Knowledge Representation in the Semantic Web

Presented by
Huiyong Xiao (hxiao@cs.uic.edu)
Wei Zhang (wzhang13@uic.edu)

Outline

- **Traditional knowledge representation**
- **Current world wide web**
- **Semantic web architecture**
- **Ontology**
- **Protégé-2000 as Ontology Editor (Demo)**
- **Ontology representation language: RDF**
- **Protégé-2000 as RDF(S) Editor (Demo)**
- **Extensions of RDF(S)**

What is Ontology?

- Ontology ....

  philosophical discipline, branch of philosophy that deals with the nature and the organization of reality
- Science of Being
- Tries to answer the questions:
  - *What is being?*
  - *What are the features common to all beings?*

What are ontologies in computer science?

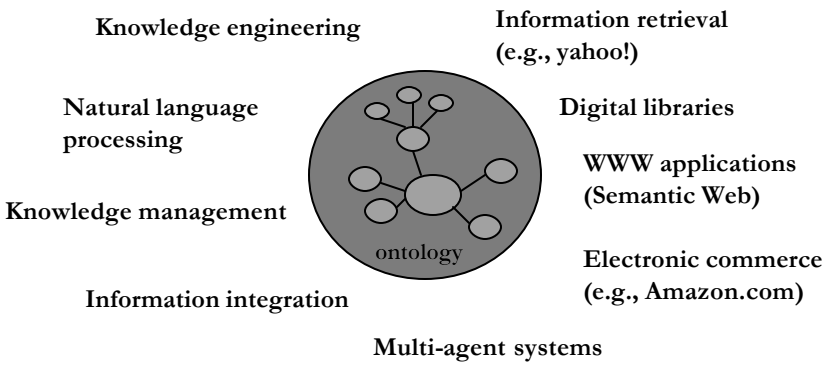
- An ontology is an explicit specification of a conceptualization [Gruber, 93]
- An ontology is a shared understanding of some domain of interest. [Uschold, Gruninger, 96]
- There are many definitions. In general, an ontology (in our sense) is
  - a **formal specification** => executable
  - of a **conceptualization of a domain** => community
  - of some part of world that is of interest => application

Why Ontology?

- Lack of a shared understanding leads to poor communication
  - People, organizations and software systems must communicate between and among themselves
- Disparate modeling paradigms, languages and software tools limit
  - Interoperability
  - Knowledge sharing & reuse

Application areas

- Ontologies have become a popular research topic in various communities



A concrete ontology example

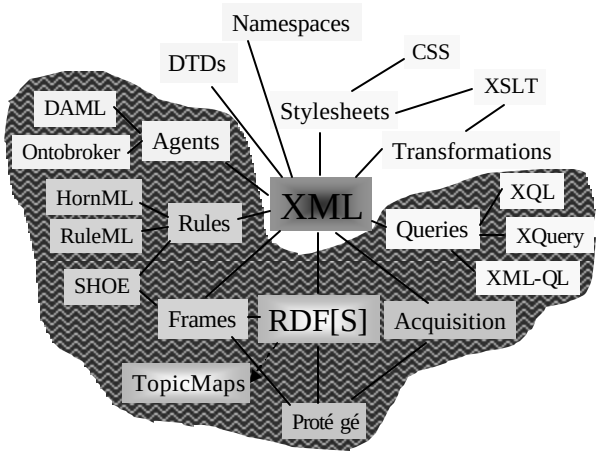
Protégé-2000 as Ontology Editor

- Protégé-2000 is an integrated software tool used by system developers and domain experts to develop knowledge-based systems
- Applications developed with Protégé-2000 are used in problem-solving and decision-making in a particular domain
- A uniform GUI (graphical user interface) whose top-level consists of overlapping tabs for compact presentation of the parts and for convenient co-editing between them.

