Introduction to The Semantic Web – Part I

By Shasha(Amy) Liu
WWW history

- Web 1.0
- Web 2.0
- The Semantic Web
Web 1.0

“Web 1.0 is the first state of World Wide Web which was in the basic Read Only hypertext system.”

- Tim Berners-Lee, 1989

- Web 1.0 included most websites in the period between 1994 and 2004
Web 2.0 (1)

-- where we are now

“Web 2.0 is a set of economic, social, and technology trends that collectively form the basis for the next generation of the Internet—a more mature, distinctive medium characterized by user participation, openness, and network effects.”

- O'Reilly Media and MediaLive International popularized the term
- Google is now seen as the torch bearer of the term by the media
Web 2.0 (2)

Web 1.0 vs. Web 2.0
Web 2.0 (3)

Web 1.0 vs. Web 2.0

- Content distribution
- Organizational identity
- Binary relationship
- Passive reading
- Focus on companies
- Imparting information
- Content created by web coders
  ex:- websites

- Content collaboration
- Individual identity
- Communal relationship
- Active writing
- Focus on communities
- Fastering conversation
- Content created by everyone
- Web services architecture
  ex:- blogs, facebook, wikipedia, you tube.
Web 2.0 (4)

Technologies used by web 2.0

A web 2.0 website may usually feature a number of following techniques:

• Rich Internet application techniques, Ajax, Adobe Flash, Flex, Nexaweb, OpenLaszlo and Silverlight and many more.

• Cascading Style Sheet, CSS

• Organization and collection of data in RSS

• Clean and meaningful URLs

• Excessive use of folksonomies (in the form of tags or tagclouds)

• Use of wiki software

• Mashups (A mix up of content and Audio usually from different musical style)
Web 2.0 (5)

What is AJAX?

- Ajax is not a technology in itself
- Shorthand for **Asynchronous JavaScript and XML**
- Removes the need to reload entire web page each time the user makes a change. Increase the web page's interactivity, speed, and usability.
- XML is used as the format for transferring data between the server and client. XML files may be created dynamically by some form of server-side scripting
- The Ajax technique uses a combination of: **XHTML** and JavaScript.
Web 2.0 (6)

How AJAX works?

See http://www.developersdex.com/gurus/articles/845.asp
Web 2.0 (7)
Two Sides of Web 2.0

- Two-Way Communication
  - Creates conversations between people
  - Examples
    - Blogs
    - Wikis
    - Photo and video sharing
    - Social networks

- Web Services
  - Software online
    - Gmail
    - Google Docs
  - Mashups
    - Photos overlaid on Google Earth
  - Embed services
    - Google custom search engine, Google calendar, etc.
The Semantic Web (1)

- Although the existing web is powerful, it is still a places where
  - computers do the presentation (easy) and
  - people do the linking and interpreting (hard).

- Why not get computers to do more of the hard work?

Machine-to-human, not machine-to-machine
The Semantic Web (2)

- Tim Berners-Lee’s original vision of the Web was much more ambitious than the reality of the existing Web:
  
  “... a goal of the Web was that, if the interaction between person and hypertext could be so intuitive that the **machine-readable** information space gave an accurate representation of the state of people’s thoughts, interactions, and work patterns, then **machine analysis** could become a very powerful management tool, seeing patterns in our work and facilitating our working together through the typical problems which beset the management of large organizations.”

- Tim Berners-Lee (and others) have since been working towards realising this vision, which has become known as the **Semantic Web**
  - E.g., see article in May 2001 issue of Scientific American
The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in co-operation.

[Berners-Lee et al., 2001]
The Semantic Web (4)

- Architecture Requirements
  - Extensibility
    - Each layer should extend the previous one(s)
  - Support for data interchange
    - Using data from one source in other applications
  - Support for ontology description with different complexity
    - Including rules
  - Support for data query
  - Support for data provenance and trust evaluation

The semantic web architecture
See: http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html
The Semantic Web (5)

- UNICODE is the standard international character set
  - E.g. used to encode the data in the repository
- Uniform Resource Identifiers (URIs) identify things and concepts
  - E.g. used to identify resources on the Web and in the repository
- eXtensible Markup Language (XML) is a markup language used for data exchange
  - E.g. format that can be wrapped into RDF and imported into the repository

A refined SemWeb architecture
The Semantic Web (6)

- Resource Description Framework (RDF) is the HTML of the Semantic Web
  - Simple way to describe resources on the Web
  - Based on triples <subject, predicate, object>
  - Various serializations, including one based on XML
  - A simple ontology language (RDFS)
  - E.g. language used to store the data in the repository

- Web Ontology Language (OWL) is a more complex ontology language than RDFS
  - Layered language based on DL
  - Overcomes some RDF(S) limitations
  - E.g. ontology language used to define the schemas used in the repository

A refined SemWeb architecture
The Semantic Web (7)

- **SPARQL**
  - Query language for RDF triples
  - A protocol for querying RDF data over the Web
  - E.g. language used to query the repository from the user interface

