G52IWS: The Semantic Web

Chris Greenhalgh
2007-11-10

Contents

• Introduction to the Semantic Web
• Semantic Web technologies
  – Overview
  – RDF
  – OWL
• Semantic Web Services
• Concluding comments

See "Developing Semantic Web Services", Alesso et al. 2004. (eBook) And W3C semantic web activities

The Semantic Web

• Current web
  – HTML pages for human consumption (reading)
  – Not machine processable
  – Simple text searching only
• Semantic Web
  – "...allows software agents to analyze the Web on our behalf, making smart inferences that go beyond the simple linguistic analyses performed by today's search engines."
  – Alesso, H. Peter. Developing Semantic Web Services, p.182
  – E.g. "find me biographies for friends of Chris Greenhalgh"

Semantic web elements

• "For the Semantic Web to function, computers must have access to structured collections of information and sets of inference rules that they can use to conduct automated reasoning, ..." The objective of the Semantic Web, therefore, is to provide a framework that expresses both data and rules for reasoning from a Web based knowledge representation."
  – Ibid. pp39-40
• "The goal of the Semantic Web is to provide a machine-readable intelligence that would come from hyperlinked vocabularies that Web authors would use to explicitly define their words and concepts."
  – Ibid. p166

Semantic Web Services

• Current Web services
  – XML documents as inputs and outputs
  – Syntax specified by XML Schema & WSDL
  – Still not directly "meaningful" to an application
    - Programmer has the semantics in their head and writes the code to manipulate the information appropriately, e.g. what to use an "<address>" element for
• Semantic Web Services
  – Allow web services to interoperate at a semantic level even if their detailed syntax/types differ
  – Allow software agents to (a) find and (b) make use of web services with no prior knowledge of their particular syntax/types

Semantic Web technologies

Wikimedia commons. C.f. "Developing Semantic Web Services", fig 6-1
Resource Description Framework (RDF)

- W3C standard, Feb. 2004
  - http://www.w3.org/TR/rdf-syntax-grammar/
  - “The Resource Description Framework (RDF) is a general-purpose language for representing information in the Web.”
- Existed since 1997

XML and meaning

- E.g.
  - <book>
    <title>War and Peace</title>
  </book>
- May well mean
  - “the book has the title ‘War and Peace’”
  - But this is not inherent in the XML or the XML schema
    - A computational agent would not be able to work this out by itself
    - and what is a “book” or a “title”, in any case

XML and structure

- Tree-structured document
  - Named elements
  - Optional cross-references
- Only syntactic relationships possible between element definitions
  - i.e. containment and specific type relationships

RDF model

- RDF is a simple knowledge representation language
- Comprising statements with exactly one
  - Subject – the resource about which the statement is made, which is identified by a URI or blank node ID
  - Predicate – identifies the property of the subject about which the statement is made by a URI
  - Object, which may be a
    - Literal value, such as an simple XSD datatype value or
    - another resource, identified by URI or Blank node ID
**RDF statements**

- E.g.
  - "A thing [subject] has property [predicate] with a specific value [object]"
  - "[subject] has [predicate] [object]"
  - (subject, predicate, object)
- Specific example:
  - "A thing <the book> has property <title> with a specific value 'War and Peace'"
  - "<the book> has <title> 'War and Peace'"
  - (<the book>, <title>, 'War and Peace')

**RDF graph structure**

- A set of RDF statements describe an information graph:
  - **Nodes** are subjects or objects, and are therefore
    - URIs
    - Blank (anonymous) nodes or
    - Literal values (which are never subjects)
  - **Arrows** are statements, which are labelled by the predicate

**Example**

- The book → title → ‘War and Peace’

**More examples**

- RDF/XML is an XML serialisation of RDF
  - Valid XML
  - Encodes a set of RDF statements
- E.g.
  - `<?xml version="1.0"?>
    <rdf:RDF
      xmlns:rdf=*
      xmlns:dc="http://purl.org/dc/elements/1.1/"
      xmlns="http://www.w3.org/1999/02/22rdf-syntax-ns#">
      <rdf:Description
        rdf:about="http://www.amazon.com/books">
        <dc:title>War and Peace</dc:title>
      </rdf:Description>
    </rdf:RDF>

**Other RDF serialisations**

- Notations3
- N3
- N-triples
- E.g.
  - `<#book> <#title> “War and Peace.”`
**rdf:type Property**