Protégé-2000 as Ontology Editor

- **This "tabbed" top-level design permits an integration of**
  - **the modeling of an ontology of classes describing a particular subject,**
  - **the creation of a knowledge-acquisition tool for collecting knowledge,**
  - **the entering of specific instances of data and creation of a knowledge base, and**
  - **the execution of applications.**
- **A Protege knowledge base is a frame-based knowlege base. People generally don't extract rules from a Protege knowledge base. To do rule-based programming using information stored in a Protege knowledge base, you can tabs such as JessTab and Algernon.**
- **(Demo)**

Web knowledge representation



- XML- & RDF-based markup languages provide a 'universal' **storage/interchange** format for such **Web-distributed knowledge representation**

Ontology representation language

- Various kinds of (formal) languages are used for representing ontologies
  - LOOM, CyCL, F-Logic, Conceptual Graphs, Ontolingua, and KIF etc.
- Nowadays, there are more languages for expressing ontologies
  - **RDF-Schema**: Vocabualry for RDF.
  - **UML**: Unified Modeling Language
  - **OIL**: Ontology Interchange and Inference

Web Languages for knowledge capturing

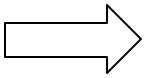
- Human knowledge is (partially) captured on the Web as *informal texts*, *semiformal documents*, and *structured metadata*
- Each kind of knowledge has its (preferred) markup language

Knowledge:	Informal	Semiformal	Metadata
Language:	HTML	XML	RDF

Address example

External Presentation:

Mary Stewart
911 S. Miller ST.
Chicago



HTML Markup:

```
<em>Mary Stewart</em>
<br>
911 S. Miller ST.
<br>
<strong>Chicago</strong>
```

Document Type Definition (DTD):

```
<!ELEMENT address  (name, street, town) >
<!ELEMENT name     (#PCDATA) >
<!ELEMENT street   (#PCDATA) >
<!ELEMENT town     (#PCDATA) >
```

XML Markup:

```
<address>
  <name>Mary Stewart</name>
  <street>911 S. Miller ST.</street>
  <town>Chicago</town>
</address>
```

Web Languages for knowledge capturing

- **XML (**E**x**tensible **M**arkup **L**anguage**):** Semiformal documents range between *non-formatted texts* and *fully formatted databases*
- **RDF (R**esource **D**escription **F**ramework**):** Structured metadata describe arbitrary *heterogeneous* Web pages/objects in a *homogeneous* manner
- Machines (e.g. search engines) can analyze XML or RDF markups better than full HTML.

XML

- XML offers new general possibilities, from which AI knowledge representation (KR) can profit:
  - Definition of self-describing data in worldwide standardized, non-proprietary **format**.
  - Structured data and knowledge **exchange** for enterprises in various industries.
  - **Integration** of information from different sources (into uniform documents).

XML

- **Key idea**: Separate **structure** from **presentation**
- **XML DTDs** or **Schemas** define document structure
- **XSL (Extensible Stylesheet Language)** specifies the document presentation
- Replace HTML with two things
  - A domain specific markup language (defined in XML)
  - A map from that markup language to HTML (defined using XSLT)

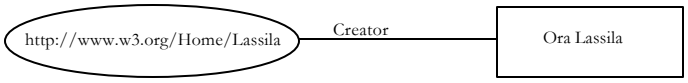
RDF – Why XML is Not Enough?

- Main advantage of using XML is reusing the parser and document validation
- Many different possibilities to encode a domain of discourse (The same semantic may have different structures)
- Leads to difficulties when understanding of foreign documents is required

==> Next step: separate **semantic** from **structure**

Creator example

"The Creator of the Resource " http://www.w3.org/Home/Lassila" is Ora Lassila



Endless encoding possibilities in XML:

```
<Creator>
    <uri>http://www.w3.org/Home/Lassila</uri>
    <name>Ora Lassila</name>
</Creator>
```

```
<Document uri=" http://www.w3.org/Home/Lassila"
    <Creator>Ora Lassila</Creator>
</Document>
```

```
<Document uri=" http://www.w3.org/Home/Lassila"  Creator="Ora Lassila"/>
```

Introduction to RDF

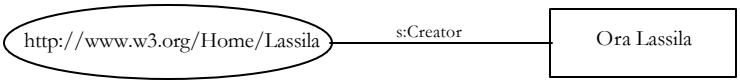
- RDF beyonds *Machine readable* to *Machine understandable*
- RDF consists of two parts
  - RDF Model (a set of triples)
  - RDF Syntax (different XML serialization syntaxes)
- RDF Schema defines the vocabularies for RDF.

