

Tutorial

Semantic Web: A Road to the Knowledge Infrastructure on the Internet

Hideaki TAKEDA
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan
takeda@nii.ac.jp

Received 4 February 2004

Revised manuscript received 7 May 2004

Abstract In this article, I describe the basic technologies for Semantic Web and relationship between Semantic Web and Knowledge Representation in Artificial Intelligence. Semantic Web is planned as an extension of the current web in order to help cooperation between computers and humans, i.e., computers and humans are expected to understand each other in the knowledge level. I first describe the vision of the Semantic Web, then introduce the current Semantic Web technologies, i.e., RDF, RDFS, and OWL. I describe relationship between the trend of Semantic Web and Knowledge Representation, and clarify challenges and difficulties of Semantic Web from the point of view of Knowledge Representation.

Keywords: Metadata, Knowledge Representation, Knowledge Sharing, Ontology.

§1 Introduction

The World Wide Web (WWW) has been quite widely spread out all over the world so that none can imagine information activities without WWW now. One of the great impact of WWW is that sharing is a primary role for electronically accessible information. As a result, a tremendous number of web pages have become accessible and the number is still growing. This is the first experience that we face such an amount of information.

WWW promotes accessibility of information for sharing. It was a difficult task to achieve before WWW, but now we should consider a step further for sharing, i.e., sharing of meaning or understanding. This is what Semantic Web is aiming. Sharing information is not completed just by acquiring and accumulating digital data for information but by understanding or utilising information in

addition to the former. The difference is already obvious when using the current WWW. Search engines can answer with a plenty of pages to our questions while it is a painful task to select pages really related to the intension of our questions from them.

This is a task for WWW technologies as well as for Artificial Intelligence. Artificial Intelligence has developed theories and technologies to realize knowledge representation and knowledge sharing especially for symbolic information. It is expected that these theories and technologies are applicable to our real world in which a huge amount of symbolic information exists. But no such large-scale databases of symbolic information had existed until WWW has been invented. This situation has brought a difficulty but it was also a good excuse for Artificial Intelligence, because we could not conduct real-scale experiments for theories on Artificial Intelligence. Now WWW offers such a database containing information on almost any aspect of our life. In this context, WWW is an unavoidable theme for researchers in Artificial Intelligence, especially Semantic Web for those in knowledge representation.

In this article, I explain the basic ideas and concepts behind technologies as well as developed or currently developing technologies on Semantic Web. This domain is very young and currently growing so rapidly that technologies can be changeable but knowing the basic concepts will contribute to understanding future technologies and even to creating them.

In this article, I firstly introduce the concept of Semantic Web briefly in Section 2. Then I explain technologies already developed for Semantic Web in Section 3. I overview relationship between technologies for Semantic Web and those for knowledge representation from the research point of view in Section 4. Finally I show some references for further information and conclude the article in Section 5 and 6 respectively.

§2 Vision of Semantic Web

Semantic Web is what Tim Berners-Lee, the inventor of the World Wide Web, also initiated as a future of WWW. He wrote in the well-known article¹⁾ as follows;

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 cooperation.

Another definition is found in pages by Semantic Web Activity in The World Wide Web Consortium (W3C)^{*1};

The Semantic Web is a vision: the idea of having data on the web defined and linked in a way that it can be used by machines not just for display purposes, but for automation, integration and reuse of data across various applications.

*1 <http://www.w3.org/2001/sw/>

Information on WWW is now covering almost every domain or field in our life from telephone directories to daily news. We can now obtain information we need just by clicking links or by typing keywords. On the other hand, we are always frustrated because we have to read many pages before we reach the right information. Furthermore we have to elaborate pages if we wish to integrate information from different sites, e.g., finding clinics which are open on Sunday and within 30 minutes by train. All information is already digitised and published in the specific format then. Why can computers not help us?

