UML for ontology development

Article *in* The Knowledge Engineering Review · February 2002

Impact Factor: 0.77 · DOI: 10.1017/S0269888902000358

7 authors, including:

Paul A. Kogut
Lockheed Martin Corporation

Stephen Cranefield
University of Otago

Kenneth Baclawski
Northeastern University

Mieczyslaw M. Kokar
Northeastern University

UML for Ontology Development

Paul Kogut, Lockheed Martin Management & Data Systems
Stephen Cranefield, University of Otago
Lewis Hart, GRC International
Mark Dutra, Sandpiper Software, Inc.
Kenneth Baclawski and Mieczyslaw Kokar, Northeastern University
Jeffrey Smith, Mercury Computer

Introduction

Ontologies are becoming increasingly important because they provide the critical semantic foundation for many rapidly expanding technologies such as software agents, e-commerce and knowledge management (McGuinness, 2001). The Unified Modeling Language (UML)[1] has been widely adopted by the software engineering community and its scope is broadening to include more diverse modeling tasks. This paper discusses the recent convergence of UML and ontologies and suggests some possible future directions.

Current ontology research and practice have their origins in declarative AI knowledge representations such as semantic networks and frames, which were primarily designed for runtime reasoning and inference. Recent ontology efforts such as the World Wide Web Consortium (W3C) Semantic Web initiative[2] and the DARPA Agent Markup Language (DAML) Program[3] have focused on formal logic-based and web-based knowledge representation (Berners-Lee, 2001). UML was originally designed for human-to-human communication of models for building systems in object-oriented programming languages. UML is now being used for designing artifacts that are more declarative (i.e., similar to ontologies) such as XML DTDs and Schemas (Carlson, 2001), Resource Description Framework (RDF) schemas[4], database schemas (Naiburg and Maksimchuk, 2001) and knowledge models (Schreiber, 1999). The Object Management Group (OMG) Model Driven Architecture (MDA)[5] which is based on UML and related standards such as the Meta-Object Facility (MOF) and XML Metadata Interchange (XMI) is evolving to support generation of application and middleware code and translation of data in a heterogeneous environment[6]. The MDA is driving UML to become more formal and machine-processable so that models can be used at compile time and runtime and not just as a graphical notation for human-to-human communication. Researchers are developing formal semantics for UML[7] and the MOF (Baclawski et al., September 2001). These efforts will remove one of the most commonly stated criticisms of the suitability of UML for representing formal models such as ontologies.

There is already growing momentum for the use of UML to design agent-based systems [8]. This paper will argue that UML is an excellent notation for the development and maintenance of ontologies for agent-based systems and the Semantic Web. This hypothesis is supported by explaining the advantages of UML and by discussing recent research and commercial efforts.

Why Use UML to Develop Ontologies?

Ontologies include class/subclass hierarchies, relationships between classes, class attribute definitions and axioms that specify constraints. In UML, this ontology information is usually modeled in class diagrams and Object Constraint Language (OCL) constraints. Other UML diagrams such as Statecharts and Activity Diagrams are also useful for service and process-related ontologies.

There are a number of good reasons why UML is a promising notation for ontologies:

---

[1] http://www.omg.org/technology/uml/
[2] http://www.w3.org/2001/sw/WebOnt/
[3] http://www.daml.org/
[4] http://Xmodel.sourceforge.net/
[5] http://www.omg.org/mda/
[6] http://www.omg.org/technology/cwm/
[7] Precise UML Group http://www.puml.org
[8] http://www.auml.org/

- UML is a graphical notation based on many years of experience in software analysis and design by a variety of companies in a wide spectrum of industries and domains.
- UML is an open standard maintained by the OMG.
- UML has standard mechanisms for defining extensions for specific application contexts such as ontology modeling.
- UML is widely adopted in industry and taught in many university courses. Current techniques for ontology development are based on knowledge representations such as Knowledge Interchange Format (KIF)[9] which are not widely known outside the AI research community.
- UML is supported by widely-adopted CASE tools. These UML CASE tools are more accessible to software practitioners than current ontology tools from the research community such as Ontolingua[10] and Protege[11], which require expertise in knowledge representation.
- Real world industrial agent-based systems need to interact with legacy enterprise systems, which often have existing UML models.