- **Rule languages** (esp. Rule Interchange Format RIF)
  - Extend ontology languages with proprietary axioms
  - Based on different types of logics
    - Description Logic
    - Logic Programming
  - E.g. used to enable reasoning over data to infer new knowledge

A refined SemWeb architecture
The Semantic Web (8)

Unifying logic
- Bring together the various ontology and rule languages
- Common inferences, meaning of data

Proof
- Explanation of inference results, data provenance

Trust
- Trust that the system performs correctly
- Trust that the system can explain what it is doing
- Network of trust for data sources and services
- Technology and user interface

Many open problems, topics for future research

A refined SemWeb architecture
Bottom Two Layers of the Semantic Web

- Unicode + URI
- XML
Unicode

- Unicode is the universal standard encoding system and provides a unified system for representing textual data.
- One million characters can be encoded to specify any character in any language without a single escape sequence or control code.

$ Å Ð Æ Œ ή ½ ♥ Ж ญ
U+0024 U+00C5 U+0139 U+00C6 U+03AE U+0643 U+215D U+2665 U+0416 U+0E0D
URI

**URI = Uniform Resource Identifier:**

- a compact string of characters used to identify or name a resource.
- The URL to a web site (e.g. http://www.uga.edu) is a popular example of a URI.
  - The URL is a type of URI that does provide a way to get information about a resource, or perhaps to retrieve the resource itself
  - But the URI may **or may not** provide a way for your computer to get more information about that resource.
- The URI is the foundation of the Web. While nearly every other part of the Web can be replaced, the URI cannot: it holds the rest of the Web together
• **Background for XML**
  ◦ An Extensible Markup Language (XML) document describes the *structure of data*
  ◦ XML and HTML have a similar syntax ... both derived from SGML (Standard Generalized Markup Language)
  ◦ XML has no mechanism to specify the format for presenting data
  ◦ An XML document resides in its own file with an `.xml` extension

Figure 1. A bibliography entry (a) in HTML and (b) in XML. The HTML description is layout oriented, while the XML description is structure oriented.
XML(2)

- **XML Syntax**
  - XML is case sensitive
  - An XML file starts with a **prolog**
  - Every document must contain a root element
  - Elements must be properly nested
  - Attribute values must have quotation marks
  - … …

Legal Building Blocks of XML

- A Document Type Definition (DTD) allows the developer to create a set of rules to specify legal content and place restrictions on an XML file.

- An XML document that conforms to the rules within a DTD is said to be **valid**.
Why Use a DTD?

- A single DTD ensures a common format for each XML document that references it.
- An application can use a standard DTD to verify that data that it receives from the outside world is valid.
- A description of legal, valid data further contributes to the interoperability and efficiency of using XML.
Some Example DTD Declarations

- To specify that an element must have a single child element, include the element name within the parenthesis.
- An element can have multiple children. A DTD describes multiple children using a sequence, or a list of elements separated by commas. The XML file must contain one of each element in the specified order.

Figure 3. A DTD for the bibliography example. The DTD defines a grammar for documents.
XML Schema

- A grammar definition language
  - Like DTDs but better
    - Uses XML syntax
    - Defined by W3C

- Primary features
  - It allows the user to define datatypes
  - More powerful content models
    - e.g. namespace-aware, type derivation, etc…

The complexType element indicates a complex datatype associated with the nonterminal element BOOK and consisting of other elements and attributes.

The user can simulate the +, *, and ? of DTDs or regular expressions using the minOccurs and maxOccurs attributes.

simpleTypes, such as string, decimal…

Figure 8. The XML Schema for the bibliography in Figure 2. Like a DTD, the XML Schema defines a grammar for the document. However, XML Schema uses XML syntax and is more expressive.
Extending Capabilities: extensions to XML let users link more than one source document to more than one target document
  ◦ Namespaces
  ◦ Addressing and linking

Extending Capabilities -- Namespaces
  ◦ Using namespaces avoids name clashes
    • Example 1: albums have a <name>
    • Example 2: customers have a <name>
    • How do you differentiate between the two?
Extending Capabilities -- Namespaces --Contd…

- In an XML document, namespaces are declared using the `xmlns` attribute.
  - Named prefix
    - `<BIB xmlns:mybib="http://www.myserver.net/">`
  - Default prefix
    - `<BIB xmlns="http://www.myserver.net/">`

- Namespaces can be defined in any element.
- The scope of namespaces is the element in which they are defined, plus all its descendants.
- To avoid confusion, you’d better define all namespaces within the root element and use unique prefixes.
Extending Capabilities -- Address and Linking

- HTML uses URL to link documents
  - URLs point only to a document, and it is not possible, for example, to address the fifth items in the second list in a document directly
  - HTML links are one-way, and external link definitions are not possible

- XML extends HTML’s linking capabilities with
  - XLink: describes how two documents can be linked
  - XPointer: enables addressing individual parts of an XML document, and
  - XPath: is used by XPointer to describe location paths
Extending Capabilities -- Address and Linking -- Contd…

