

Richer Semantic Representations of Simulations Using DAML/OWL

Arsalan Tavakoli

David Chu

Paul F. Reynolds, Jr.

University of Virginia

Department of Computer Science

151 Engineer's Way, P.O. Box 400740

Charlottesville, VA, 22904

(434) 982 – 2200

Arsalan@virginia.edu, davidchu@virginia.edu, Reynolds@virginia.edu

Keywords:

OMT, DAML, HLA

ABSTRACT: *A detailed capture of model semantics is essential to the correct functioning of federations of simulations, as well as to reuse, composability and multi-resolution modeling. The difficulties often encountered by practitioners, and the unsatisfactory results that ensue, generally arise from an incomplete capture and therefore an incomplete association of two or more models. The DoD High Level Architecture offers Object Model Templates as a means of capturing model and federation semantics. However, model semantics are not generally fully captured in the OMT. The OMT does not capture semantic interoperability as well as it captures syntactic interoperability. Often, semantics are left to natural language descriptions. We have investigated a method for capturing semantic information that can later be processed by a machine.*

We have applied the DARPA DAML technology to the simulation semantics problem. DAML is a markup language based on XML/RDF and, unlike XML alone, can be used to automatically draw logical inferences from user-defined objects and relationships. We plan to translate our work into the W3C OWL, a successor to DAML, that improves on DAML and is fast becoming an industry standard. We have developed an ontology for modeling and simulation that is based on the HLA/OMT, and which can serve as a powerful tool for future development of Simulation and Federation Object Models.. We have mapped the nine table types occurring in the OMT into a richer DAML description, one which provides for richer semantic description, often removes ambiguity, and provides for automated processing. As a result, practitioners can expect an improved opportunity to build meaningful federations, compose useful suites of models and link multi-resolution models meaningfully.

Mapping from the OMT to DAML required the derivation of semantic descriptions for capabilities tacitly defined in the OMT. For example the OMT does not capture relationships among objects identified in different tables. Our solution abstracts objects, apart from tables, so that they do not lose their uniqueness when placed into the tables. This abstraction supports identification of relationships among objects, and system level properties. It supports automated identification and analysis of these relationships and properties, thus creating greater likelihood that federated or composed models are linked in meaningful ways. We have applied our technology to the Restaurant Simulation Object Model in the OMT Version 1516.2. In that example, we exploited the class-like structure that occurs in our solution technique, to create an object structure that better revealed how objects related to each other, and a richer description of properties. This richer description environment greatly enhances opportunities to form sound inferences and to perform mechanical processing of object descriptions.

Future work includes the construction of tools for automated processing of simulation descriptions constructed using our methodology. Quite possibly this construction will occur in the DAML/OWL community, since the tool needed does not need to be specific to the M&S community.

1. Introduction

An original goal of the High Level Architecture (HLA) was to create a standard which would serve as the foundation for creating an environment of interoperability among simulations. However, by many measures the HLA has fallen short of achieving that goal. There has been much criticism in the form of calls for a new standard or a complete restructuring of the HLA. One paper likened the trend towards obsolescence of HLA to the same path followed by Ada years earlier, [1], while another claims that the current usage of OMT's in the HLA is not at all what was intended by initial authors of the HLA [2]. However, we believe that the majority of problems lie with the method in which simulation descriptions, or object models are created. The HLA Object Model Template (OMT) relies heavily on natural language, and allows too much individual user interpretation, thereby effectively eliminating the opportunity to create a standard for object models, a necessary requirement for future simulation interoperability.

In our work we have converted the object models from the OMT format into DAML, a semantically richer and more machine readable language. Although an XML based version of the OMT exists [3], it still is restrained by the limitations of XML. Although it improves on the natural language specification by including some semantics, it still fails to provide machine understandability, mainly through its inability to draw inferences based on its own specification. Through the use of a more semantic and structured language, the amount of user discretion is greatly reduced, and the specification is far more specific. Since the element of variance has been reduced, the goal is to create a standard that will allow users to create simulations that will be assured to work with ones created in the future. We have converted example SOM's from OMT to DAML, created a schema for future conversions, and we have created a template for designing new SOM/FOM's in DAML.

The scope of our project focuses strictly on the conversion of object models, and does not currently address the ramifications that this may have on the Runtime Interface (RTI) and other HLA components. Coupled with the fact that our object model implementations in DAML would not currently work with any existing HLA-compliant implementations, quantitative measures of the effectiveness of our work are not currently achievable. Rather, we present qualitative arguments that our implementation is an improvement over the current format by analyzing the OMT to discover its weaknesses, and then explain how the DAML version is an improvement in those areas.

