CSc 8711 Report: OWL API

Syed Haque
Department of Computer Science
Georgia State University
Atlanta, Georgia 30303
Email: shaque4@student.gsu.edu

Abstract: The Semantic Web is an extension of human-readable webpages towards making them machine processable so that software agents can be deployed on the web to intelligently perform automated tasks in place of users. Semantic Web makes use of technology which includes Resource Description Framework (RDF) , RDF Schema , Web Ontology Language (OWL) and other data interchange formats to define concepts, relationships in a given knowledge domain. Web Ontology Language (OWL) is one of the knowledge representation language for creating, manipulating and processing ontologies and is a W3C recommendation since May, 2004. This report is a study of the OWL API a java based API for semantic web ontologies. The WonderWeb group [6] produced the initial implementations of the API and the current development is supported by the CO-ODE group [7] which also provides the Protégé tool development.

History: The history of OWL can be traced back to two independent works DAML-ONT and OIL. OIL was a frame-based language developed by group of European researchers with RDFS based syntax and semantics described in terms of Descriptive Logics. DAML-ONT was developed largely by US based researchers which extended RDFS with constructs from object-oriented language and frame-based languages. The problems with DAML-ONT included problems with both human and machine interpretation. DAML+OIL language came into being with the merging of OIL and DAML-ONT which had a RDFS syntax based on Descriptive Logic. DAML+OIL was submitted to W3C for standardization which led to the Web Ontology Language (OWL).

Ontology in Computer Science: An ontology is defined as an engineering artifact constituted by a specific vocabulary used to describe a certain reality and a set of explicit assumptions regarding the intended meaning of the vocabulary. Thus an ontology describes a formal specification of a certain domain that allows for shared understanding of a domain of interest and formal and machine manipulable model of a domain of interest. Ontology represents knowledge in terms of concepts defined by classes, properties and individuals and a set of axioms that assert how those concepts are to be interpreted. The building of a semantic web requires following : i) Annotation – Associating metadata with resources ii) Integration – Integrating information sources iii)

Inference – Reasoning over the information we have iv) Infrastructure - Could be light-weight or Could be heavy-weight v) Interoperation and Sharing are key goals

Semantics and Languages: Semantic Web is based on frame based modeling approaches that describes concepts in terms of classes, slots, fillers. The language constructs is derived from RDF, RDF(S) and includes logical foundation through which well-formed inference can be derived. The increased formality makes languages more amenable to machine processing. The formal semantics provides an unambiguous interpretation of the descriptions and well defined semantics are vital to support machine interpretability and to remove ambiguities in the interpretation of the descriptions. The work on Semantic Web has concentrated on the definition of a collection or "stack" of languages namely XML, RDF, RDF(S), OWL. These languages provide basic machinery that we can use to represent the extra semantic information needed for the Semantic Web.

OWL API Philosophy and Abstract Syntax: There are three variations of OWL existing currently: OWL Full, OWL DL and OWL Lite . The OWL API is targeted primarily at representing OWL-DL. An Ontology is represented as a collection of axioms that assert information about the classes, properties and individuals that are in the ontology. The API is build with the philosophy that a higher level data model can help to insulate us from the vagaries of concrete syntax, make it clear what is happening in terms of functionality, increase the likelyhood of interoperating applications and make it easy to write code spotting "internal errors". The OWL abstract syntax provides a definition of the language in terms of the constructs and assertions allowed. Semantics are then defined in terms of this abstract syntax. OWL API data model is based largely on this abstract syntax presentation.

OWL API Implementation: The OWL API implementation includes the following functionalities on an ontology: i) Modeling – A set of data structures to represent OWL ontologies/documents ii) Parsing - Taking some syntactic presentation, e.g. OWL-RDF and converting it to some (useful) internal data structure iii) Serializing - Producing a syntactic presentation, e.g. OWL-XML from a local data structure iv) Manipulation/Change - Being able to manipulate the underlying objects v) Inference - Providing a representation that implements/understands the formal semantics of the language. The API provides classes to create, manipulate, parse, render and reason about those structures. The model data structures provide representations of the basic building blocks.The basic data structure represents the objects in the ontology and corresponds roughly to the abstract syntax of OWL.

OWL Classes: OWL allows to describe a domain in terms of individuals, classes and properties and a collection of axioms describing how these classes, individuals, properties etc. should be interpreted. Some of the important classes in the API are :-

i) OWLOntology - An OWLOntology represents an ontology and consists of a collection of OWLAxioms. The OWLOntology provides an explicit context within which we can assert information about classes and properties

ii) OWLEntity - OWLEntity is the fundamental building block of the ontology classes, properties, individuals and datatypes.

iii) OWLClass – OWLClass represents an OWL Class. The class itself is a lightweight object and axioms relating to the class are held by an OWOntology object
iv) OWLProperty – OWLProperty is the base class for OWLObjectProperty and OWLDataProperty. Properties can have associated domains and ranges.
v) OWLObjectProperty - Represents an Object Property that can be used to relate two individuals.
vi) OWLDataProperty - Represents an Object Property that can be used to relate an individual to a value.
vii) OWLAxiom – OWL Axiom class contains a collection of OWLAxioms namely Annotation Axioms, Declaration Axioms, Import Axioms, Logical Axioms

The API also provides a collection of interfaces for inference services over the ontology model:-

i) OWLClassReasoner – Checks for class satisfiability
ii) OWLConsistencyChecker – Checks for Ontology consistency
iii) OWLIndividualReasoner - Reasoning about individuals
iv) OWLPropertyReasoner - Reasoning about properties

The Reasoner implementation is not a part of the OWL API but the API can make use of the following reasoner implemenatations:-

i) Pellet , Hermit – Pure Java implementation and implements OWL API reasoner interfaces
ii) FaCT++ - C++ Implementation that acts as OWL API wrapper implementing OWL API interfaces.
iii) Debuggers – Reside on top of reasoner

Ontology Management: The Management and creation of ontologies is controlled by an OWLOntologyManager class object. This replaces OWLConnection/OWLManager in the original OWL 1.1 implementation. The Manager is responsible for keeping track of the ontologies and concrete formats for storage of the ontologies.

