Knowledge Base (1)
—Semantic Web(1)—

Masaharu Yoshioka
Plan of this class

- **Topics**
  - Semantic web and ontology
  - Information Retrieval
  - Information extraction from Texts and its application
  - Database for large scale data

- **Resume will be distributed from website (will be uploaded at the afternoon of the day before (Tuesday and Friday))**
  
  http://www-kb.ist.hokudai.ac.jp/~yoshioka/kb/

- **Contact mail address:**
  yoshioka@ist.hokudai.ac.jp
Guidance for this year’s class

- Lecture by Lecturer David Fisher from University of Massachusetts
  - July 1st and 3rd
  - Related event
    - Indri Workshop (one day workshop to learn how to use Indri search engine software) by Lecturer David Fisher

- For each class, we will ask to submit a short report at the end of the class.
Semantic Web

- The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries.
  - Website
    https://www.w3.org/2001/sw/
  - This activity has been subsumed in December 2013, by the W3C Data Activity and “traditional” Semantic Web technologies are now part of activity
Why do we need common framework?

- Useful information is scattered among various websites.

I feel bad. How should I do?

- Simple search using Web

Better to go to Hospital

Which hospital should I go?

- Hospital Web page
  - Address
  - Service time

Hospital A accepts patients.

How can I go to hospital A?

- Transportation
  - Travel navigation

Information to go to hospital A
Why do we need semantic information?

- **Text base information retrieval**
  - Check existence of the keyword
    - without considering the role of the keyword
      - Ramen Sapporo
        » Sapporo Ramen shop at anywhere
        » Ramen shop at Sapporo
        » …
    - Word sense disambiguation
      - Bridge
        » Bridge over the river
        » Bridge (card game)

- **Possibility to integrate results from multiple sites**
  - Address → Nearest station→ Transit information
How to construct Semantic Web

- Necessity to handle structured data
  - Sapporo is address or type of ramen?
  - What kind of categories are necessary?

- Assimilation of different style of writing
  - Synonyms, different language, …
  - Organization of the vocabularies
    - Ontology: Specification of the conceptualization

- Reasoning
  - We need doctor $\rightarrow$ Doctors work at hospital $\rightarrow$ Search hospital
Semantic Web

- From web pages for good user readability to ones for good computer understandability
  - Good user readability
    - Good layout
  - Good computer understandability
    - Precise description about the meaning of the contents appropriately
Semantic Layer Cake (2002 version)

http://www.w3.org/2002/Talks/04-sweb/slide12-0.html
# Technologies for Semantic Web

<table>
<thead>
<tr>
<th>Trust</th>
<th>Framework to evaluate the trustiness of the results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proof</td>
<td>Explain reason and justification of the reasoning results</td>
</tr>
<tr>
<td>Logic</td>
<td>Description based on the first order logic</td>
</tr>
<tr>
<td></td>
<td><em>KIF, N3(?)</em></td>
</tr>
<tr>
<td>Rules</td>
<td>Rules for query resolution</td>
</tr>
<tr>
<td></td>
<td><em>RDQL, N3(?)</em></td>
</tr>
<tr>
<td>Ontology</td>
<td>Precise definition of vocabularies and associations with different vocabularies set for reasoning</td>
</tr>
<tr>
<td></td>
<td><em>OWL, DAML+OIL</em></td>
</tr>
<tr>
<td>RDF Schema</td>
<td>Vocabulary definition (class, property)</td>
</tr>
<tr>
<td></td>
<td><em>RDF Schema</em></td>
</tr>
<tr>
<td>RDF MS</td>
<td>Machine readable metadata representation (data model)</td>
</tr>
<tr>
<td></td>
<td><em>RDF Model &amp; Syntax</em></td>
</tr>
<tr>
<td>XML/Namespace</td>
<td>Markup language for structured data (XML) vocabulary resolution system (namespace)</td>
</tr>
<tr>
<td></td>
<td><em>XML, XML-NS</em></td>
</tr>
<tr>
<td>URI/Unicode</td>
<td>Global resource identifier (URI) and Global data representation (Unicode)</td>
</tr>
<tr>
<td></td>
<td><em>URI, Unicode</em></td>
</tr>
</tbody>
</table>
Modification of Layer Cake

- **Update**
  - by Dr. Tim Berners-Lee (2005)
    http://www.w3.org/2005/Talks/0511-keynote-tbl/
  - by Dr. Steve Bratt (2007)
  - Modification of Rules, SPARQL,…
URI/IRI

- **URI (Uniform Resource Identifier)**
  - Description for identify resources
  - **URL (Uniform Resource Locator)**
    - Internet address based identifier
    - Example: http://www.hokudai.ac.jp/
  - **URN (Uniform Resource Name)**
    - Scheme without using internet address