Recent Applications of UML for Ontology Representation

A variety of different research projects and commercial initiatives have been applying UML for ontology representation. This section briefly describes these efforts. The approaches taken in these efforts vary in a number of different ways including:

- UML has been used directly as an ontology representation and as a graphical front-end for another ontology representation language (e.g., DAML+OIL[12] - referred to as DAML in the rest of this paper).
- UML has been used with a variety of agent infrastructures and knowledge base implementations (e.g., Java objects and the Open Knowledge Base Connectivity[13] (OKBC) API).
- UML has been applied to a variety of ontology related tasks (e.g., ontology mapping and consistency checking).

Cranefield and Purvis (1999, 2000) have investigated the use of UML class diagrams for representing ontologies and UML object diagrams for representing instance knowledge. A "UML Data Binding" tool for Java (Cranefield, 2001) has been developed to generate Java classes and RDF schemas from a class diagram encoded in the XMI format. In this work, UML is used directly, not as a graphical syntax for another knowledge representation language.

Bergenti and Poggi (2000) have proposed an approach to agent-oriented software engineering based on the use of UML to model various aspects of a multi-agent system. One of their proposed diagrams is an "ontology diagram", which depicts classes representing agents and domain entity types, and associations representing domain predicates that can be encoded as KIF or FIPA-SL agent message content.

The UML Based Ontology Tool-set (UBOT) project[14] is building ontology engineering and natural language processing-based text annotation tools for DAML. UML is used as a front-end for visualizing and editing DAML ontologies. The approach is to extend UML by defining a prototype UML profile for DAML which maps UML stereotypes to DAML-specific elements (Baclawski et al., October 2001). The UBOT tools use Telelogic Tau UML Suite for editing and generating XMI that is translated to DAML. The UBOT project has been experimenting with formal methods to check the consistency of DAML ontologies[15]. The UBOT tools are being evaluated in a satellite imagery analysis workflow agent application.

---

[9] http://logic.stanford.edu/kif/kif.html
[10] http://www.ksl.stanford.edu/sns.shtml
[11] http://www.smi.stanford.edu/projects/protege/
[12] http://www.daml.org/language/
[13] http://www.ai.sri.com/~okbc/
[14] http://ubot.lockheedmartin.com/
[15] http://vis.home.mindspring.com/

The Components for Ontology Driven Information Push (CODIP) project[16] is using UML to build and map DAML ontologies in support of publish-subscribe channels for disseminating DAML messages in domains such as military logistics. Message source ontologies are mapped to message destination ontologies by creating a third articulation ontology that is a collection of concepts and relations relating semantically equivalent concepts. The CODIP project is building a DAML-UML Enhanced Tool (DUET) which is based on Rational Rose add-ins and a UML profile for DAML (Baclawski et al., October 2001). DUET works with an ontology articulation builder that provides automated analysis of potential mappings allowing the user to interactively build an articulation between the UML models.

Sandpiper Software[17] is developing commercial grade tools that support knowledge modeling and information brokering (Dutra, 2001). Sandpiper has extended UML to enable representation of rich ontological knowledge through the creation of a UML profile for frame-based knowledge representation. They have implemented an add-in to Rational Rose that supports consistency checking and wizards that prompt the user to enter the information required. The wizards allow domain experts with little knowledge engineering expertise to build ontologies. Sandpiper has developed an OKBC-based interface for a commercial OODBMS for its knowledge base implementation. Included with Sandpiper's ontology modeling tool are base ontologies such as the IEEE Standard Upper Ontology (SUO)[18].

In summary, UML has been successfully used for a wide range of ontology-related tasks and to model a variety of ontologies including challenging abstract upper ontologies such as IEEE SUO.