Pseudo-Code Examples:

1) The following code snippet shows loading of an ontology using a logical URI by creating a mapping of logical URI and physical URI :-

```
//create OntologyManager object for working with ontologies
OWLOntologyManager manager = OWLManager.createOWLOntologyManager();
//Logical URI
IRI ontologyIRI = IRI.create("http://www.test.com/ontologies/testont.owl");
//Physical URI
```

```
IRI documentIRI = IRI.create("file:/C:/Users/Syed/Desktop/animal.owl");
//Create a mapping for logical and physical URI using SimpleIRIMapper class object
SimpleIRIMapper mapper = new SimpleIRIMapper(ontologyIRI, documentIRI);
manager.addIRIMapper(mapper);
```

2) The following pseudo-code shows how to create a class in the ontology:-

```
//Create a OWLDataFactory object that actually contains reference to all entities in an ontology namely classes, individuals, properties
OWLDataFactory factory = manager.getOWLDataFactory();
//Create a class using the OWLDataFactory object
OWLClass animal = factory.getOWLClass(IRI.create(ontologyIRI + "#animal"));
```

Or use a PrefixManager class create a class

```
PrefixManager pm = new DefaultPrefixManager("://www.test.com/ontologies/testont.owl#");
OWLClass animal = factory.getOWLClass(":animal", pm);
```

3) The following pseudo-code creates an axiom that specifies one class as the subclass of another:-

```
//Create two classes
OWLClass animal = factory.getOWLClass(IRI.create(ontologyIRI + "#animal"));
OWLClass herbivore = factory.getOWLClass(IRI.create(ontologyIRI + "#herbivore"));

//Create the subclass axiom
OWLAxiom axiom1 = factory.getOWLSubClassOfAxiom(herbivore,animal);

//Save the axiom in the ontology by using applyChange() method of OWLOntologyManager
AddAxiom addAxiom1 = new AddAxiom(ontology, axiom1);
manager.applyChange(addAxiom1);
```

4) The following pseudo-code shows an ObjectProperty axiom :-

```
PrefixManager pm = new DefaultPrefixManager("http://www.test.com/ontologies/testont.owl");

//Create two individuals of the class
OWLIndividual lion = dataFactory.getOWLNamedIndividual(":lion", pm);
OWLIndividual cow = dataFactory.getOWLNamedIndividual(":herbivore", pm);

//Create the object property eats
 OWLObjectProperty eats = dataFactory.getOWLObjectProperty(IRI.create(animal + "#eats"));
```

```
//Create the ObjectPropertyAssertionAxiom
 OWLObjectPropertyAssertionAxiom axiom1 =
dataFactory.getOWLObjectPropertyAssertionAxiom(eats, lion, cow);
      manager.addAxiom(animal,axiom1);
```

5) The following pseudo-code shows how to create DataTypeProperty axiom :-

```
//Create a typed datatype object(of type integer)
OWLDatatype integer = dataFactory.getIntegerOWLDatatype();

//Create an untyped datatype
OWLLiteral hundred = dataFactory.getOWLLiteral(100);

//Create a MIN_INCLUSIVE restriction on a typed datatype
OWLDatatypeRestriction range = dataFactory.getOWLDatatypeRestriction(integer,
OWLFacet.MIN_INCLUSIVE, hundred);

//Create a OWLDataProperty object
OWLDataProperty tall = dataFactory.getOWLDataProperty(IRI.create(animal + "#tall"));

// Create the restriction and save it
OWLDataPropertyRangeAxiom axiom2 =
dataFactory.getOWLDataPropertyRangeAxiom(tall,range);
 manager.addAxiom(animal,axiom2);
 manager.saveOntology(animal);
```

6) The following pseudo-code shows a Hermit reasoner in action in OWL API. The reasoner checks if the ontology is consistent or not

```
//Create a OWLReasonerFactory object
OWLReasonerFactory reasonerFactory = new Reasoner.ReasonerFactory();

//Reasoner progress monitor object
ConsoleProgressMonitor progressMonitor = new ConsoleProgressMonitor();

// Create a configuration object for the reasoner to be created
OWLReasonerConfiguration config = new SimpleConfiguration(progressMonitor);

//Create a reasoner object
 OWLReasoner reasoner = reasonerFactory.createReasoner(animal,config);
  reasoner.precomputeInferences();
  //Check if ontology is consistent or not
  boolean consistent = reasoner.isConsistent();
  //Print the result
```

```
System.out.println("Consistent: " + consistent);
```

Conclusion: The OWL API has no native support for SPARQL. However, a Jena model can be created with the contents of an OWLAPI reasoner so SPARQL query answering can be done on the Jena model. OWLAPI reasoner and Jena model can be queried together but all updates should be done on the OWLAPI ontology. Semantic Web as originally envisioned, a system that enables machines to understand and respond to complex human requests based on their meaning, has remained largely unrealized but it is expected to grow in the coming years.

References:

[1] http://owlapi.sourceforge.net/index.html
[2] http://en.wikipedia.org/wiki/Semantic_Web
[3] http://www.w3.org/2007/OWL/wiki/Document_Overview
[4] http://sourceforge.net/projects/owlapi/
[5] http://owl.man.ac.uk/api/readme.html
[6] http://wonderweb.semanticweb.org/
[7] http://www.co-ode.org/
[8] http://hermit-reasoner.com/