- RDF defines a standard property rdf:type
  - (rdf: is the namespace [http://www.w3.org/1999/02/22-rdf-syntax-ns#](http://www.w3.org/1999/02/22-rdf-syntax-ns#))
    - It's subject is any node
    - It's object is the node which represents the class of the subject node
  - E.g.
    - `<book1> <rdf:type> <Book>`.
    - => `<book1>` is a member of the class `<Book>`, i.e. `<book1>` is a `<Book>`

**Vocabulary**

- A set of agreed URLs to be used as predicates and classes in RDF
  - E.g. Dublin core, a set of predicates for describing publication-related metadata
    - Unique namespace makes predicates unique
      - xmlns:dc="http://purl.org/dc/elements/1.1/"

**RDF Schema**

- RDF encoding of definitions of classes and properties in terms of their relationships
  - rdfs:Class is the class of classes
    - i.e. anything with `<rdf:type>` of `<rdfs:Class>` is a class
  - rdfs:Property is the class of properties
  - rdfs:subClassOf is the property which relates a sub-class to its super-class (see over)
  - rdfs:subPropertyOf is the property which relates a sub-property to super-property

**Sub-class example**

- E.g.
  - `<my:Computer> <rdf:type> <rdfs:Class>`
    - i.e. Computer is a Class
  - `<my:Laptop> <rdfs:subClassOf> <my:Computer>`
    - i.e. Laptop is a subclass of Computer
    - i.e. every thing which is a Laptop is also a Computer
    - i.e. every Laptop is a Computer
    - E.g. `<my:PC1> <rdf:type> <my:Laptop>` => `<my:PC1> <rdf:type> <my:Computer>`

**Sub-property example**

- E.g.
  - `<my:Size> <rdf:type> <rdf:Property>`
    - i.e. Size is a Property
  - `<my:Length> <rdfs:subPropertyOf> <my:Size>`
    - i.e. Length is a sub-property of Size
    - i.e. every Length value is also a Size value
      - E.g. `<my:PC1> <my:Length> "10cm"` => `<my:PC1> <my:Size> "10cm"`

**Reification**

- In RDF a statement can be re-ified, i.e. made into a thing in itself
- All the usual RDF facilities can then be used to describe this statement
  - E.g. `<the book> <title> 'War and Peace'` => `<statement1> <subject> <the book> <statement1> <predicate> <title> <statement1> <object> 'War and Peace'`
  - And hence (for example):
    - `<statement1> <author> <bob.jones>`
    - `<statement1> <created> '20071110T211300Z'`
RDF capabilities and limitations
- Syntax for representing semantics of data
- Means to describe relationships between resources
- Limited expressive power, e.g.
  - No properties of properties
  - No necessary & sufficient conditions for class membership
  - No equivalence or disjointness of classes

Web Ontology Language (OWL)
- W3C standard, Feb 2004
  - http://www.w3.org/TR/owl-ref/
  - “The Web Ontology Language OWL is a semantic markup language for publishing and sharing ontologies on the World Wide Web. OWL is developed as a vocabulary extension of RDF (the Resource Description Framework) and is derived from the DAML+OIL Web Ontology Language.”
- i.e. builds on RDF Schema but has more expressiveness

Ontologies
- “An ontology defines the terms used to describe and represent concepts. Ontologies are used to describe concepts as well as their inter-relationships... For example...
  - A paperback is a type of book.
  - A book is a written document.
  - Documents have writers.
  - Writers are humans.
  - Humans read paperbacks.


Ontology concepts
- Classes
- Relationships between classes
- Properties of classes
- Constraints on relationships between classes and the properties of the classes

Information in an OWL document
- Class hierarchy – defines class/subclass relationships
- Synonym – identifies equivalent classes and equivalent properties
- Class association – maps classes to one another through use of a property (domain/range)
- Property metadata
- Class definition – specifies the composition of a class

OWL versions
- OWL Lite
  - Classification hierarchy and simple constraints, e.g. 0 or 1 cardinality
- OWL DL (Description Logic)
  - All constructs, but with some restrictions
  - Maximum expressiveness with computational completeness & decidability
- OWL Full
  - Maximum expressiveness but no guarantees of decidability
  - E.g. a class can be both a collection of individuals and an individual in its own right
OWL class descriptions

- Describe a class as one of
  - a class identifier (a URI reference)
  - an exhaustive enumeration of individuals that together form the instances of a class
  - a property restriction
  - the intersection of two or more class descriptions
  - the union of two or more class descriptions
  - the complement of a class description