The reason is clear, i.e., the current WWW pages are designed just for human, not for computers. The format, i.e., HTML is used for visibility of information. What we should do is to make WWW more processible by computers as well as human. That is the goal of Semantic Web.

There are two points to be noted in the above sentence. One is that information should be understandable *both* for humans and computers, not just for computers. It is a big challenge for computer science because the gap between human and computer understanding has been one of the biggest issues in computer science since it began. The other is that computers are expected to understand information *to some extent* but not whole. The main role to understand information is still left with human and the role of computers is to assist human. It is to ease the challenge because we can extend levels of computer understanding step by step. Languages for Semantic Web, which I will explain in the next section, are different from other programming languages in the above two points.

§3 Technologies for Semantic Web

What are enabling technologies for Semantic Web? The key strategies to develop technologies for Semantic Web are *metadata*, *ontology* and *trust* that are important in technological development.

Metadata means “data about data” that is to describe the content, quality, condition, and other characteristics of data. Information in Semantic Web should hold all meanings for human and have some meanings easy for computers additionally. It can be achieved by letting the original information be as it is and by adding metadata to it. Then computers can *know* something easy to process that is difficult to understand only with the original information.

But adding metadata is just the first step. Metadata provide a method to add descriptions to information. The problem is how such descriptions can be computer-understandable. In Semantic Web, it is to be done by organising information in metadata. Each symbol in metadata should be associated with the others so that computers can *know* symbols more by gathering and processing associated symbols. Such organisation of symbols is called *ontology*. That is the next step of computer understanding on information. Ontology can give computers a chance to process information comprehensively.

The third step is understanding with reliability. Truth values in the real world are not just choices of truth or false like formal logic. Adding trust to WWW information is a way to approach our natural understanding of informa-

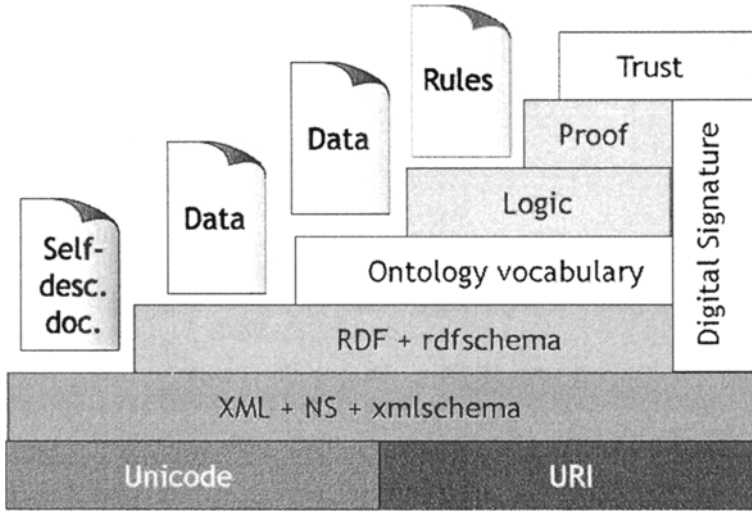


Fig. 1 The Layer Cake for Semantic Web²⁾

tion.

These steps are summarised as so-called the layered cake (see Fig. 1). This figure shows a road-map from the current WWW to future Semantic Web. In this figure, up to the second layer from the bottom is the current status of WWW, i.e., information is represented by XML or HTML. Uniform metadata are realized on the third layer, and metadata is organised as ontology on the fourth layer. The above three layers (Logic, Proof, and Trust) are the next step to reach trustful WWW. Since technologies on the last part are still under discussion, I will mainly explain languages for the third and fourth layers in the rest of this section.