Issues and Future Directions

The OMG MDA and the W3C Semantic Web have a similar semantic interoperability vision. However, they are pursuing different metadata approaches. Both are using XML syntax but OMG and W3C have different semantic models. The OMG community is already using XMI to transfer data between disparate databases. The Semantic Web community is experimenting with DAML as a content language for agent messages. It is interesting to note that OMG has built a meta-metamodel (i.e., the MOF) by identifying a core set of elements from existing complex metamodels (i.e., UML and CORBA IDL) whereas the W3C is building new layers of complexity on top of simpler layers (i.e., XML -> RDF -> RDF Schema -> DAML+OIL...). There have been some efforts to bridge these two disparate metadata approaches[19]. However, there are serious concerns about significant semantic mismatches between the OMG and W3C approaches. Some of these significant mismatches are:

- The W3C approach does not have a clean layered architecture (Pan and Horrocks, 2001). This leads to confusing situations like a class being an instance of another class. The OMG follows the software engineering philosophy of strict separation between classes and instances.
- The OMG approach does not have a first-class concept of an "association" (analogous to a "property" in W3C terminology) (Baclawski et al., October 2001). Associations in UML and MOF can only exist in the context of two or more classes. Properties in RDF and DAML are first-class elements that can be defined in an ontology without reference to classes. If an RDF schema or DAML ontology states that "company owns vehicle" and "person owns dog", "owns" is the same property whereas they would be different associations in a UML model.
- The OMG axiomatic notation, OCL, was designed to specify constraints on models with little concern for the computational complexity of runtime reasoning. In contrast, tractable reasoning is an explicit design goal for the emerging W3C ontology and axiomatic notations.

The ongoing OMG UML2.0 revision efforts and the new W3C Semantic Web Activity are good opportunities to bring the metadata approaches closer together to avoid confusion and costly workarounds. The OMG-based technology has mature support for graphical visualization of models and database interoperability. The OMG community needs to evolve its technology to provide better support for agent-based systems and the Semantic Web.

---

[16] http://codip.grci.com/
[17] http://www.sandsoft.com/
[18] http://suo.ieee.org/
[19] http://www-db.stanford.edu/~melnik/rdf/uml/

Acknowledgments

Some of this material is based upon work supported by the Air Force Research Laboratory, Contract Number F30602-00-C-0188. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Air Force.

References

Baclawski, K., Kokar, M., and Smith, J., "Metamodeling Facilities" working paper presented at OMG Technical Meeting, September 2001, available at http://vis.home.mindspring.com/

Baclawski, K., Kokar, M., Kogut, P., Hart, L., Smith, J., Holmes, W., Letkowski, J., and Aronson M., "Extending UML to Support Ontology Engineering for the Semantic Web." *Proc. of the Fourth International Conference on UML (UML2001)*, Toronto, October 2001

Bergenti, F., and A. Poggi. "Exploiting UML in the Design of Multi-Agent Systems." In Engineering Societies in the Agents World, Lecture Notes in Computer Science 1972 (Springer), 2000.

Berners-Lee, T., Hendler, J., and Lassila, O., "The Semantic Web," *Scientific American*, May, 2001

Carlson. D., *Modeling XML Applications with UML*, Addison-Wesley 2001

Cranefield, S., and Purvis, M. "UML as an Ontology Modeling Language," *Proc. of the Workshop on Intelligent Information Integration, 16th Int. Joint Conference on AI (IJCAI-99)*, Stockholm, 1999

Cranefield, S., and Purvis, M. "Extending Agent Messaging to Enable OO Information Exchange." in "Cybernetics and Systems 2000," R. Trappl (ed.), Vienna, Austrian Society for Cybernetic Studies (2000)

Cranefield, S. "UML and the Semantic Web," *Proc. of the International Semantic Web Working Symposium*, Palo Alto, 2001

Dutra, M., "UML for Knowledge Representation," OMG's Second Workshop on UML for Enterprise Applications, December 2001

McGuinness. D., "Ontologies Come of Age," to appear in Fensel, Hendler, Lieberman, Wahlster editors *The Semantic Web: Why, What, and How*, MIT Press 2001

Naiburg, E., and Maksimchuk, R., *UML for Database Design*, Addison-Wesley 2001

Pan, J., and Horrocks, I., "Metamodeling Architecture of Web Ontology Languages," *Proc. of the Semantic Web Working Symposium*, Palo Alto, August 2001

Schreiber, G., *Knowledge Engineering and Management*, MIT Press, 1999