- **IRI (Internationalized Resource Identifier)**
  - Extension of URI that allows Unicode description

- **Reference**
  - http://www.w3.org/TR/uri-clarification
Unicode

- Unicode is a coding for representing characters used in multiple languages
  - Old encoding scheme cannot handle characters from multiple languages (difficulties to represent Chinese characters and Arabic characters in one text file)
  - UCS-2: representing all characters in 2 bytes
    - In order to reduce the varieties of characters, characters with different shape (that are common in Chinese, Japanese, Korean characters) are assigned in a same code.
  - UCS-4: extension of UCS-2 that can handle more characters in 4 bytes
Encoding scheme

- UTF (Unicode (or UCS) Transformation Format)
  - Encoding method to use UTF characters for storing text data
  - Common encoding
    - UTF-8: Variable length encoding that have good backward compatibility with ASCII code file.
      - ASCII code (1 byte) files can be handled as UTF-8 file.
      - However, most of the UCS characters are represented by 3 bytes.
    - UTF-16: 16 bits (2 bytes) based variable length encoding
      - No backward compatibility with ASCII code file.
      - Most of the UCS-2 characters are represented as 2 bytes.
XML (eXtensible Markup Language)

- Markup Language that can be extensible
  - Recommended by W3C
  - Structured data can be represented in the plain text format (it can be used for data exchange over the internet)

```xml
<?xml version="1.0"?>
<!DOCTYPE school SYSTEM "school.dtd">
<school>
  <student id="1900012">
    <name>Taro Johou</name>
    <email>taro@fakefake.ac.jp</email>
  </student>
  <student id="1900013">
    <name>Hanako Hokudai</name>
    <mobile number="090-xxxx-xxxx"/>
  </student>
</school>
```

- XML documents and DTD
XML in 10 points

1. XML is a method for putting structured data in a text file
2. XML looks a bit like HTML but isn't HTML
3. XML is text, but isn't meant to be read
4. XML is a family of technologies
5. XML is verbose, but that is not a problem
6. XML is new, but not that new
7. 
8. These I don’t know yet.
9. 
10. XML is license-free, platform-independent and well-supported
XML in 10 points (no longer exists)
https://www.w3.org/XML/1999/XML-in-10-points

1. XML is a method for putting structured data in a text file
2. XML looks a bit like HTML but isn't HTML
3. XML is text, but isn't meant to be read
4. XML is a family of technologies
5. XML is verbose, but that is not a problem
6. XML is new, but not that new
7. XML leads HTML to XHTML
8. XML is modular
9. XML is the basis for RDF and the Semantic Web
10. XML is license-free, platform-independent and well-supported
History of XML (Markup language before XML)

- **SGML (Standard Generalized Markup Language)**
  - Markup language to describe varieties of structured documents (e.g., research paper, book, …)
  - **DTD (Document Type Definition)**
    - Definition of structured data scheme using tag set
    - Structured document is organized by using a defined tag set.
    - It is possible to define new DTD by users, but it is difficult to make complete DTD
  - It is not so easy to make appropriate SGML file based on the DTD.

- **HTML (HyperText Markup Language)**
  - Subset of SGML tag + Network extension (hyperlink)
  - Simple tag set: easy to remember and easy to use
  - However, it is not sufficient to describe document structure
The design goals for XML
https://www.w3.org/TR/xml/

- XML shall be straightforwardly usable over the Internet.
- XML shall support a wide variety of applications.
- XML shall be compatible with SGML.
- It shall be easy to write programs which process XML documents.
- The number of optional features in XML is to be kept to the absolute minimum, ideally zero.
- XML documents should be human-legible and reasonably clear.
- The XML design should be prepared quickly.
- The design of XML shall be formal and concise.
- XML documents shall be easy to create.
- Terseness in XML markup is of minimal importance.
Basic Components of XML

- XML Declaration
  - Declaration for using XML

- DTD
  - Definition of the tag set used in the document (not necessary)

- Instance
  - Text with tags
  - The validity of the document is checked by using XML Declaration and DTD (if exists)
    - Evaluation with DTD: Valid
    - Evaluation without DTD: Well Formed
### Example of XML file

- **Structured data description using tags**

```xml
<?xml version="1.0"?>
<!DOCTYPE school SYSTEM "school.dtd">
<division>
  <student id="1900012">
    <name>Taro Johou</name>
    <email>taro@fakefake.ac.jp</email>
  </student>
  <student id="1900013">
    <name>Hanako Hokudai</name>
    <mobile number="090-xxxx-xxxx"/>
  </student>
</division>
```

- **XML Declaration**
- **reference to DTD**
- **division data**
- **element of division (1st student)**
- **element of student**
- **element of student**
- **end of definition for 1st student**
- **definition of 2nd student**
- **end of definition for division**
Data structure definition