3.1 A Simple Problem

We start with XML to clarify what we want to achieve. XML is not sufficient as a Semantic Web language. Suppose an XML description shown in Fig. 2(a). You can easily understand that it is a description about a person whose name is “Hideaki Takeda” and whose age is “100”. It is not a computer but a human that *knows* what “person”, “name”, and “age” are. This difference may be clear if a description shown in Fig. 2(b) is given to people who do not



Fig. 2 A Simple Problem

understand Chinese characters.*²

We do not want to discuss how much computers should catch meaning that human understand. We need functions to decrease such ambiguity and increase inter-operability among these descriptions. For example,

- Are descriptions Figs. 2(a) and (b) equivalent?,
- Are they sufficient descriptions on “person” ?, or
- Are they valid descriptions on “person” ?

It is not a good idea to expect such functions in XML because XML is intended to provide syntax for descriptions. So we need other languages in which such semantics are described.

3.2 RDF

RDF (Resource Description Framework)**³ is a standard for Web metadata developed by W3C.³ RDF is now a W3C Recommendation. By expanding metadata notion like library catalogs, RDF is suitable to describe metadata for WWW resources. For example, RDF is used to add metadata like title and date to a WWW page. Adding such metadata is similar to what librarians have done to books in libraries. In traditional metadata notion, properties like title and date are fixed, but in RDF there are no restrictions for properties, rather we can describe more complicated metadata like “the email of the creator of this page is abc@de”.

RDF defines a simple data model for metadata, i.e., the data model that consists of three elements;

Resources A resource is anything that can have a URI (Uniform Resource Identifier).⁴⁾**

Properties A property is a specific attribute or relationship that is used to describe a resource. A property of a resource has a value that consists of a description about the resource. A value is either a resource or a literal like a string. A property can be also a resource.

Statement A resource with a property that has a value consists of a statement. In other words, a statement has a subject (a resource), a predicate (a property), and an object (a value). A statement can be also a resource.

Let’s consider an example shown in Fig. 3. This is a statement that means

http://www-kasm.nii.ac.jp/~takeda has a property **creator** of which value is **Hideaki Takeda**.

The graph representation is the formal model for RDF, but we may use a simpler textual representation called *triples* equivalent to the graph model. As a triple, the example is shown like

*² Tags in Fig. 2(b) mean exactly the same to those in Fig. 2(a) in Japanese.

*³ <http://www.w3.org/RDF/>

*⁴ URI includes well-known URL(Uniform Resource Locator) and URN (Uniform Resource Name). A URN differs from a URL in that it’s primary purpose is persistent labelling of a resource with an identifier.

Resource (subject) `http://www-kasm.nii.ac.jp/~takeda`

Property (predicate) `creator`

Value (object) "Hideaki Takeda"

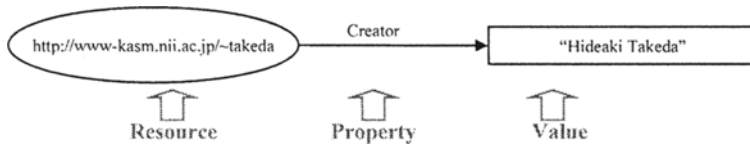


Fig. 3 A RDF Model Example (1)

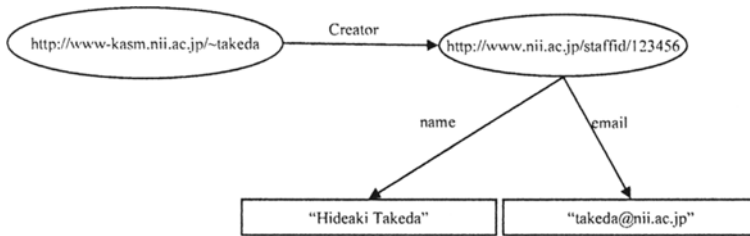


Fig. 4 A RDF Model Example (2)

```
<http://www-kasm.nii.ac.jp/~takeda> creator ``Hideaki Takeda`` .
```

An RDF triple contains three elements of an RDF statement followed by “.”. This statement describes some properties on a WWW page. But a resource is not restricted to a WWW page. It includes

- network-accessible documents and services
- things that are not network-accessible like objects in the real world
- abstract concepts such as the concept of a “creator”

The only restriction is that it should have a URI. Note that the last category means that properties can be also resources (but not necessary). Then, as shown in Fig. 4, we can describe more complicated metadata like

`http://www-kasm.nii.ac.jp/~takeda` has a property **`creator`** of which value is **`http://www.nii.ac.jp/staffid/123456`**, which has a **`Hideaki Takeda`** as **`name`**, and **`takeda@nii.ac.jp`** as **`email`**.

