

Metamodels for RDF Schema and OWL

Daniel T. Chang
IBM Silicon Valley Lab
555 Bailey Ave., San Jose, CA 95141
dtchang@us.ibm.com

Elisa Kendall
Sandpiper Software, Inc.
2053 Grant Road, #162, Los Altos, CA 94024
ekendall@sandsoft.com

ABSTRACT

This paper presents the working draft MOF™ (Meta-Object Facility) metamodels for the Resource Description Framework (RDF Schema) and the Web Ontology Language (OWL), two of the six metamodels currently envisioned for the Ontology Definition Metamodel (ODM) standards effort in the Object Management Group (OMG®), which enable model-driven development of RDF vocabularies and OWL ontologies, respectively. We provide insight into some of the design principles used in developing these metamodels, major challenges addressed to date, and the resolution of some of these issues that has influenced the resultant products. We also briefly review ongoing and future work needed to complete the subset of the ODM specific to these representation formalisms and fully support model driven development of RDF vocabularies and OWL ontologies.

INTRODUCTION

Over the course of the last five years, and more specifically, since the emergence of Semantic Web Activity from the World Wide Web Consortium (W3C) [1], the development of ontologies—explicit formal specifications of the concepts in a domain and relations among them [2]—has been moving from the research community to early adoption by industry. Increasing evidence of collaborative development of large, standardized controlled vocabularies and ontologies for specific applications and domains, such as in bioinformatics and pharmacogenomics research, is appearing in the literature. Broadly applicable general-purpose ontologies, for example those supporting the Semantic Web Services Initiative [3], are emerging as well.

Ontologies are primarily captured in knowledge representation languages developed by the artificial intelligence community. Most of the more commonly-used authoring languages, such as the description logics family of languages [4] or first-order and predicate logic languages, such as Simple Common Logic (SCL) [5] or its predecessor, the Knowledge Interchange Format (KIF) [6], were designed to support machine reasoning. Their text and logic based structure, however, has led to a language syntax that is unfamiliar and awkward for subject matter experts to learn and use effectively, which has been a major hindrance to the development of ontologies. Notably, the recently adopted W3C specifications for the Resource Description Framework (RDF), RDF Schema [7] and the Web Ontology Language (OWL) [8,9] in particular, are somewhat less intimidating to those familiar with XML syntax or constructs used in object-oriented programming, such as classes, properties, and individuals. OWL is an extension of RDF Schema, which is the vocabulary description language for RDF. Both RDF Schema and OWL are key components of the W3C Semantic Web initiative and are likely to gain increasing support in the

industry.

Having domain expert friendly languages, however, is necessary but not sufficient to promote widespread adoption of these technologies. Current approaches to ontology development are at best, more art than science, and in general, ad hoc. First, the process of ontology development is extremely time consuming and not at all visually intuitive. Any non-trivial ontology represented in OWL, as is, is challenging for domain experts to understand and maintain. Secondly, developing ontologies in isolation of business requirements is of little practical value. Ontology development must become an integral part of the systems analysis and engineering activities of the CIO function. That is, the ontologies that an enterprise develops must form an integral part of that enterprise's information and application infrastructure

This paper promotes the use of Model Driven Architecture (MDA®) and related methodologies for ontology development. This is prompted by current trends towards MDA in software engineering and best practices, as a result of the related standards efforts in the OMG, and the availability of EMF (Eclipse Modeling Framework) as an open source model-driven software development platform. Key standards in the MDA family include the Meta-Object Facility™ (MOF®) [10] and the Unified Modeling Language™ (UML®) [11].

UML is an industry-standard, graphical language that is used by software engineers for conceptual modeling. The similarity of UML constructs to constructs used in defining ontologies suggests that UML could be leveraged by the large community of existing practitioners to promote broader use and increasing development of ontologies. UML is well established in many commercial and government software engineering organizations with extensive tool support from both commercial and open source vendors. We believe, therefore, that UML is an excellent candidate notation for the graphical development and maintenance of ontologies.

