CSc 8711
Databases and the Web

Syed Haque
shaque4@student.gsu.edu
Department of Computer Science
Georgia State University
Atlanta, GA 30303
OWL API
Outline

- INTRODUCTION
- OWL API BACKGROUND AND PHILOSOPHY
- OWL API
- EXAMPLES
- REFERENCES
Introduction – OWL API

- Familiarity with Java

- (Some) Familiarity with Semantic Web Technologies:
  - RDF
  - RDF Schema
  - OWL

- Being at least aware of the existence of:
  - Description Logics
Ontology in Computer Science

- An ontology is an engineering artifact:
  - It is constituted by a specific vocabulary used to describe a certain reality, plus
  - a set of explicit assumptions regarding the intended meaning of the vocabulary
- Thus, an ontology describes a formal specification of a certain domain:
  - Shared understanding of a domain of interest
  - Formal and machine manipulable model of a domain of interest
Building a Semantic Web

- **Annotation**
  - Associating metadata with resources

- **Integration**
  - Integrating information sources

- **Inference**
  - Reasoning over the information we have.
    - Could be light-weight (taxonomy)
    - Could be heavy-weight (logic-style)

- **Interoperation and Sharing** are key goals
Languages

- Work on Semantic Web has concentrated on the definition of a collection or “stack” of languages

- The languages provide basic machinery that we can use to represent the extra semantic information needed for the Semantic Web:
  - XML, RDF, RDF(S), OWL, ….
Why (Formal) Semantics?

- Increased formality makes languages more amenable to machine processing (e.g. automated reasoning).
- The formal semantics provides an unambiguous interpretation of the descriptions.
  - What does an expression in an ontology language mean?
  - The semantics of a language tell us precisely how to interpret a complex expression.
- Well defined semantics are vital if we are to support machine interpretability
  - They remove ambiguities in the interpretation of the descriptions.
OWL

- OWL is a language for representing Ontologies in a Web context
  - Web Ontology Language

- A W3C Recommendation
  - Since February 2004
APRIORI ALGORITHM

\[ L_1 = \{ \text{large 1-itemsets} \} \]

for \( k = 2 \) to \( L_{k-1} \neq \emptyset \) do begin

\[ C_k = \text{apriori-gen}(L_{k-1}) \]  // new candidate itemsets generated

for all transactions \( t \in D \) do begin

\[ C_t = \text{subset}(C_k, t) \]  // transaction \( t \) contains in the candidate itemsets

for all candidates \( c \in C_t \) do \( c.\text{count}++ \)

end

\[ L_k = \{ c \in C_k \mid c.\text{count} \geq \text{minsup} \} \]

end

Answer = \( L_k \)
Points from History

• Influence of frame based modelling approaches
  – Classes, slots, fillers

• Influence of logical foundations
  – Well-formed semantics
  – Inference

• Influence of Web Languages
  – RDF, RDF(S)
OWL API

- OWL allows us to describe a domain in terms of:
  - Individuals
  - Classes
  - Properties

- Plus a collection of axioms describing how these classes, individuals, properties etc. should be interpreted
Why build an OWL API?

- The use of 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 likelihood of interoperating applications.
- Ontology level objects made it easy to write code spotting “internal errors”
ASSUMPTIONS

- Primarily targeted at OWL-DL

- Java based:
  - Interfaces
  - Java reference implementation – Main memory based

- INFRASTRUCTURE – THIN OR THICK
OWL Implementation

- Modelling
- Parsing
- Serializing
- Manipulation/Change
- Inference
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.
  - Conceptually cleaner.
  - Syntax doesn’t get in the way
Overall Philosophy

- 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
Basic Data Structures

- Classes to Help
  - Create
  - Manipulate
  - Parse
  - Render
  - Reason about those structures

- At its heart, the OWL API provides data structures representing OWL ontologies

- The basic data structure represents the objects in the ontology and corresponds roughly to the abstract syntax of OWL.
Logical and Physical Mapping

- The API defines the notion of logical and physical URIs
  - Logical: the URI used to name the ontology
  - Physical: the location where the ontology was retrieved from

- An OntologyURIMapper is then used to map between physical and logical URIs
  - Allows for local copies or repositories of ontologies
Classes

- OWLOntology
- OWLClass
- OWL Entity
- OWLProperty
- OWLObjectProperty
- OWLDataProperty
- OWLAxiom
Inference

- OWLClassReasoner
- OWLConsistencyChecker
- OWLIndividualReasoner
- OWLPropertyReasoner
Reasoner Implementations

- Pellet
- FaCT++
- Hermit
Managing Ontologies

- The model data structures provide representations of the basic building blocks.
- Management and creation of ontologies is controlled by an OWLOntologyManager. This replaces OWLConnection/OWLManager in the original implementation.
- The Manager is responsible for keeping track of the ontologies and concrete formats for storage of the ontologies.
- Handles Ontology changes.
References