In this case, three statements are used. Note that the node “`http://www.nii.ac.jp/staffid/123456`” does not have to indicate a web page. The function of this URI is just identification of a person. In this example, the actual information on that person is provided by the other statements on “`name`” and “`e-mail`”.

We can even describe statements whose nodes are blank. This notation is useful when we want to describe more structured properties. Figure 5 describes:

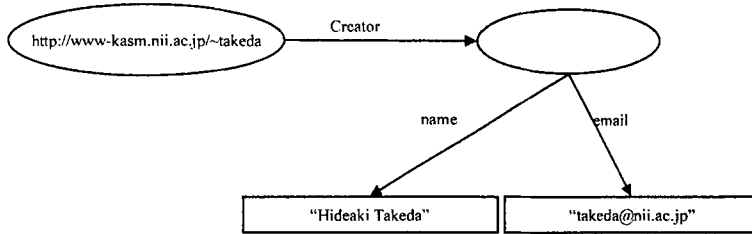


Fig. 5 A RDF Model Example (3)

http://www-kasm.nii.ac.jp/takeda has a **creator** of which value is a resource, which has a name **Hideaki Takeda**, and e-mail **takeda@nii.ac.jp**.

Since RDF allows only binary relations, using blank nodes is one way to describe more complicated relations.

RDF models statements just as nodes and arcs in a graph like Fig. 3. RDF distinguishes modelling statements and representing them as text. The former is a graph representation and the latter is realized by giving some syntax in other languages like XML. We have seen **triples** for textual representation, but they are just intended as a shorthand notion. I introduce XML syntax for RDF in the rest of this section. There are other forms for syntax like N3.⁵⁾

Figure 6 shows the RDF/XML syntax corresponding to Fig. 3. The first line is declaration of XML. The second line indicates the following lines (until `</rdf:RDF>`) are RDF statements. As attribute of `rdf:RDF` tag, namespaces are declared. They mean that tags prefixed with `rdf:` are part of the namespace identified by the URI `http://www.w3.org/1999/02/22-rdf-syntax-ns#` and `dc:` by `http://dublincore.org/2001/08/14/dces#` respectively. The fifth to seventh lines are an actual description of the RDF statement. The `rdf:Description` start-tag with the attribute `rdf:about` is used to specify a resource. Between `rdf:Description` start-tag and end-tag, properties and their values are described. The tag `dc:creator` specifies a property, and its value is described between `dc:creator` start-tag and end-tag.

Since XML syntax allows some alternative ways to describe content, I can describe the same content in a different way shown in Fig. 7. Line 05 – 07 in Fig. 6 is replaced by Line 01 – 03 in Fig. 7. In this case, `dc:creator` is an empty-element tag instead of start- and end-tags, and the value of the property is specified by attribute `rdf:resource` in `dc:creator` tag. It is a little bit

```

01 <?xml version="1.0"?>
02 <rdf:RDF
03   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
04   xmlns:dc="http://dublincore.org/2001/08/14/dces#">
05   <rdf:Description rdf:about="http://www-kasm.nii.ac.jp/~takeda">
06     <dc:creator>Hideaki Takeda</dc:creator>
07   </rdf:Description>
08 </rdf:RDF>

```

Fig. 6 A RDF/XML Syntax Example (1)

```

01 <rdf:Description about="http://www-kasm.nii.ac.jp/~takeda">
02   <dc:creator rdf:resource="Hideaki Takeda" />
03 </rdf:Description>