To facilitate the development of tools and methods for MDA-based ontology development, as UML is defined using MOF, what is needed is a MOF based metamodel for OWL. Such a metamodel will enable:

- Forward engineering: development of OWL ontologies using MOF based (in particular, UML) modeling tools.
- Reverse engineering: leveraging existing OWL ontologies for ontology modeling and UML modeling.
- Integrated ontology/software development: making OWL ontology development an integral part of

software development.

This paper presents the draft set of MOF metamodels for RDF Schema and OWL that are currently proposed as a part of the Ontology Definition Metamodel (ODM) activity in the OMG, enabling model-driven development of RDF vocabularies and OWL ontologies, respectively. Because of space limitations, we provide only a brief overview of the metamodels here. We present insight into some of the design principles used in developing these metamodels, major challenges addressed to date, and the resolution of some of these issues that has influenced the resultant products. We also briefly review ongoing and future work needed to complete the subset of the ODM specific to these representation formalisms and fully support model driven development of RDF vocabularies and OWL ontologies.

DESIGN PRINCIPLES AND MAJOR CHALLENGES

Both RDF Schema and OWL are defined using RDF Schema. That is, RDF Schema serves as the meta language that defines itself and OWL. However, to leverage MDA, EMF, and UML technologies, MOF serves as the meta language, or meta-metamodel used to define the metamodels for RDF Schema and OWL. In developing these metamodels, we encountered a number of distinctions between MOF and RDF Schema that have either challenged our ability to leverage MOF for this purpose or have resulted in metamodels that are less than ideal, as discussed below.

RDF Schema uses URI references for naming. Definitions specified in RDF documents (class, property, individual) have globally unique names qualified by namespaces, such as `rdfs:Class`, `rdf:Property`, `rdfs:subClassOf`, `owl:Class`, and `owl:equivalentClass`. MOF, on the other hand, uses package-scoped naming for classes and locally scoped naming for associations. In the current set of metamodels, to clearly indicate what a particular notion, such as a class or an association, describes (i.e., its source), we have elected to use prefixes to denote namespaces. Thus, the entities itemized above are called, respectively: `RDFSClass`, `RDFProperty`, `RDFSsubClassOf`, `OWLClass`, and `OWLequivalentClass`.

RDF Schema and OWL (and actually most knowledge representation formalisms and methodologies) do not make distinctions between meta-levels. For example, `rdfs:Class` is used to define `owl:Class` (i.e., `owl:Class` is an instance of `rdfs:Class`), but `owl:Class` is also defined as a subclass of `rdfs:Class`. From an MDA perspective, however, these distinctions are extremely important: MOF class, at the M3 level, is used to define M2 level classes, such as `RDFSClass` and `OWLClass`. UML does not support representation of classes and objects on the same diagram, which is a frequent requirement in knowledge representation. As a result, we have attempted to respect separation of meta-levels whenever possible in the ODM metamodels, for example, `OWLClass` is defined as a subclass of `RDFSClass`, but it is not an instance of `RDFSClass`.

RDF Schema and OWL are based on concepts from formal logic, as mentioned above. They provide unary predicates, i.e., classes and individuals, for representing concepts in a domain and binary predicates, in the form of properties, for representing relations between concepts. Both RDF classes and RDF properties are first-class entities and have globally unique identifiers. MOF, on the other hand, is class based. It provides classes for representing concepts in a domain, attributes for representing characteristics that are common to all instances of a given concept, and associations for representing relations between concepts. MOF classes are first-class entities that may have globally unique identifiers (package scoped). Attributes and associations, however, are not first-class citizens. They have only class-scoped, local names, as mentioned previously. This raised a fundamental question as to how best to represent RDF properties in the metamodels: reified as MOF classes so that they have globally unique identifiers or as MOF associations so that they appear more natural to MOF/UML users. In the current metamodels, we have described predefined RDF properties, e.g., `rdfs:subClassOf` and `owl:equivalentClass`, as MOF associations in order to make the metamodels easier to understand, simplify the serialized XMI rendering of the metamodels, and for ease of use in MOF and UML tools. Three prefixes (RDF, RDFS and OWL) have been used to denote namespaces, for example `RDFSsubClassOf` and `OWLequivalentClass`, to “simulate” globally unique names, however.

