Semantic Web:
Overviews and Applications

Prof. Joongmin Choi
Intelligent Systems Laboratory
Dept. of Computer Science and Engineering
Hanyang University
http://cse.hanyang.ac.kr/~jmchoi

Contents

- Semantic Web in General
- Roles of XML and RDF in Semantic Web
- Ontologies
- Web Ontology Language (OWL)
- Semantic Web Applications
- Trends

Current Web is for Humans

- The current Web is for humans
  - The current Web represents information using
    - Natural language
    - Graphics, multimedia, page layout, etc.
  - Humans can process this easily
    - Can deduce facts from partial information
    - Can create mental association
    - Are used to various sensory information
  - Tasks often require to *combine* data on the Web
    - Hotel and travel info may come from different sites
    - Searches in different digital libraries
  - Again, humans combine these information easily

Current Web is *Not* for Machines

- Machines are ignorant
  - Partial information is unusable
  - Difficult to make sense from, eg, an image
  - Drawing analogies automatically is difficult
  - Difficult to combine information
    - Is `<foo:creator>` same as `<bar:author>` ?
    - How to combine different XML hierarchies?
- Problems
  - Search engines generates too many false hits
  - Web service is only described in terms of input and output : necessary to characterize the service in terms of *semantics*
  - Etc.

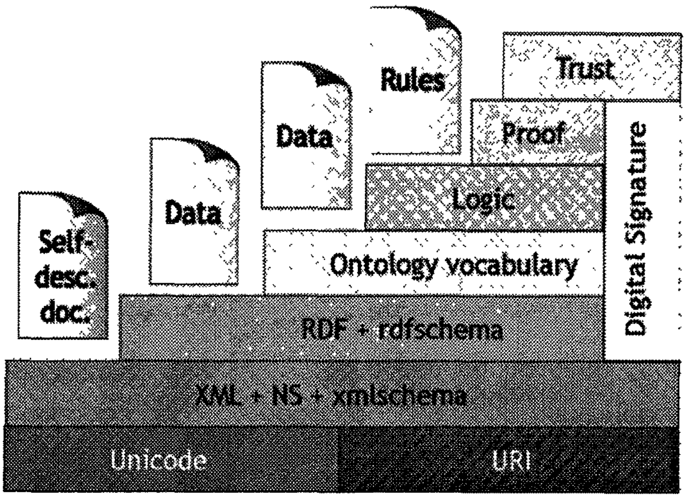
What is Needed?

- A resource should provide information about itself, called *metadata*
  - Metadata should be in a machine processable format
  - Agents should be able to *reason* about (meta)data
  - Metadata vocabularies should be defined
- To make metadata machine processable, we need
  - Unambiguous names for resources (URIs)
  - A common data model for expressing metadata (RDF)
    - .. and ways to access the metadata on the Web
  - Common vocabularies (Ontologies)

Nature of the Semantic Web

- What is the Semantic Web?
  - *A metadata based infrastructure for reasoning on the Web*
  - It *extends* the current Web (and does not replace it)
  - Information is given *well-defined meaning*
- Goals of the Semantic Web
  - To develop enabling *standards* and *technologies* designed to help machines understand more information on the Web
  - To support richer discovery, data integration, navigation, and automation of tasks

Semantic Web Layers



How XML Fits into the Semantic Web?

- XML characteristics
  - XML has a standard syntax for metadata
  - XML has a standard structure for both documents and data
- XML is the *syntactical foundation layer* of Semantic Web

```
<note>
<date>12/11/99</date>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

XML is *Not* Enough

- Limitations of XML
  - Many ways to say the same thing
    - Multiple valid structures for the same data

```
<note>
<date>
 <day>12</day>
 <month>11</month>
 <year>99</year>
</date>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

  - Not impose a common interpretation of a data
    - heading vs. title
    - price vs. cost

RDF (Resource Description Framework)

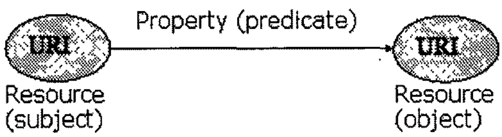
- RDF is an infrastructure that enables *the encoding, exchange and reuse of structured metadata*
- RDF is a foundation for processing metadata
  - Provides interoperability between applications that exchange machine-understandable information on the Web
- RDF is the *semantic foundation layer* of Semantic Web