OWL and RDFS

- Share some common concepts
  - Classes, subclass relationships, properties
- OWL Lite/DL have some restrictions compared to RDFS
  - owl:Class is not rdfs:Class except in OWL Full
- OWL has extra expressiveness supporting additional reasoning
  - E.g. boolean combination of class expressions, property restrictions, property types (e.g. transitive, inverse)

Other semantic web technologies

- SPARQL
  - Query language for RDF
- Rules Interchange Format
  - Specifying inference rules for RDF

Semantic Web Services

- Using ontologies and computational reasoning to support automated use of web services
  - In particular…

Semantic web service possibilities for automation

- Discovery - Locating Web Services.
- Invocation - Execution of service by an agent or other service.
- Interoperation - Breaking down interoperability barriers and automatic insertion of message parameter translations.
- Composition - New services through automatic selection, composition and interoperation of existing services.
- Verification - Verify service properties.
- Execution Monitoring - Tracking the execution of composite tasks and identifying failure cases of different execution traces.

Semantic web service requirements

- Metadata in service descriptions that is sufficient and computational useful to e.g.
  - Map between semantically equivalent data types
  - Map between semantically equivalent operations
  - Map between semantically equivalent services
  - Manipulate and exploit descriptions of composite services

"Developing Semantic Web Services", chapter 10
OWL-S

- OWL-S (Service): Semantic Markup for Web Services
- Submitted to W3C Nov 2004 but not standardised
  - http://www.w3.org/Submission/OWL-S/
- Provides an ontology for this service-describing metadata

Service profile

- High-level description of service and provider
- Used to request advertise services
- Includes:
  - Human-readable description
  - Link to service provider
  - Specification of functionalities
    - Process IOPEs (see later)
    - Functional attributes
      - E.g. geographical location, quality, service type

Process modelling

- Found in profile (for discovery) and model (for interaction)
- Has information transformation element:
  - Inputs
  - Outputs – both parameters
- Has state change element
  - Preconditions (NB not parameters as in text)
  - Effects
- Together referred to as “IOPEs”

Service Model

- Detailed view of service as a process
- Including option of decomposition through expanding composite (multi-part) services into their component parts
  - With various concurrency/choice structures
Notes

- Atomic processes typically correspond to single Web Service operations
  - See following notes
- Composite processes correspond to possible sequences of related operations
  - E.g. in an extended business process such as placing and tracking an order

Service Grounding

- Maps abstract service profile and model to protocol-specific details
  - E.g. to WSDL services, operations and messages, and to XML schema types and elements

Mapping between OWL-S and WSDL

Example: book ordering web service

- Suggested ontology development process:
  - Describe individual operations as atomic processes
  - Describe the groundings for each
  - Describe their composition as a composite book ordering process
  - Describe it as a simple overall process (optional, for profile)
  - Provide a declarative description for the profile

"Developing Semantic Web Services", chapter 11

Uses

- Service profile
  - Supports automatic discovery
- Service model
  - Supports automatic composition
- Service grounding
  - Supports automatic invocation
Notes

- SAWSDL
  - gives mechanisms to associate semantic annotations (e.g. ontology classes) with WSDL elements
  - Allows XML schema data elements/types to be mapped to/from an more abstract "semantic" layer
- Sidesteps use of any particular ontology or ontology technology
  - Such as OWL or OWL-S
  - Presumes "less ambitious" local establishment of ontologies for specific purposes?!
  - And/or human-in-the-loop resolution of (e.g.) conflicts

General Note

- The "Developing Semantic Web Services" text presents a strongly "evangelistic" case for machine intelligence and semantic web services
  - Practically achievable and useful automated support in relation to web services may be much simpler
    - see SAWSDL

Simpler examples of the semantic web…

- Without attempting to support extensive automated reasoning Semantic web technologies still allow useful distributed data integration
  - Ontologies are more easily extended and distributed than (e.g.) relational database schemas
  - RDF allows more flexible graph-oriented queries (and inference) than (e.g.) relational queries (such as SQL)

E.g. the Gene Ontology

- a simple (but large) ontology of concepts connected with gene products
  - Allows new levels of consistency and interoperability between gene-related biology databases
E.g. RDFa

- = “RDF attributes”
- W3C working draft
  - http://www.w3.org/TR/xhtml-rdfa-primer/
- Approach to dual-purposing XHTML pages to also present an RDF interpretation of their contents
  - Achieved by including additional XML attributes in the page elements
  - Which add the “semantics” to the raw structure and information already present in an XHTML page by linking to ontology classes & properties