2. Object Model Templates

The High Level Architecture is composed of many components, each of which is integral to its success. One component which is arguably the most important in achieving interoperability is Object Models. "The HLA requires that federations and individual federates be described by an object model that identifies the data exchanged at runtime to achieve federation objectives" [3]. Therefore it is critical that the representation of these object models be clear and unambiguous so as to facilitate a standard for creating future simulations.

The current form of representation of these object models is in a format known as Object Model Templates (OMTs). The definition of OMTs can be divided into two parts: theory and representation. Many authors who criticize the OMT consider it to be sound in theory but flawed in representation. In theory the OMT is composed of 9 different tables, ranging from an 'Object Model Identification Table' to a 'FOM/SOM lexicon'. Full details can be found in the [IEEE OMT 1516.2 Specification](#) [3]. These tables focus on the external interface of the federate or federation, as the underlying implementation is not important, only how it will interact with other simulations. Therefore, items such as shared attributes and time representation format which are needed by other simulations in order to work together are provided.

The [IEEE OMT 1516.2 Specification](#) [3] identifies XML as its formal method of implementing object model templates. In the appendixes the complete schema and an example are provided for developers. However, this specification is too general because it doesn't clearly define all relationships between items in various tables. Also it leaves too much freedom to the individual creator of the OMT, hence making adherence to one standard difficult. In addition, XML is a poor choice for the OMT representation owing to limitations already mentioned in the previous section, and mainly owing to its failure to provide machine understandability. When multiple simulations are run simultaneously, they interface through the Runtime Interface component of the HLA. If the RTI is able to infer relationships based on the SOM's, it can provide extra useful information to the other simulations. This ability is key in fostering greater simulation interoperability.

3. DAML/OWL

3.1 DAML

The DARPA Agent Markup Language (DAML) was an initiative funded by DARPA that started in 2000 with the

goal of developing “a language and tools to facilitate the concept of the Semantic Web.” [4] The Semantic Web Project refers to facilitating interoperability between multiple World Wide Web agents in order to simplify common tasks. When developers set out to implement the Semantic Web, they discovered that no existing standalone language had all the qualities and tools necessary to achieve this goal. The Extensible Markup Language (XML) was designed by the World Wide Web Consortium (W3C) and allows “information to be accurately described using tags” [4], yet it is still limited in its ability to describe relationships between various objects. The Resource Description Framework (RDF) was also designed by the W3C for representing information on the web. While XML focuses on providing interoperability within one application using the given schema, RDF provides interoperability across applications. RDF started as a framework for metadata; providing interoperability between applications that exchange machine-understandable information on the Web. It emphasizes facilities to enable automated processing of Web resources and as such provides the basic building blocks for supporting the Semantic Web. [5]. Seeing desirable elements in both languages, DAML was designed as an extension to both XML and RDF and boasts the unique ability to draw inferences based solely upon its own language specification. For example, if a particular schema states that Parenthood is a more general property than Motherhood, and it was provided that Mary is John’s mother, then a query asking ‘Who are John’s parents?’ will return Mary as DAML is able to identify the relationship, even though it is not explicitly stated. [6]. This ability provides a crucial tool for simulation interoperability as it allows the determination of information not explicitly stated in the SOM and that could previously only be inferred by humans.

The core of the DAML language is DAML+OIL, which “provides a rich set of constructs with which to create ontologies and markup information so that it is machine readable and understandable” [4]. DAML+OIL was used to design DAML-S, which is a set of ontologies designed to specifically facilitate the idea of the Semantic Web. We used the tools provided by DAML+OIL to create a set of ontologies designed for simulations, specifically for building SOMs and FOMs. This included all relationships and properties that we felt were necessary and critical to good simulation descriptions in the object models.

3.2 OWL

A recent successor to DAML is beginning to replace DAML. The Ontology Web Language (OWL) is a ‘revision to the DAML+OIL web ontology language incorporating lessons learned from the design and

application of DAML+OIL.” [7]. The first version of the OWL specification was made available in August 2003. Although OWL was built by improving upon DAML, there is no major revision of the syntax. We do not expect that porting our set of ontologies, designed with DAML, to OWL will be a challenge.

4. Translating the OMT to DAML/OWL

Apart from the selection of DAML/OWL as a candidate replacement for the OMT, we also identified goals and assumptions to serve as guidelines. A major assumption limited the scope of our project to the Object Model Templates, and did not extend to other HLA components, such as the Run Time Interface (RTI). We recognize that many features implemented in the DAML Object Models would require a new RTI interface specification to handle them. Second, although we believed that the current Object Model Template design was vague and left much room for improvement, we attempted to follow the current specification as closely as possible in order to preserve the original HLA principles. Addition of future semantic definitions to our new specification is left for future work.