Again, the primary goal of this work is to enable model driven development of RDF vocabularies and OWL ontologies. As such, we want the metamodels to be as directly representative of RDF Schema and OWL, respectively, as possible. Therefore, we have kept to a minimum any construct that is not explicitly defined in RDF Schema or OWL. Where we felt such a construct was necessary, for example, Vocabulary / Graph for scoping purposes in the RDF Schema metamodel, we have named these artifacts without any global prefix for further clarification.

THE RDF SCHEMA METAMODEL

The RDF Schema metamodel uses diagrams to control complexity and promote understanding, as exemplified in Figures 1 and 2. The MOF classes and MOF associations are grouped into seven diagrams:

- **Classes** — Contains classes and associations that can be used to define RDF classes and RDF datatypes.
- **Properties** — Contains classes and associations that can be used to define RDF properties.
- **Containers** — Contains classes and associations that can be used to define RDF containers and their members.
- **Collections** — Contains classes and associations that can be used to define RDF collections (i.e., lists) and their members.
- **Reification** — Contains classes and associations that can be used to define RDF statements.

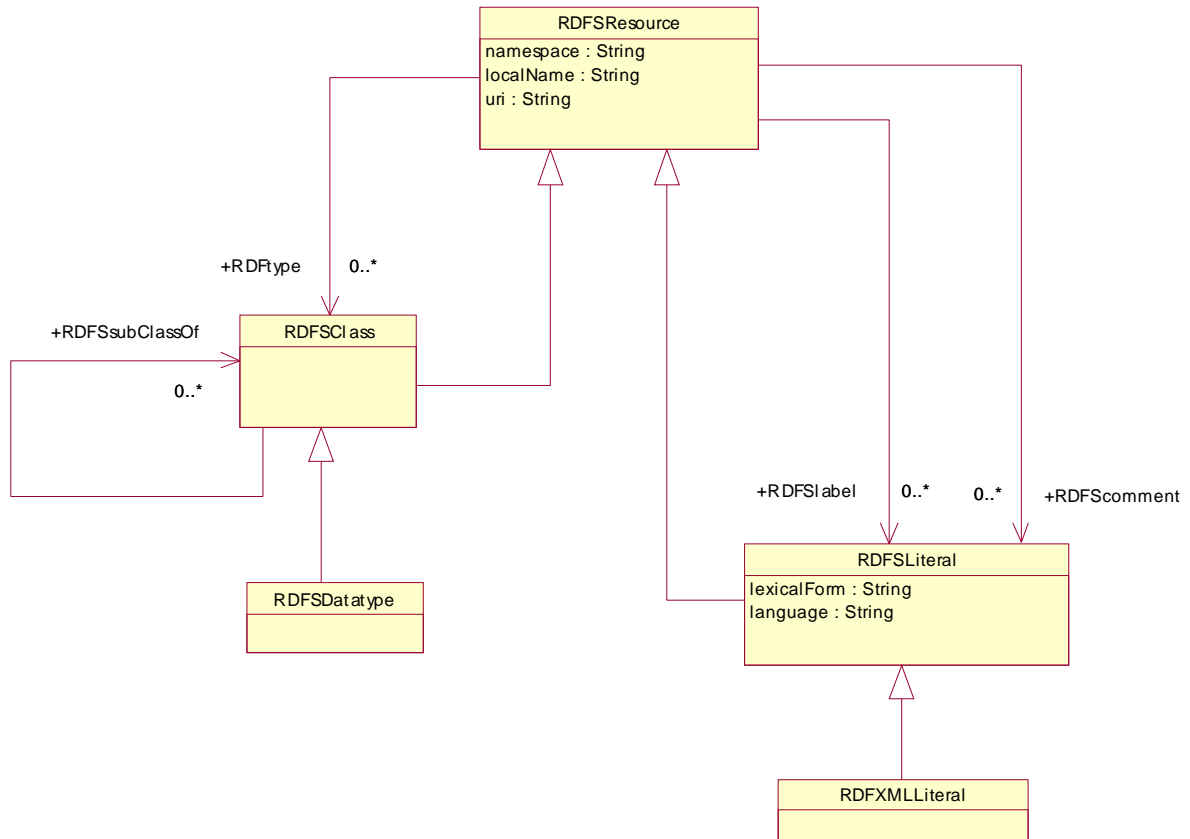


Figure 1. RDFS Classes Diagram

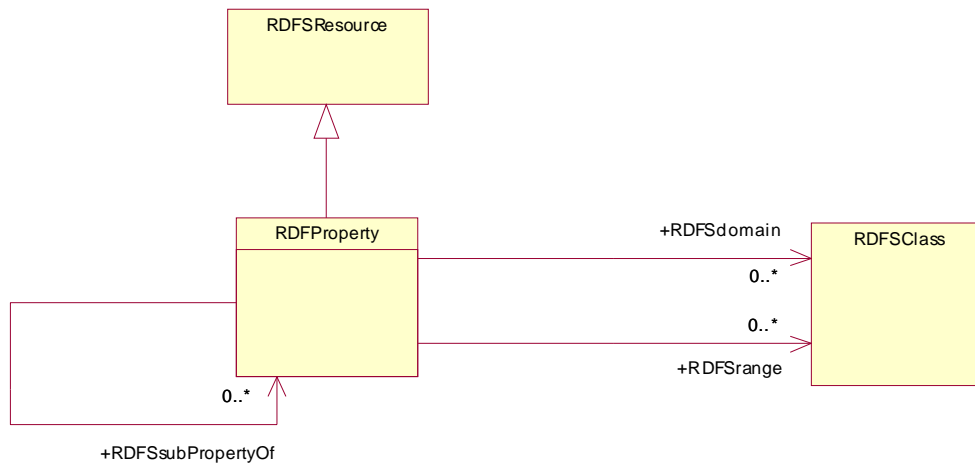


Figure 2. RDFS Properties Diagram

- Utilities — Contains classes and associations that can be used to define utility RDF properties.
- Vocabulary — Contains classes and associations that can be used to define the scope of an RDF vocabulary.

Two of the primary diagrams, the RDFS Classes

diagram and the RDFS Properties diagram, are shown in Figures 1 and 2, respectively.

THE OWL METAMODEL

The OWL Metamodel, as illustrated in the OWL Classes Diagram (Fig. 3), also uses diagrams to control complexity and promote understanding.