```

Fig. 7 A RDF/XML Syntax Example (1b)

```

01 <rdf:Description about="http://www-kasm.nii.ac.jp/~takeda">
02   <dc:creator>http://www.nii.ac.jp/staffid/123456</dc:creator>
03 </rdf:Description>
04 <rdf:Description about="http://www.nii.ac.jp/staffid/123456">
05   <p:name>Hideaki Takeda</p:name>
06   <p:email>takeda@nii.ac.jp</p:email>
07 </rdf:Description>

```

Fig. 8 A RDF/XML Syntax Example (2)

```

01 <rdf:Description about="http://www-kasm.nii.ac.jp/~takeda">
02   <dc:creator>
03     <rdf:Description resource = "http://www.nii.ac.jp/staffid/123456">
04       <p:name>Hideaki Takeda</p:name>
05       <p:email>takeda@nii.ac.jp</p:email>
06     </rdf:Description>
07   </dc:creator>
08 </rdf:Description>

```

Fig. 9 A RDF/XML Syntax Example (2b)

```

01 <rdf:Description about="http://www-kasm.nii.ac.jp/~takeda">
02   <dc:creator>
03     <rdf:Description>
04       <p:name>Hideaki Takeda</p:name>
05       <p:email>takeda@nii.ac.jp</p:email>
06     </rdf:Description>
07   </dc:creator>
08 </rdf:Description>

```

Fig. 10 A RDF/XML Syntax Example (3)

shorter than the previous one.

In a similar fashion, Fig. 4 is represented as Fig. 8.^{*5} In this case, there are two `rdf:Description` tags at the top level, and the second one (Line 04 – 07) contains two elements each of which describes a property and its value.

By unifying the node `http://www.nii.ac.jp/staffid/123456` in the two `rdf:Description` elements, we can obtain more compact one shown in Fig. 9. In this case, two `rdf:Description` elements are nested. The nested descriptions are not usually easy to read, but there is a benefit to use this style, i.e., we can describe blank nodes in this fashion. Fig. 10 is the XML syntax for Fig. 5. The only difference is the inner `rdf:Description` tag does not have `rdf:about` attribute (Line 03). It is permitted because the nest structure can assure the `rdf:Description` element uniquely.

3.3 RDF Schema

RDF Schema (or RDFS, for short) provides additional modelling primi-

^{*5} Hereafter, I only show RDF statements inside `rdf:RDF` tags just for space.

tives on top of RDF, i.e., a simple model of classes and their relations can be defined with RDFS. Although RDF provides a basic level of functions to describe simple metadata like bibliography, it is difficult to represent metadata with a more complicated structure. We can use RDFS to define a structure of classes then it acts as a vocabulary of RDF statements in turn. RDFS is also a W3C Recommendation.⁶⁾

Typical characteristics of “class” are as follows: A class is a sub-class of some other class and may have some sub-classes of its own. A class may have some properties to be filled with some values. Most of these characteristics are realized in RDFS except relationship between classes and properties. I will explain it in Section 3.4 as the difference between RDFS and OWL.

The first step is to declare something as a class. In the following examples, we assume a namespace denoted `ex:` is declared in advance. For example, we can declare that “Publication” is a class in the following way:

```
ex:Publication rdf:type rdfs:Class .
```

`rdf:type` is a RDF property to indicate that a resource is an instance of a class. Then we can define a subclass of a class like:

```
ex:Book rdf:type rdfs:Class .  
ex:Book rdfs:subClassOf ex:Publication .
```

You can describe hierarchy of classes in this way. In the same way, we can define a sub-property of a property with `rdfs:subPropertyOf`. An instance of a class is described in the following way:

```
ex:theTeXbook rdf:type ex:Book .
```

Typical classes have properties or attributes, e.g., “a book has an author of which value is a person” is described as follows:

```
ex:Person rdf:type rdfs:Class .  
ex:author rdf:type rdf:Property .  
ex:author rdfs:range ex:Person .  
ex:author rdfs:domain ex:Book .
```

The first line defines Class `Person`, then the following lines define Property `author`. The second line indicates that `author` is a property. The third line describes that its value is to be an instances of Class `Person`, and the last line does that it is a property for Class `Book`, i.e., predicate `author` only allows `Book` as subject and `Person` as object in SVO (Subject-Verb-Object) model.