Our major goal in converting the OMT was to remove ambiguity by adding semantic descriptions, therefore creating a standard that would foster interoperability of simulations. Our method was to initially convert a Simulation Object Model described using the OMT format into the new DAML Object Model format. Our plan was that along the way we would face many challenges and have to make many decisions in order to be able to accurately and fully translate the description. After the conversion of a few examples, we would then be able to create our final product, a schema that would serve as a basis for converting existing OMT models and creating new DAML models from scratch.

4.1 Initial Example Conversion

The initial SOM that we translated as an example was the Restaurant example similarly used by the IEEE 1516.2 OMT Specification [3]. This example demonstrates various features of the OMT format, and so fully converting this example without sacrificing any information could transitively demonstrate that the DAML Object Model format was a viable candidate for replacing the OMT format without a loss of functionality. The conversion presented two major problems/decisions regarding the choice of a method of implementation.

Tables such as the object class structure and the interactions table required the ability to describe hierarchical relationships. Even though the OMT specification laid out these relationships in a table, the

actual OMT implementation failed to characterize any relationships between classes. However, with DAML we were afforded the ability to define certain properties and ontologies in our schema that would allow for relationships that would be machine understandable, as well as human readable. We created a ‘subClassOf’ property (See Figure 4.1) which effectively enabled an object to specify its parent, and also allows a future DAML parser to be able to create a hierarchical tree based on the relationships. The idea of using ontologies to describe relationships is critical to utilizing DAML’s ability to draw inferences.

```

<Object rdf:ID="Employee"/>
<Object rdf:ID="Greeter">
  <rdfs:subClassOf rdf:resource="#Employee" />
</Object>
<Object rdf:ID="Waiter">
  <rdfs:subClassOf rdf:resource="#Employee" />
</Object>

<Interaction
  rdf:ID="Customer_Employee_Transactions
  "/>

<Interaction rdf:ID="Customer_Seated">
  <rdfs:subPropertyOf
    rdf:resource=
    "#Customer_Employee_Transactions" />
</Interaction>

<Interaction rdf:ID="Order_Taken">
  <rdfs:subPropertyOf
    rdf:resource=
    "#Customer_Employee_Transactions" />
</Interaction>

```

Figure 4.1. Subclasses and subproperties.

In the current OMT format, objects are instantiated as flat two dimensional objects and therefore although an object may appear in multiple tables, the various instances are not linked in any way. In the XML-based OMT version, an attempt has been made at linking objects with their attributes. However, this approach is combining the attribute and object class structure tables, not actually linking the tables while keeping them as two separate entities. Furthermore, the limitations of XML limit the relationship to one that is only machine readable and further relationships can not be developed using only the XML specification itself. This limitation was shown in the ParentOf example previously mentioned. Finally, this linkage is restricted to just the object class structure – attribute relationship. Objects are not linked to interactions, and interactions are not linked to their parameters, again creating disjoint tables. The approach that we took was to create a three dimensional object that

would be linked to various tables, instead of creating an instance for each table. For example, every attribute that is listed in the attribute table must be linked back to an object in the object class structure table. We created an Attribute class for all the attributes listed in the attribute table. We then used an object property, one of DAML’s built-in constructs, which allows two objects to be linked. For example, we created a ‘hasAttribute’ property (See Figure 4.2) whose domain would be an object in the object class structure table and its range would have to be an attribute. This way, no attribute can be instantiated without being linked to an object. Again, this method has the advantage of being easy for a human to understand as well as being machine understandable, a critical step towards automating interactions across simulations.

```

<Attribute rdf:ID="Home_Address">
  <hasDataType rdf:resource=
  "myschema#Address"/>
</Attribute>

<Object rdf:about="#Employee">
  <hasAttribute rdf:resource="#Pay_Rate"/>
  <hasAttribute rdf:resource="#Home_Address"/>
</Object>

```

Figure 4.2. Linking attributes and objects.

One of our requirements was that no functionality be lost when the object model was converted from OMT format to DAML. Two components we focused on were Time Management, and Data Distribution Management (DDM). Both play critical roles in assuring that the High Level Architecture functions correctly and efficiently.