RDF Model

- The metadata is a set of *statements*
- Statements can be modeled with:
  - *Resources*: an element, a URI, a literal, ...
  - *Properties*: directed relations between two resources
  - *Statements*: "triples" of two resources bound by a property
    - Usual terminology: (s,p,o) for subject, property, object
- RDF is a general model for such statements
  - ...with machine readable formats (e.g., RDF/XML, n3, Turtle, RXR)
  - RDF/XML is the official W3C format

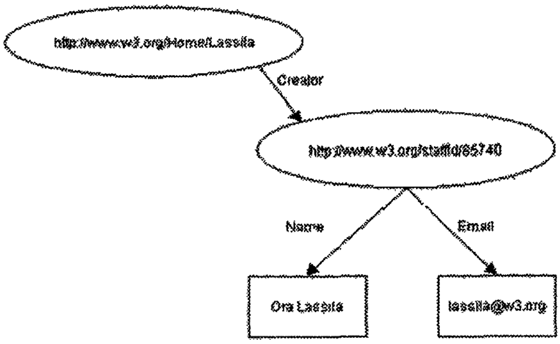
RDF is a Graph

- An (s,p,o) triple can be viewed as a labelled edge in a graph
  - A set of RDF statements is a *directed, labelled graph*
  - Both "objects" and "subjects" are the graph nodes
  - "properties" are the edges
- One should "think" in terms of graphs
  - XML or n3 syntax are only the tools for practical usage!
  - The term "serialization" is often used for encoding



RDF Graph Example

"The individual referred to by employee id 85740 is named Ora Lassila
and has the email address lassila@w3.org.
The resource http://www.w3.org/Home/Lassila was created by this individual."



Structured value with identifier

RDF/XML Serialization

"The individual referred to by employee id 85740 is named Ora Lassila
and has the email address lassila@w3.org.
The resource http://www.w3.org/Home/Lassila was created by this individual."

```
<rdf:RDF>
  <rdf:Description about="http://www.w3.org/Home/Lassila">
    <s:Creator>
      <rdf:Description about="http://www.w3.org/staffId/85740">
        <v:Name>Ora Lassila</v:Name>
        <v:Email>lassila@w3.org</v:Email>
      </rdf:Description>
    </s:Creator>
  </rdf:Description>
</rdf:RDF>
```

Containers

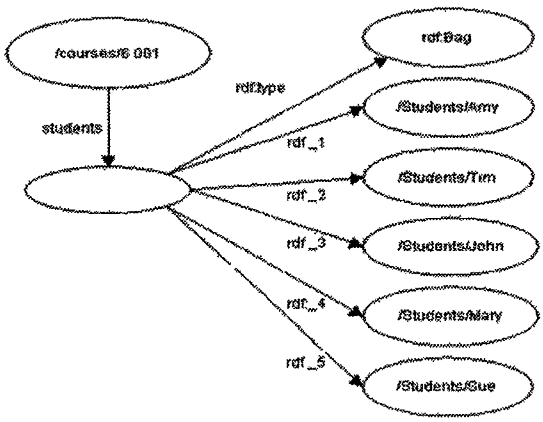
IS LAB HANYANG UNIV.

- For referring to a collection of resources
  - E.g, to say that a work was created by more than one person, or to list the students in a course, or the software modules in a package
- Three types of container objects in RDF
  - Bag, Sequence, Alternative
- RDF representation of a collection of resources
  - RDF uses an additional resource that identifies the specific collection.
  - This resource must be declared to be an instance of one of the container object types, using the *type* property.
  - The membership properties are named simply "_1", "_2", "_3", etc.

Bag Example

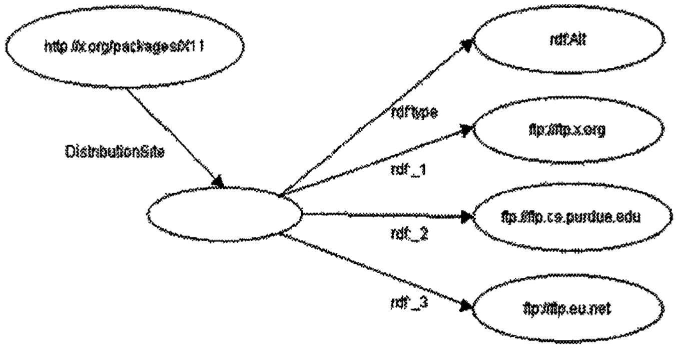
IS LAB HANYANG UNIV.