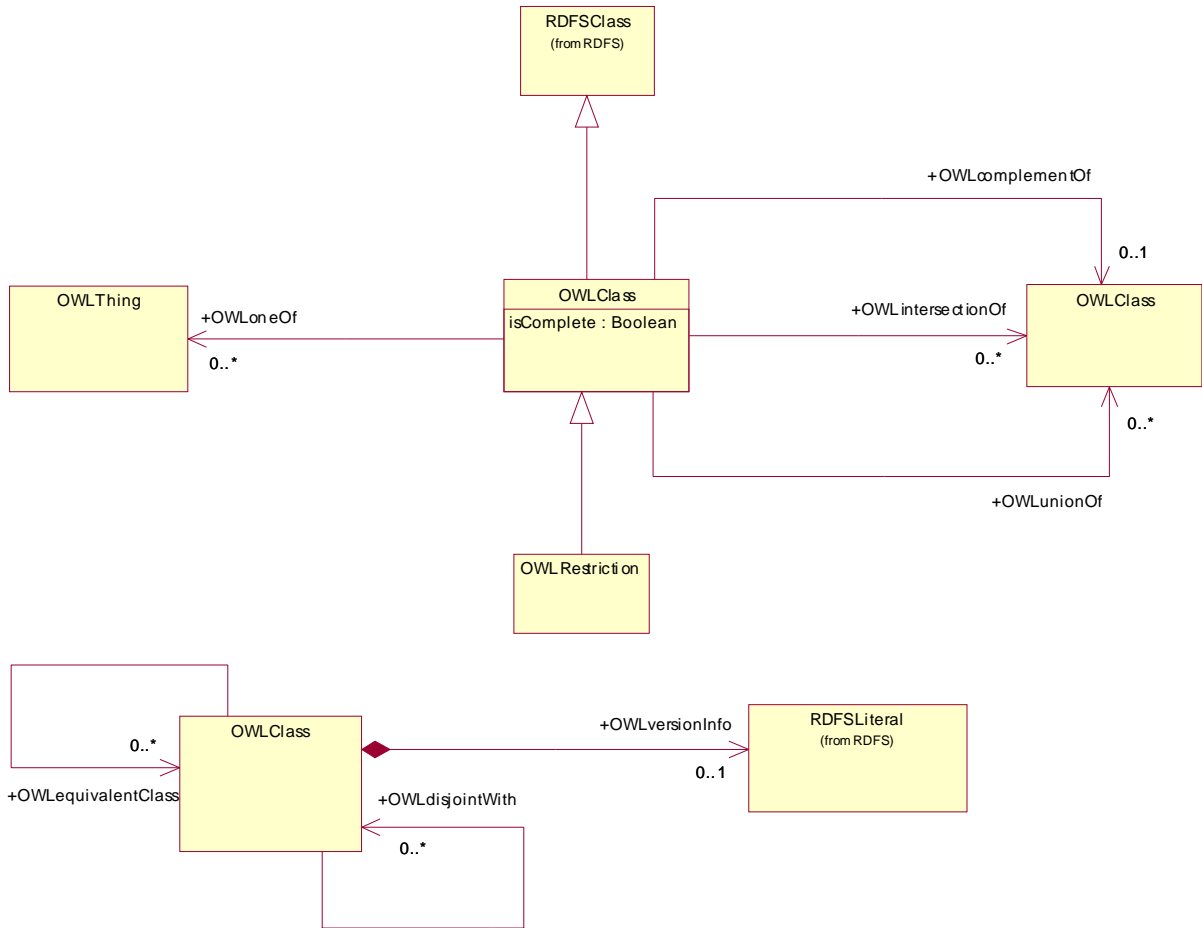


Figure 3. OWL Classes Diagram

The MOF classes and MOF associations are grouped into seven diagrams:

- **Classes** — Contains classes and associations that can be used to define OWL classes.
- **Restrictions** — Contains classes and associations that can be used to define OWL restrictions.
- **Properties** — Contains classes and associations that can be used to define OWL properties.
- **Individuals** — Contains classes and associations that can be used to define OWL individuals.
- **Datatypes** — Contains classes and associations that can be used to define OWL datatypes.
- **Utilities** — Contains classes and associations that can be used to define utility OWL classes.
- **Ontology** — Contains classes and associations that can be used to define the properties of an OWL ontology.

Several of these diagrams are essential to understanding the

metamodels and our approach, so we encourage interested researchers and practitioners to review the entire metamodel [12], but to further illustrate what we have done to date, the OWL Properties diagram is shown in Figure 4.

FUTURE WORK

OWL consists of three increasingly expressive representation formalisms designed for use by specific communities:

- **OWL Lite** — supports users primarily concerned with classification hierarchies and simple constraints.
- **OWL DL** — supports users who want the maximum expressivity without loss of computational completeness and decidability in certain reasoning systems.
- **OWL Full** — supports users who want maximum expressivity and the syntactic freedom of RDF, with less concern for computational guarantees.