4.2 Time Management

Time management is crucial in simulations because the order that messages are received greatly affects the outcome. Since we were addressing the conversion of SOM’s and FOM’s rather than the RTI, we only needed to ensure that each simulation maintained a correct ordering of events. DAML creators had already performed research on incorporating a unit of time into the language, and had a preliminary working ontology implementation, the DAML Time Ontology [8]. We employed their constructs. The DAML Time Ontology provides a method for attaching timestamps to simulation events, which, because the RTI maintains event ordering, constituted the bulk of the effort required of us.

4.3 Data Distribution Management

The High Level Architecture’s Data Distribution Management maintains routing spaces which allow simulations to receive only data that is relevant and hence desired. When making the conversion from the Object

Model Template to DAML, we needed to ensure we maintained this method for performing Data Distribution Management. We assumed during the coding process that a central interface, such as the RTI, would still exist. Our method for implementing the DDM was based on an algorithm outlined in previously published literature [9][10]. A routing space class is created, and it contains a vector comprised of UpdateRegion and SubscriptionRegion class instances. Each UpdateRegion object contains exactly one instance of an 'Attribute' class, to specify the attribute the region is designated for, and contains at least one instance of the 'Dimension' class, which specifies the boundaries of the region. Every UpdateRegion object will also have a Boolean value, called 'UpdateValue', which will be set by the RTI to let the federate know whether it needs to be recalculating its update region periodically and sending it to the RTI. This value is set to false when the RTI has no valid subscription regions for that specific attribute, and so there is no purpose in updating the UpdateRegion. Each SubscriptionRegion also has one Attribute class, and at least one Dimension class. However, this SubscriptionRegion is for the attributes that the current federate wants to subscribe to, not vice versa. In this way, the routing space table created is similar in functionality to the way it currently works with the OMT, assuming that there is a central interface.

4.4 Main Challenges

The conversion from OMT to DAML presented many challenges along the way. One of the most difficult problems to deal with was converting from a loose specification to a much more strict specification. There were many areas in which the current OMT proved to be ambiguous, and as we attempted to eliminate these uncertainties, we were faced with many different interpretations. In the end, after reviewing the IEEE 1516.2 Specification [3] thoroughly, we made the decisions that we felt created the specification that the HLA authors intended.

Since DAML was created specifically for the semantic web, and therefore Web Services, it lacked many of the ontologies and relationships we felt were necessary to adequately describe simulations. Therefore, we had to use the tools provided by DAML+OIL to create ontologies for simulations. Initially attempting to convert SOM examples helped us discover what relationships and properties were necessary to create a complete and accurate translation.

Finally, it was very difficult to be assured that no functionality had been lost in the conversion and that the DAML version was indeed an improvement over the OMT version since we had no way of executing our

specification to obtain quantitative results. Therefore, we had to measure the achievement of our goals quantitatively. Going through the IEEE 1516.2 Specification [3], we listed all of the components and functionality that the current OMT embodied and then examined our DAML specification to ensure they were accounted for. Lastly, our claims for improvement were supported by the plethora of features that DAML provided that XML did not.

5. Major Improvements

We find that DAML offered the following improvements over the OMT:

1. *Creation of a more precise specification.* Although the XML-based OMT specification provided more structure than the natural language based version, it still allowed for too much user interpretation. Our DAML-based version is, comparatively speaking, precise in its requirements. It requires that every attribute be linked to an object, and therefore it reduces the number of design decisions available to the user. We anticipate this strictness will enable the creation of a standard that will allow interoperable simulations to be created with greater ease.
2. *Three dimensional objects and linked definitions.* Again, the previous OMT version attempted to provide this ability, but it only provided a limited embedding of attributes within their respective objects, and ignored other relationships. Our version actually instantiates objects and attributes and then links them in a manner designers intended. Furthermore, we extend these properties to interactions and parameters.
3. *Ability to draw inferences based on a specification.* While XML does provide tools to describe information, it lacks the ability to draw logical conclusions regarding relationships. DAML supports the discovery that if A and B are equivalent properties, and A has a certain relationship with C, then B maintains that same relationship with C.
4. *Machine understandable, not just machine readable.* This allows for an 'intelligent' implementation of the Runtime Interface which would be able to infer additional information that a simulation doesn't explicitly provide, which would be useful for another simulation.
5. *Opportunity to exploit existing DAML ontologies and objects.* Since DAML was designed in order to facilitate the Semantic Web project, it was created with interoperability in mind, and therefore our DAML-based specification is able to potentially interact with all the currently existing DAML libraries.

