DL language tests to determine whether or not all OWL language elements in use belong to its DL class so as to qualify the resulting ontology as a OWL DL one. (c) **Sanity Tests**: Check the integrity of the ontology by performing tests such as whether

or not redundant classes have been used in the range of a property, domain of a sub-property has only narrowed the one in its super-property etc. Note that all the three tests are being performed through the Protege Ontology development environment in use by the Protege-OWL plugin.

3.3 Design Overview

STOWL Phases. The design overview for STOWL is shown Fig. 4. STOWL basically involves the use of SEDRIS and Semantic Web based technologies. Implementation has been done using both C++ and Java, with integration involving the use of Java Native Interface. The following are the important components that make up STOWL: (a) **Transmittal Import**: Involves import of the synthetic environment represented in the SEDRIS Transmittal Format (STF) using the SEDRIS read API. The import layer constructs a object-oriented view, similar to the DRM, of the imported transmittal.(b) ***sed*Onto Import**: Import our SE representation ontology, sedOnto, using the Jena 2 Ontology API. (c) **JNI Bridge**: Data imported from a transmittal is provided to the transformer as input with the JNI bridge acting as link between the two. (d) **OWL Transform**: This refers to the actual transformation engine. Starting at the root of the DRM class hierarchy, this involves a depth-first traversal of the input transmittal. (e) **Instance Ontology Export**: This again involves use of the Jena Ontology API, albeit not directly, for serialisation of the transformed model.

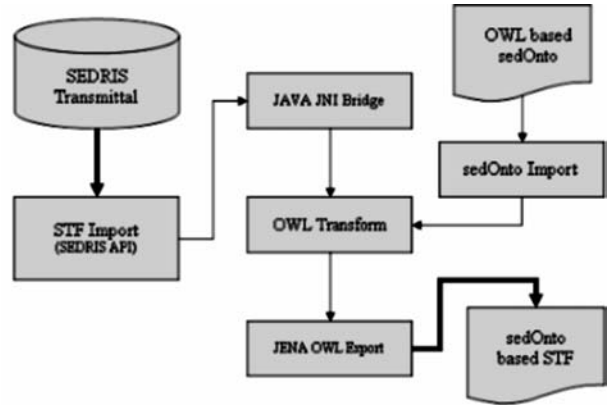

Fig. 4. STOWL – Design Overview

Implementation Details. For the construction of *sed*Onto, we have utilised version 3.1 of the SEDRIS DRM and *Protege*, which is a open-source development environment for ontologies and knowledge based systems[4]. We utilize the C++ based SEDRIS SDK (release 3.1.2) [6] for importing SEDRIS transmittals. For purposes of importing sedOnto and exporting the transformed transmittal to the OWL serialization syntax, we

[4] *Protege*: http://protege.stanford.edu

have utilized the JENA 2.1 Ontology API [5]. The export layer also utilizes Kazuki[6], which is a library for generating an object oriented interface for instance objects from an OWL Ontology file.

4 Conclusion and Further Work

We propose a novel approach involving the use of ontological primitives for the specification of Synthetic Environment Representational Semantics. A prototype, namely STOWL, has been implemented to automate the creation of the desired representation scheme. The paper also presented the design and implementation details for STOWL alongwith a illustrative walkthrough of the transformation.

The use of a industry based standard (SEDRIS) as the basis of our SE ontology makes our approach practically applicable in industrial settings such as Defence and/or Environmental Simulation systems where SEDRIS is generally used. Moreover, *sed*Onto and STOWL are also in line with the broader research goals within the Modeling & Simulation community for the development of Web-Enabled Simulation systems (XMSF). By mapping the SEDRIS DRM to the OWL language, we make explicit the SE representational semantics of the DRM using a language, which unlike UML is inherently suitable to do so. The logical basis of the language means that automated reasoning procedures can be utilized to perform ontological reasoning over SE objects – *subsumption, satisfiability, equivalence, retrieval* [7] etc. Currently, work pertaining to the applications of *sed*Onto and STOWL, viz Web based sharing of SE representational semantics and Terminological reasoning over SE objects [3], is in progress. We are extending a description logic based reasoner, namely RACER [5], so as to be able to provide synthetic environment specific query answering capabilities.

References

1. Horrocks, I.: DAML+OIL: A Reasonable Ontology Language. In: Proceedings of EDBT-02, Volume 2287 of LNCS. (2002) 2–13
2. Brutzman, D., Zyda, M., Pullen, M., Morse, K.: XMSF 2002 Findings and Recommendations Report: Technical challenges workshop and strategic opportunities symposium. Technical Report: XMSF Basis (2002)
3. Bhatt, M., Rahayu, W., Sterling, G.: sedOnto: A Web Enabled Ontology for Synthetic Environment Representation Based on the SEDRIS Specification. In: Proceedings of the Fall Simulation Interoperability Workshop. (2004)
4. W3C: OWL Web Ontology Language Guide. http://www.w3.org/TR/owl-guide/ (2004)
5. RACER: (Renamed ABox and Concept Expression Reasoner) http://www.sts.tu-harburg.de/ r.f.moeller/racer/.
6. SEDRIS: (The Source for Environmental Representation and Interchange) http://www.sedris.org.
7. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F.: The description logic handbook: theory, implementation, and applications. Cambridge University Press (2003)

[5] *JENA Ontology API*: http://www.hpl.hp.com/semweb/jena.htm
[6] *Kazuki*: http://projects.semwebcentral.org/projects/kazuki/