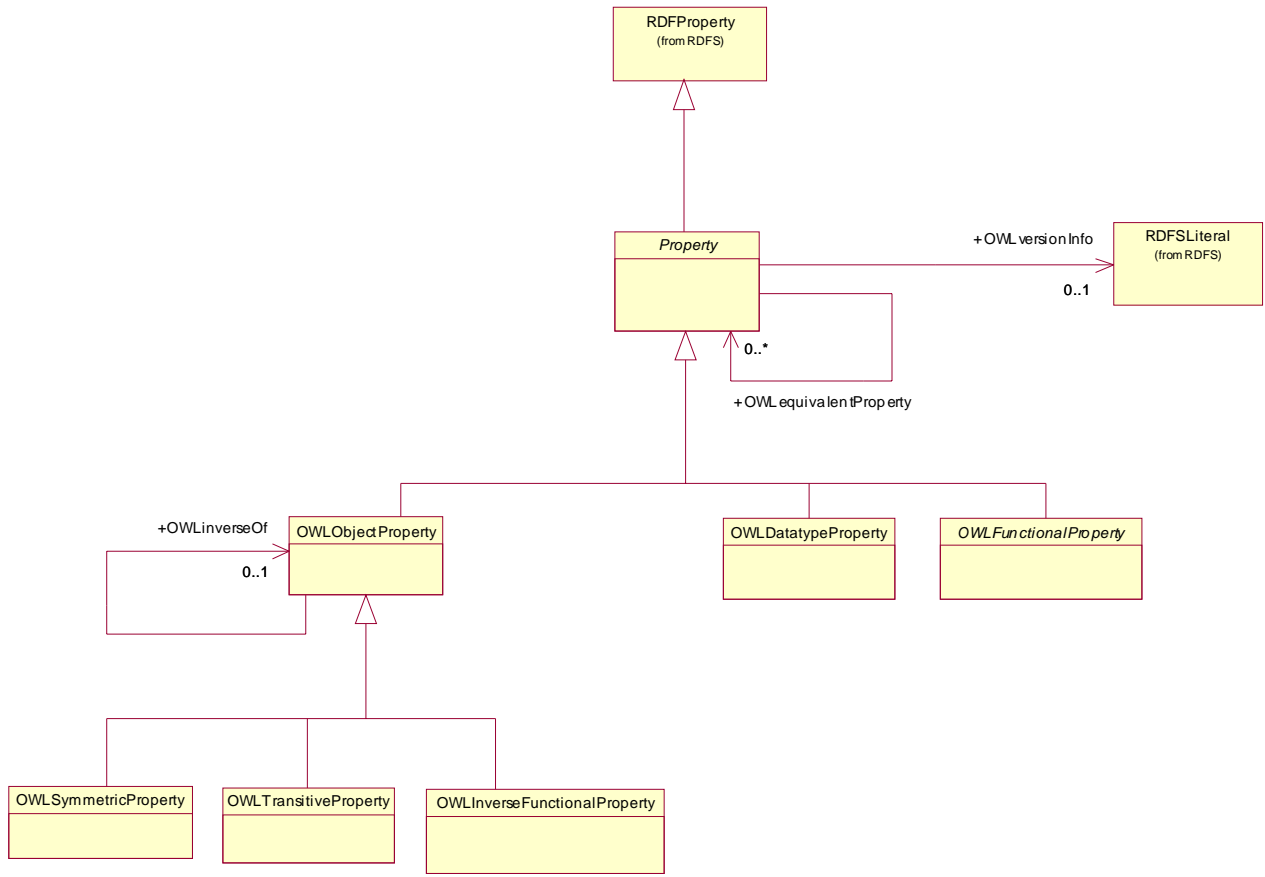


Figure 4. OWL Properties Diagram

The OWL metamodel as currently defined is capable of representing any of the three sublanguages of OWL. However, it does not yet include the constraints required to limit OWL Lite and OWL DL per the W3C recommendations. Future work is planned to incorporate such constraints in the submission using OCL.

Additionally, we have used the RDFS and OWL metamodels to generate EMF-based Java APIs. These APIs are part of JODM, which includes Java APIs for the full set of ODM metamodels, and consists of six Java packages: org.omg.odm.dl, org.omg.odm.er, org.omg.odm.rdfs, org.omg.odm.owl, org.omg.odm.scl, and org.omg.odm.tm. The org.omg.odm.rdfs package has been used as the basis for developing the EODM tool that is part of the IBM Semantics Toolkit [13]. We plan to make JODM an open source project so that it can be widely utilized.