"The students in course 6.001 are Amy, Tim, John, Mary, and Sue."



Alternative Example

"The source code for X11 may be found at ftp.x.org, ftp.cs.purdue.edu, or ftp.eu.net."

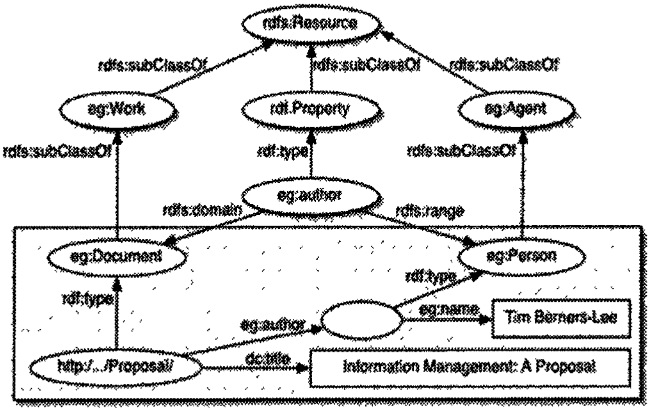


RDF Schema (RDFS)

- RDF Schemas are used to declare vocabularies
  - A schema defines the terms that will be used in RDF statements and gives specific meanings to them.
  - RDF Schema: officially "RDF Vocabulary Description Language"
- Provides a basic *type system* for use in RDF models
  - Defines resources and properties such as `Class` and `subClassOf` that are used in specifying application-specific schemas
  - A bit like defining record type for a database

RDFS Example

This example illustrates the use of the RDF Schema vocabulary for describing classes and properties, and the connection to application-level data



Ontology

- Why Ontology?
  - RDFS is useful, but does not solve all the issues
  - Complex applications may want more possibilities
  - Can a program *reason* about some terms?
    - "if <A> is left of <B> and <B> is left of <C>, is <A> left of <C>?"
    - obviously true for humans, not obvious for a program..
    - programs should be able to *deduce* such statements
  - If somebody else defines a set of terms: are they the same?
  - *Construct* classes, not just name them
  - Restrict a property range *when used for a specific class*

Ontology Definition

- An ontology is a formal, explicit specification of a shared conceptualization.
  (by T. Gruber, 1993, Studer et. al. 1998)
  - Conceptualization: an abstract model of how people think about things in the world, usually restricted to a particular subject area
  - Explicit specification: the type of concepts used and the constraints on their use are explicitly defined
  - Formal: the ontology should be machine understandable
  - Shared: an ontology captures consensual knowledge; it is not restricted to some individual but is accepted by a group

Web Ontology Language

- The Semantic Web needs a support of ontologies
  - Defines the *concepts* and *relationships* used to describe and represent an area of knowledge
  - The most typical kind of ontology for the Web has a *taxonomy* and a set of *inference rules*
- We need a *Web Ontology Language* to define
  - The terminology used in a specific context
  - More constraints on properties
  - The logical characteristics of properties
  - The equivalence of terms across ontologies
  - Etc.

OWL : W3C's Ontology Language

- A layer *on top* of RDFS with additional possibilities
- Outcome of various projects:
  - DARPA project: DAML
  - EU project: OIL
  - An attempt to merge the two: DAML+OIL
  - The latter was submitted to W3C
  - Lots of coordination with the core RDF work
  - Recommendation since early 2004

Classes in OWL

- In RDFS, you can subclass existing classes .. but, otherwise, that is all you can do
- In OWL, you can *construct* classes from existing ones:
  - Enumerate its content
  - Through intersection, union, complement
  - Through property restrictions
- To do so, OWL introduces its own `Class`.. and `Thing` to differentiate the *individuals* from the *classes*