As we have seen in the previous sub-section, RDFS statements are also described with RDF/XML syntax. A class declaration is represented in Fig. 11. RDF/XML provides an abbreviation for describing resources having an `rdf:type` property. Then the above example is also described as shown in Fig. 12. By using this abbreviation, descriptions look like those in object-oriented programs.

The examples I have explained throughout this sub-section are shown in

```

01 <rdf:Description rdf:ID="Publication">
02   <rdf:type rdf:resource="rdfs:Class"/>
03 </rdf:Description>

```

Fig. 11 A RDF/XML Syntax Example (4)

```

01 <rdfs:Class rdf:ID="Publication"/>

```

Fig. 12 A RDF/XML Syntax Example (4b)

```

01 <?xml version="1.0"?>
02 <rdf:RDF
03   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
04   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
05
06 <rdfs:Class rdf:ID="Person"/>
07
08 <rdfs:Class rdf:ID="Publication"/>
09
10 <rdfs:Class rdf:ID="Book">
11   <rdfs:subClassOf rdf:resource="#Publication"/>
12 </rdfs:Class>
13
14 <rdf:Property rdf:ID="author">
15   <rdfs:domain rdf:resource="#Book"/>
16   <rdfs:range rdf:resource="#Person"/>
17 </rdf:Property>
18
19 <Person rdf:ID="donaldEKnuth"/>
20
21 <Book rdf:ID="theTeXBook">
22   <author rdf:resource="#donaldEKnuth">
23 </Book>
24
25 </rdf:RDF>

```

Fig. 13 A RDF/XML Syntax Example (5)

Fig. 13. Note that the declaration of an instance `Person` and `Book` is also an abbreviation of `rdf:type` property.

There are two points for RDFS to be noted from knowledge representation view.

If we regard RDFS as an object-oriented language or frame language, there is a unique or rather curious characteristics on a property, i.e., the scope of a property is not bound to a class but global. In the previous example, I intended to declare `author` as a property of Class `Book`, but in a precise sense, it is not true. Property `author` is globally defined, but its subject is restricted to `Book`. That is why I declared Property `rdfs:domain` for `author`. It is the significant difference between RDFS and other object-oriented or frame-based systems. The difference is apparent if you want to declare Class `Software` with Property `author`. Since Property `author` is already used, you extend the domain and ranges of Property `author`⁶ or use a different property instead.

The other is a more serious issue as knowledge representation language. As many programming languages do, RDFS looks like a language that restricts

⁶ `rdf:range` and `rdf:domain` can be used twice or more for a property.

representation for information or knowledge and therefore gives us benefits like unambiguous and shared interpretation among applications. RDFS is not similar to other computer languages in this sense. RDFS provides syntax for object-oriented style, but it does not force a unique interpretation to applications, rather they are left on their own. For example, even if `Book` is declared as Fig. 13, you can add the other property like `technical-editor` without changing descriptions in Fig. 13. This loose restriction is reasonable as metadata description because descriptions should be open in order to be shared with a wide variety of applications. But it is insufficient as a programming language.

3.4 OWL

As we have seen in the previous sub-section, RDFS provides some expressive functions for an object-oriented or frame-like structure, but they are not sufficient in many points. OWL (Web Ontology Language)⁷⁾ adds more vocabulary for describing properties and classes: relations between classes (e.g. disjointness), cardinality (e.g. “exactly one”), equality, richer typing of properties, characteristics of properties (e.g. symmetry), and enumerated classes. OWL is mainly developed by Web-Ontology Working Group in W3C and it is now a W3C Recommendation.