The RDFS and OWL metamodels enable model-driven ontology development using MOF based tools, as illustrated by EODM. However, they may not be sufficient for enabling such development using UML based tools, such as IBM Rational Rose®, MagicDraw™, ArgoUML, and others. We are currently investigating the need to define UML profiles for RDF Schema and OWL to enable the use of UML notation for modeling purposes and for subsequent

generation of corresponding RDF Schema and OWL ontologies. Reference [14] provides additional insight into some of the overall design decisions driving the architecture of the ODM. Depending on the outcome of our research and prototyping activities, such profiles will be included as part of the joint ODM revised submission. We also plan to define mappings from the RDFS and OWL metamodels to RDF Schema and OWL, respectively, and include them in the joint revised submission.

ACKNOWLEDGMENT

The authors would like to recognize the contribution of Yiming Ye to the initial design of the metamodels for RDF Schema and OWL. They would also like to thank the rest of the ODM submission team for ideas that contributed to the final design of the metamodels: Robert Colomb, Dave Frankel, Patrick Emery, and Lewis Hart. This paper represents the personal perspectives of the authors and does not necessarily represent the common perspectives of the whole team.

REFERENCES

[1] Tim Berners-Lee and Mark Fischetti, *Weaving the Web: the Past, Present and Future of the World Wide Web by Its*

Inventor, Orion Business, London, 1999.

[2] T. R. Gruber. *Towards Principles for the Design of Ontologies Used for Knowledge Sharing*, Technical Report KSL-93-04 Knowledge Systems Laboratory, Stanford University, 1993.

[3] See <http://www.daml.org/services/swsl/>

[4] F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi, P.F. Patel-Schneider (editors), *The Description Logic Handbook: Theory, Implementation and Applications*, Cambridge University Press, January 2003.

[5] Michael Genesereth & Richard Fikes, *Knowledge Interchange Format, Version 3.0 Reference Manual*, KSL Report KSL-92-86, Knowledge Systems Laboratory, Stanford University, June 1992.

[6] Harry Delugach, *ISO/IEC WD 24707 Information technology – Common Logic (CL) – A Framework for a Family of Logic-Based Languages*, Pacific Northwest National Laboratory, Chantilly, VA, 7 June 2004.

[7] *RDF Vocabulary Description Language 1.0: RDF Schema*. W3C Recommendation 10 February 2004. Latest version is available at <http://www.w3.org/TR/rdf-schema/>.

[8] *OWL Web Ontology Language Reference*. W3C Recommendation 10 February 2004. Latest version is

available at <http://www.w3.org/TR/owl-ref/>.

[9] *OWL Web Ontology Language Semantics and Abstract Syntax*. W3C Recommendation 10 February 2004. Latest version is available at <http://www.w3.org/TR/owl-semantics/>.

[10] *Unified Modeling Language™ (UML®) Infrastructure Specification, Version 2.0*, Object Management Group, Inc., Needham, MA, December 2003.

[11] *Meta-Object Facility (MOF™) Specification, Version 2.0*, Object Management Group, Inc., Needham, MA, August 2003.

[12] See <http://codip.grci.com/odm/draft/>.

[13] Guotong Xie, Shixia Liu, Zhuo Zhang, Li Ma, Yue Pan, Li Zhang, Zhong Su, and Li Qin Shen, “SemanticWare: An EMF-Compatible RDF Infrastructure,” in *Proceedings, the 8th International IEEE EDOC Conference, 1st International Workshop on the Model Driven Semantic Web (MDSW 2004)*, Monterey, CA, September 21, 2004.

[14] Daniel T. Chang and Elisa Kendall, “Major Influences on the Design of ODM,” in *Proceedings, the 8th International IEEE EDOC Conference, 1st International Workshop on the Model Driven Semantic Web (MDSW 2004)*, Monterey, CA, September 21, 2004.