6. Discussion

The current HLA OMT greatly restricts the effectiveness of the HLA. Our conversion of OMT descriptions to a semantic language ameliorates this deficiency and offers opportunity for HLA improvement. The OMT format is quite ambiguous, greatly reducing the coherence and compatibility of different object models. Our DAML Object Model has potential for greater precision and creates the opportunity to infer an entirely new set of requirements, much richer than those in the original OMT specification. Our specification provides strict guidelines for how object model creators must describe their simulations, effectively creating a new standard. This new standard will greatly facilitate interoperability and composability as simulation designers can be sure of the interface of other simulations, even those not yet created.

Although the scope of our project was limited to OMT conversion, DAML's support of ontologies and other features provides a plethora of additional opportunities for research, as the new object models are highly machine readable and understandable. For example, a rudimentary DAML parser already exists which can process DAML code and visualize relations and other information. Since our schema and ontologies were created using DAML+OIL, the DAML parser can be modified to create visualizations that are designed specifically for use with simulations. Furthermore, an enhanced version of the RTI could be implemented that is designed to work with and support all the features of the DAML Object Model format. This new RTI would employ semantic relationships to gather more information to provide to federates. Furthermore, linkage to pre-existing DAML relationships and objects becomes a possibility

Much work remains. A major task we plan to undertake is the conversion of more SOM's and FOM's into the DAML format in order to ascertain the need for additional tools for creating new object models. Regarding automation, a front-end to the DAML code is desirable which would allow users to enter simulation information at higher levels of abstraction than currently practiced. This front-end would support the automatic conversion of this information into DAML code. Lastly, as OWL becomes more popular, we anticipate porting our code from DAML into OWL, although the changes are seemingly minimal as OWL is based on DAML+OIL.

7. Summary

The High Level Architecture has been criticized for failing to foster interoperability and composability between simulations due to its inability to capture model semantics adequately. We argue that the majority of these

complaints are not about the HLA as a whole, but rather about Object Model Templates. We have explored remedying this problem by translating object models from OMT to DAML, a new semantics-based language that boasts the ability to express relationships in the form of ontologies and draw inferences based upon its own specification. This semantic format supports the creation of a standard protocol for object models through its specification requirements, which is the first step towards creating interoperable simulations. Also, DAML's machine readability creates future opportunities for integrating simulations with other DAML based services, such as the Semantic Web Project.

8. References

- [1] Tolk, A. "Avoiding another Green Elephant – A Proposal for the Next Generation HLA based on the Model Driven Architecture", 2002 Fall Simulation Interoperability Workshop, Orlando, Florida, 2002.
- [2] Rybka, M. "Moving Towards a Common Interpretation of HLA OMT 1.3", 2003 Spring Simulation Interoperability Workshop, 2003
- [3] Simulation Interoperability Standards Committee – "IEEE Std 1516.2: IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) – Object Model Template (OMT) Specification, Approved 21 September 2000.
- [4] "About DAML" – <http://www.daml.org>
- [5] "Frequently asked questions about RDF" <http://www.w3c.org/RDF/FAQ/>
- [6] "Why use DAML?" <http://www.daml.org/2002/04/why.html>
- [7] "OWL Web Ontology Language Overview" <http://www.w3.org/TR/2004/REC-owl-features-20040210/>
- [8] "DAML-Time" <http://www.cs.rochester.edu/~ferguson/daml/>
- [9] Crosbie, R., Zenor, J., Goberstein, S., "High Level Architecture Module 2: Advanced Topics – Data Distribution Management 1: Fundamentals and Multidimensional Regions", Cal State University, Chico, 11/4/99 <http://www.ecst.csuchico.edu/~hla/courses.html>
- [10] Crosbie, R., Zenor, J., Goberstein, S., "High Level Architecture Module 2: Advanced Topics – Data Distribution Management 2: Physically Correct Filtering", Cal State University, Chico, 11/4/99 <http://www.ecst.csuchico.edu/~hla/courses.html>

Author Biographies

ARSALAN TAVAKOLI is a 3rd year undergraduate pursuing a bachelor of science Computer Science and a double major in Economics with a concentration in

finance. His main research areas of interest include multiresolution modeling, coercible simulations, ad-hoc sensor networks, and real time databases.

DAVID CHU is a University of Virginia 4th year undergraduate studying towards a B.S. in computer science. His research interests are distributed systems with particular emphasis in simulations, mobile and context-aware computing, and sensor networks. He plans

to attend graduate school in computer science.

PAUL REYNOLDS is a Professor of Computer Science at the University of Virginia. His research interests include simulation technologies: simulation coercion, multi-resolution modeling, and parallel and distributed simulation, as well as concurrency control, language design and computing for the blind. His email address is <reynolds@cs.virginia.edu>.