Classes can be Enumerated

- Possible content is explicitly listed

```
<owl:Class rdf:ID="WineColor">
  <rdfs:subClassOf rdf:resource="#WineDescriptor"/>
  <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#White"/>
    <owl:Thing rdf:about="#Rose"/>
    <owl:Thing rdf:about="#Red"/>
  </owl:oneOf>
</owl:Class>
```

Union Of Classes

- Essentially, set-theoretical union

```
<owl:Class rdf:ID="Fruit">
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#SweetFruit" />
    <owl:Class rdf:about="#NonSweetFruit" />
  </owl:unionOf>
</owl:Class>
```

- Other possibilities:
  - `complementOf`, `intersectionOf`

Property Restrictions

- (Sub)classes can be created by restricting the behaviour of a property *on that class*
- Restriction may be by:
  - Value constraints (i.e. further restrictions on the range)
    - *All* values must be from a class
    - *At least one* value must be from a class
  - Cardinality constraints (i.e., how many times the property can be used on an instance?)
    - Minimum, maximum, exact cardinality
  - `owl:Restriction` defines a blank node with restrictions
    - Refer to the property that is constrained
    - Define the restriction itself

Cardinality Restriction Example

```
<owl:Class rdf:ID="Vintage">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasVintageYear"/>
      <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Property Characterization

- In RDFS, properties are constrained by domain and range
- In OWL, one can also characterize their *behaviour*
  - Symmetric, transitive, functional, etc
- OWL separates data properties
  - "datatype property" means that its range are *typed* literals

Transitive Property Example

```
<owl:ObjectProperty rdf:ID="locatedIn">
  <rdf:type rdf:resource="&owl;TransitiveProperty" />
  <rdfs:domain rdf:resource="&owl;Thing" />
  <rdfs:range rdf:resource="#Region" />
</owl:ObjectProperty>

<Region rdf:ID="SantaCruzMountainsRegion">
  <locatedIn rdf:resource="#CaliforniaRegion" />
</Region>