- **XPath**: is used by XPointer to describe location paths
  - A Location path consist of location steps, which in turn consist of
    - An axis
    - A node test
    - A predicate

- E.g. `child::AUTHOR[position()<3]/attribute::id`
  - The expression above consists of two location steps, evaluated left to right, one starting with `child` and the other with `attribute`.
  - It describes the *id* attributes of the first two AUTHORs within some externally defined initial context.
  - The axis (the part before the ::) helps you navigate through the document. Possible axis values include `child`, `attribute`, `parent`, and `following-siblings`.
  - The node test can be a tag or attribute name (AUTHOR in the example) or the *wildcard* `*`.
  - The predicate in the square brackets defines the filter.
XML(11)

- Extending Capabilities -- Address and Linking -- Contd…
  - **XPointer**: uses **XPath** to define fragment identifiers
    - A fragment identifier can be
      - the value of an attribute of type id
      - a sequence of numbers (like /1/4/2/10), or
      - A sequence of Xpointer expressions (xpointer(…))
    - E.g
      - http://www.myserver.net/#xpointer(//BOOK/AUTHOR[position()=1])
        - In the XPointer expression above (in parentheses), // is an abbreviation for the **descendant-or-self** axis
        - whereas AUTHOR comes without an axis: child() is the default.
    - XPointer also defines the initial context for the XPath expression
    - The most important part of the initial context is the context node, which is the root node of the document specified by the fragment identifier.
Extending Capabilities -- Address and Linking -- Contd…

- **XLink:** describes how two documents can be linked
  
  A simple link is very similar to an HTML link

```xml
<AUTHOR xmlns:xlink=http://www.w3.org/1999/xlink
  xlink:type="simple"
xlink:role="don_~knuth_homepage"
xlink:show="embed"
xlink:actuate="onLoad">
Donald Knuth </AUTHOR>
```

**show** specifies what is to be done with the linked document when it is loaded. For example, show=“embed” dictates that the linked document is to be embedded in the current document. Other values for show are replace, new, and undefined.

**role** indicates the purpose of the linked document (or, more accurately, the linked element) in the linking document.

**actuate** specifies when the action indicated by show should occur. Values for actuate are onLoad, indicating that the linked document is to be embedded while the current document is loading; onRequest, indicating that it is to be embedded when requested; or undefined.
XML(13)

- Extending Capabilities -- Address and Linking -- Contd...
  - **XLink**: describes how two documents can be linked
    - Extended links connect more than two documents (generally called resources) – that is, more than one source document or more than one target document

```xml
<BOOK>
  <AUTHOR>Ullman, J. D. </AUTHOR>
  <TITLE>Principles of Database and Knowledge Base Systems, Vol. 1</TITLE>
  <LINKS xmlns:xlink="http://www.w3.org/1999/xlink"
xlink:type="extended"
xlink:title="My Book World">
    <MYPAGE xlink:title="My Page on Ullman's Book">
      My Page on Ullman's Book
    </MYPAGE>
    <AMAZON xlink:type="locator"
    xlink:role="amazon"/>
    <BARNES_AND_NOBLE xlink:type="locator"
xlink:href="http://shop.barnesandnoble.com/booksearch/...
    xlink:role="barnes_and_noble"/>
  </LINKS>
</BOOK>
```

Figure 5. An extended Xlink. Extended XLinks let you define collections of links.
Limitations for semantic markup

XML *per se* makes no commitment on:

- **Domain specific ontological vocabulary**
  - Which words shall we use to describe a given set of concepts?
- **Ontological modelling primitives**
  - How can we combine these concepts, e.g. “car is a-kind-of (subclass-of) vehicle”

⇒ requires pre-arranged agreement on vocab and primitives

Only feasible for closed collaboration

- agents in a small & stable community
- pages on a small & stable intranet
- **not for sharable Web-resources**
Reference of Part I

- Design Patterns and Business Models for the Next Generation of Software
  by Tim O'Reilly  09/30/2005
- Web 2.0 From Wikipedia, the free encyclopedia
  http://en.wikipedia.org/wiki/Web_2.0
- How To Create Web2.0 Applications Using AJAX And clientside HTTP Requests
  http://www.w3.org/DesignIssues/Architecture.html
- http://www.w3.org/TR/webarch/
Practice for Part I

- Write an XML schema specification for a simple document that lists stock brokers with the accounts that they handle and a separate list of the client accounts. The information about the accounts includes the account ID, ownership information, and the account positions (i.e., stocks held in that account). To simplify the matters, it suffices, for each account position, to list the stock symbol and quantity. Use ID, IDREF, and IDREFS to specify referential integrity.

Solution:

```xml
<schema xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:brk="http://somebrokerage.com/documents"
  targetNamespace="http://somebrokerage.com/documents">
  <element name="Brokerage">
    <complexType>
      <sequence>
        <element name="Broker" type="brk:brokerType"
          minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </complexType>
  </element>
</schema>
```