- **Valid**: check structured data description refer to DTD
- **Well Formed**: check style of writing

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;!ELEMENT school(student)*&gt;</code></td>
<td>school has multiple student entries</td>
</tr>
<tr>
<td>`&lt;!ELEMENT student (name, (email</td>
<td>mobile))&gt;`</td>
</tr>
<tr>
<td><code>&lt;!ATTLIST employee id CDATA #REQUIRED&gt;</code></td>
<td>employee has id as required attribute and the type is CDATA(character data)</td>
</tr>
<tr>
<td><code>&lt;!ELEMENT name (#PCDATA)&gt;</code></td>
<td>inside the name tag data is described by PCDATA (Parsed Character Data)</td>
</tr>
<tr>
<td><code>&lt;!ELEMENT email (#PCDATA)&gt;</code></td>
<td>same as name</td>
</tr>
<tr>
<td><code>&lt;!ELEMENT mobile EMPTY&gt;</code></td>
<td>url does not have entry</td>
</tr>
<tr>
<td><code>&lt;!ATTLIST mobile number CDATA #REQUIRED&gt;</code></td>
<td>mobile has number as required attribute</td>
</tr>
</tbody>
</table>
Basic Component of XML (Summary)

- User can use multiple DTD with namespace

**XML declaration**

```xml
<?xml version="1.0"?>
<!DOCTYPE school SYSTEM "school.dtd">
<school>
  <student id="1900012">
    <name>Taro Johou</name>
    <email>taro@fakefake.ac.jp</email>
  </student>
  <student id="1900013">
    <name>Hanako Hokudai</name>
    <mobile number="090-xxxx-xxxx"/>
  </student>
</school>
```

**DTD**

```xml
<!ELEMENT school(student)*>
<!ELEMENT student (name, (email | mobile))>
<!ATTLIST employee id CDATA #REQUIRED>
<!ELEMENT name (#PCDATA)>
<!ELEMENT email (#PCDATA)>
<!ELEMENT mobile EMPTY>
<!ATTLIST mobile number CDATA #REQUIRED>
```

**Instance**

```
Syntax: Tag should be closed.

... ... ...
Basic tag (reserved): DOCTYPE, ...
```
Usage of XML

- **Retrieval using document structure**
  - Retrieval system can identify the role of texts by using tag information.
  - E.g., Search students by e-mail address

- **Database system with flexible (or without) schema definition**
  - It is not so easy to modify data storage scheme for relational database.
  - It is easy to add define new instance with new data scheme by using well formed text
  - However, XML itself does not support datatype schema check. (it is necessary to use additional module such as XML Schema)
XML Namespaces provide a simple method for qualifying element and attribute names

- Tag with same name should be same meaning
  - title: book  <title>Introduction to XML</title>
  - title: job  <title>Professor</title>

- One simple method is making \texttt{<bookitle>}, \texttt{<jobitle>} for discrimination, but it is complicated.

- It is enough to have a reference to the DTD that define the tag.
  - \texttt{<book> <title> </title> <author> </author> </book>}
  - \texttt{<employee><name></name><title></title></employee>}

- Use URI for the reference.

Example:

- \texttt{http://www-kb.ist.hokudai.ac.jp/yoshioka:title}
- \texttt{xmlns:yoshioka= \texttt{http://www-kb.ist.hokudai.ac.jp/yoshioka} yoshioka:title}
Integration of XML and Web Application

- Generate HTML file from XML using XSL
  - XSL defines the rule to convert data structure into html texts with specific html style format.
  - The user can separate the management of the main contents and representation issue. (e.g., it is easy to modify the style of representation such as header, banner, footer without changing all html files).

- Data access model of XML
  - DOM(Document Object Model): Data model
  - Simple API for XML(SAX): Event driven API
XPath

http://www.w3.org/TR/xpath

- XML Path language (XPath) is a language to process XML documents.
  - Recent version of XPath is XPath 3.1. https://www.w3.org/TR/xpath-31/

- By using this language, the user can select (a) specific part(s) of XML texts and extract information from them.

- Simple usage example
  - /html/head/title select title in the head part of html.
  - //a anchor text parts in whole documents.
  - //a[@href='http://www.hokudai.ac.jp'] anchor text parts whose href = http://www.hokudai.ac.jp
Summary of XML

- XML is a useful and realistic framework for handling structured data in text format.
- However, it is necessary to have DTD for sharing the data among different sites.
Summary

- Semantic Web
  - Framework to write texts that can be shared among various computers
  - Technologies for Semantic Web
    - Various technologies are integrated as modular and layered structure
- Introduction of technologies used in Semantic Web
  - URL/Unicode
  - XML/Namespace
  - XPath