<Region rdf:ID="CaliforniaRegion">
  <locatedIn rdf:resource="#USRegion" />
</Region>
```

Term Equivalence/Relations

- For classes:
  - `owl:equivalentClass`: two classes have the same individuals
  - `owl:disjointWith`: no individuals in common
- For properties:
  - `owl:equivalentProperty`: equivalent in terms of classes
  - `owl:inverseOf`: inverse relationship
- For individuals:
  - `owl:sameAs`: two URI refer to the same individual (e.g., concept)
  - `owl:differentFrom`: negation of `owl:sameAs`

Ontologies are Hard

- A full ontology-based application is a very complex system
  - In fact, it is turning mathematical logic into a program
- Hard to implement, heavy to run ..
- .. and not all applications may need it!
- Three layers of OWL are defined: Lite, DL, Full
  - Increasing level of complexity and expressiveness
  - "Full" is the whole thing
  - "DL (Description Logic)" restricts Full in some respects
  - "Lite" restricts DL even more

OWL Full

- No constraints on the various constructs
  - `owl:Class` is equivalent to `rdfs:Class`
  - `owl:Thing` is equivalent to `rdfs:Resource`
- This means that:
  - `Class` can also be an individual
    - It is possible to talk about class of classes, etc.
  - One can make statements on RDFS constructs
    - Declare `rdf:type` to be functional..
  - Etc..
- A real superset of RDFS

OWL Description Logic (DL)

- OWL DL characteristics
  - `owl:Class`, `owl:Thing`, `owl:ObjectProperty`, and `owl:DatatypeProperty` are *strictly separated*
    - A class *cannot* be an individual of another class
  - No mixture of `owl:Class` and `rdfs:Class` in definitions
    - Essentially: use OWL concepts only!
  - No statements on RDFS resources
  - No characterization of datatype properties possible
  - No cardinality constraint on transitive properties
  - *GOAL: maximal subset of OWL Full for which a decidable reasoning procedure is realizable*

OWL Lite

- OWL Lite characteristics
  - All of DL's restrictions, plus some more:
    - Class construction can be done only through:
      - intersection
      - property constraints
  - *GOAL: provide a minimal useful subset, easily implemented*
    - Simple class hierarchies can be built
    - Property constraints and characterizations can be used

Description Logic

- DL characteristics
  - An area in knowledge representation
  - Express and reason with complex definitions of, and relations among, objects and classes
  - Designed to focus on categories and their definitions
  - Inference Tasks
    - ***Subsumption***: checking if one category is a subset of another based on their definitions
    - ***Classification***: checking if an object is a subset of another based on their definitions
  - OWL DL is an embodiment of a Description Logic

Some Tools

- (Graphical) Editors
  - IsaViz
  - Protégé 2000
  - Further info on OWL tools at: http://www.w3.org/2001/sw/WebOnt/impls
- Programming environments
  - Jena 2: OWL reasoning
  - SWI-Prolog: RDF/OWL framework in Prolog
- Validators
  - For RDF: http://www.w3.org/RDF/Validator/
  - For OWL: http://owl.bbn.com/validator/

Semantic Web Applications

- Large number of applications emerge
  - Some applications use RDF only
  - Others use ontologies, too
    - Huge number of ontologies exist, using proprietary formats
    - Converting them to RDF/OWL will be a major task
    - But it will be worth it!
- SWAD-Europe survey:
  - URI: http://www.w3.org/2003/11/SWApplSurvey
  - Lists more than 50 applications in 12 categories..

SW Application Examples

- Dublin Core
  - Vocabularies for distributed Digital Libraries
  - One of the first metadata vocabularies in RDF
  - URI: http://www.dublincore.org
- Data integration
  - Achieve semantic integration of corporate resources or different databases
  - RDF/RDFS/OWL based vocabularies as an "interlingua" among system components
  - Boeing example: http://www.cs.rutgers.edu/~shklar/www11/final_submissions/paper3.pdf
  - Similar approaches: Artiste project, MITRE Corp.,...

SW Application Examples (cont.)

- Sun's SwordFish
  - Sun provides assisted support for its products, handbooks, etc
  - Public queries go through an internal RDF engine for:
    - Sun's White Papers collection (http://www.sun.com/servers/wp.html/)
    - Sun's System Handbooks collection (http://sunsolve.sun.com/handbook_pub)
- XMP
  - Adobe's tool to add RDF-based metadata to *all* their file formats
    - Eg, Photoshop in Creative Suite
    - http://www.adobe.com/products/xmp/main.html

SW Application Examples (cont.)

- Web Content Syndication (RSS)
  - Can be used to specify the *important* content of a page
  - There is a Yahoo discussion group and (non-W4C) working group
  - URI: http://purl.org/rss
  - Widely used in the weblog world!

SW Application Examples (cont.)

- Web Services Descriptions
  - Mapping of WSDL1.2 to RDF
  - Web Choreography development in terms of RDF/OWL
    - Initiative already exist, e.g., OWL-S or WSMO
- Gene Ontology Consortium
  - Controlled vocabularies to describe aspects of gene products
  - URI: http://www.geneontology.org
- OntoWeb
  - Ontology-based information exchange for knowledge management and electronic commerce
  - URI: http://ontoweb.aifb.uni-karlsruhe.de

SW Application Examples (cont.)

- Baby CareLink
  - Center of information for the treatment of premature babies
  - Provides an OWL service *as a Web Service*
    - Combines disparate vocabularies like medical, insurance, etc
    - Users can add new entries to ontologies
    - Complex questions can be asked through the service
  - ***Perfect example for the synergy of Web services and the Semantic Web!***

Semantic Web Trends

- DAML+OIL is already the most used ontology language ever!!
  - 3.5M statements on 25,000 web pages
- Gaining acceptance by web players
  - Semantic Web Track being offered at WWW 2002
  - 3x more people attended WWW2002 Developer Day on SW than attended KR
- Significant (international) government support
  - US DARPA/NSF; EU IST Framework 5,6
  - Japan, Germany, Australia considering significant investments
  - US National Cancer Institute to publish cancer vocabulary in DAML+OIL

Conclusions

- It is no longer a question of whether the semantic web will come into being, it is already here!
- We're already well past the starting gate
  - Web ontologies, term languages, "shims" to DB and services, research in proofs/rules/trust
  - Standardization providing a common denominator for KR researchers as well as web developers
  - Small companies starting to form, Big companies starting to move
- The current environment is open, encouraging, moving fast, and exciting