RDF Data Model

- **Resources**
  - A resource is a thing you talk about (can reference)
  - Resources have URI's
  - RDF definitions are themselves Resources
- **Properties**
  - slots, define relationships to other resources or atomic values
- **Statements**
  - "Resource has Property with Value"
  - (Values can be resources or atomic XML data)
- **Similar to Frame Systems**

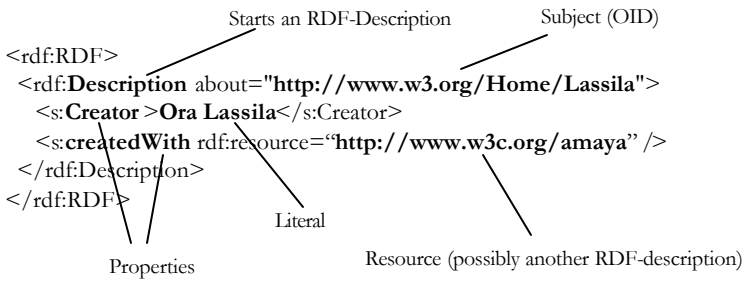
An example

- **Statement**
  - "Ora Lassila is the creator of the resource http://www.w3.org/Home/Lassila"
- **Structure**
  - **Resource** (subject) http://www.w3.org/Home/Lassila
  - **Property** (predicate) http://www.schema.org/#Creator
  - **Value** (object) "Ora Lassila"
- **Directed graph**



RDF Syntax

- Data model does not enforce particular syntax
- Specification suggests many different syntaxes based on XML
- General form:



RDF Schema (RDFS)

- RDF just defines the data model
- Need for definition of vocabularies for the data model - an Ontology Language!
- RDF schemas are Web resources (and have URIs) and can be described using RDF

Most Important Modeling Primitives

- Core Classes
  - Root-Class rdfs:Resource
  - MetaClass rdfs:Class
  - Literals rdfs:Literal
- rdfs:subclassOf - property
- Inherited from RDF: properties (rdf:Property)
- rdfs:domain & rdfs:range
- rdfs:label, rdfs:comment, etc.
- Inherited from RDF: InstanceOf (rdf:type)

Example

```
<rdfs:Class rdf:ID="Animal">
  <rdfs:label>Animal</rdfs:label>
  <rdfs:comment>
    This class of animals is illustrative of a number of
      ontological idioms.
  </rdfs:comment>
</rdfs:Class>

<rdfs:Class rdf:ID="Male">
  <rdfs:subClassOf rdf:resource="#Animal"/>
</rdfs:Class>

<rdf:Property rdf:ID="hasParent">
  <rdfs:domain rdf:resource="#Animal"/>
  <rdfs:range rdf:resource="#Animal"/>
</rdf:Property>

<rdf:Property rdf:ID="hasFather">
  <rdfs:subPropertyOf rdf:resource="#hasParent"/>
  <rdfs:range rdf:resource="#Male"/>
</rdf:Property>
```

Protégé-2000 as XML Editor

- **(Demo)**

Protégé-2000 as RDF(S) Editor

- **(Demo)**

Extensions of RDF(S) – DAML+OIL

- Ontology Language DAML+OIL: DARPA Agent Markup Language Program. (http://www.daml.org)
- Extension of RDF Schema
  - **Class Expressions** (Intersection, Union, Complement)
    - E.g., daml:intersectionOf, daml:complementOf
  - XML Schema **Datatypes**
  - **Enumerations**
  - **Property Restrictions**
    - **Cardinality** Constraints
    - Value Restrictions

Resources and References

- **Web-Ontology (WebOnt) Working Group**
  - http://www.w3.org/2001/sw/WebOnt/
- **Resource Description Framework (RDF)**
  - **http://www.w3.org/RDF**
- **RDF-Editor: Proté gé**
  - http://www-smi.stanford.edu/projects/protégé
- **RDF-Parser and APIs/Query Engines**
  - http://www-db.stanford.edu/~melnik/rdf
  - http://www.aifb.uni-karlsruhe.de/~sde/rdf
- **General Information**
  - **RDF Interest Mailing list**: www-rdf-interest@w3.org
    Archive: http//lists.w3.org/Archives/Public/www-rdf-interest
  - **SemanticWeb.org**

Resources and References

- Gruber, T. (1993). A translation Approach to Portable Ontology Specifications. Knowledge Acquisition. Vol5. 1993.
- Uschold, M.; Grüninger, M. (1996) ONTOLOGIES: Principles, Methods and Applications. Knowledge Engineering Review. Vol. 11; N. 2.