Historically OWL is a direct successor of DAML+OIL^{8,9)} that is a merger between DAML-ONT developed as part of the US DARPA Agent Markup Language (DAML) project⁷⁾ and OIL¹⁰⁾⁸⁾ developed mainly by European researchers. The theoretical background of OWL is Description Logics. Description Logics are knowledge representation formalism based on logic developed for frame-based systems, semantic network, KL-ONE-like languages, and object-oriented languages. I do not go into detail on Description Logics and please refer other articles (e.g., Reference¹¹⁾).

Because OWL has a plenty of constructs, I just pick up some of them to illustrate features of OWL in comparison to RDFS. Please consult OWL¹²⁾ to know full descriptions of OWL.

Firstly, I introduce how to define a class in OWL. Class definition is composed of class descriptions. Composition is done with construct `rdfs:subClassOf`, `owl:equivalentClass`, and `owl:disjointWith`. The first construct means that the extension of defining class is a subset of the extension of another class. The second means that they are equal, and the third that they are disjoint. The first corresponds to a necessary condition and the second a necessary and sufficient condition.

A class description is either

1. a class identifier,
2. an exhaustive enumeration of individuals,
3. a property restriction,
4. the intersection of class descriptions,
5. the union of class descriptions, or

⁷⁾ <http://www.daml.org/>

⁸⁾ <http://www.ontoknowledge.org/oil/>

```

01 <owl:Class rdf:ID="Person">
02   <rdfs:subClassOf rdf:resource="#Animal"/>
03   <rdfs:subClassOf>
04     <owl:Restriction>
05       <owl:onProperty>
06         <owl:DatatypeProperty rdf:about="#name"/>
07       </owl:onProperty>
08       <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">
09         1
10       </owl:minCardinality>
11     </owl:Restriction>
12   </rdfs:subClassOf>
13 <rdfs:subClassOf>
14   <owl:Restriction>
15     <owl:onProperty>
16       <owl:ObjectProperty rdf:about="#eat"/>
17     </owl:onProperty>
18     <owl:allValuesFrom rdf:resource="#LivingThing" />
19   </owl:Restriction>
20 </rdfs:subClassOf>
21 </owl:Class>

```

Fig. 14 An OWL Example (1)

6. the complement of a class description.

The interesting feature is the third kind of description, which enables to describe a property with a constraint that is local to a class. There are two types of property restriction, i.e., value restriction and cardinality constraint. The former uses `owl:allValuesFrom`, `owl:someValuesFrom`, and `owl:hasValue`, which restrict the value of a property to a specific range in any case, in some case, and a specific value, respectively. The latter uses `owl:minCardinality`, `owl:maxCardinality`, and `owl:cardinality`, which specify the number of values at most, at least, and exactly, respectively. All property restrictions are described within element `owl:Restriction` specifying a property by `rdf:resource` attribute.

Figure 14 shows an example to define a class. Class `Person` is defined as a subclass of three classes. It is a kind of multiple inheritance, i.e., the class should be a subclass of three classes simultaneously. One of the three classes is Class `Animal` and the others are un-named classes that have only property restrictions. The first un-named class is a class that has at least a name property and the second is a class with the following restriction on its Property `eat`. The value of the property should be an instance of Class `LivingThing`. As a result, Class `Person` is defined as a a kind of `Animal`, which has at least a name and can eat only living things. Note that the range restriction to Class `LivingThing` for Property `eat` is not global but just for this new class. Property `eat` can be used in other places independently from this restriction.

A property is defined independently with a class as RDFS, but now we have two types of property, i.e., `DatatypeProperty` and `ObjectProperty`. The former is a relation from an instance of a class to a RDF literal or an XML Schema type, and the latter to another instance of a class. Just as RDFS, a property has constraints like `rdfs:range`, `rdfs:domain`, and `rdfs:subPropertyOf`. In addition, OWL provides constructs to specify char-

```

01 <owl:ObjectProperty rdf:ID="eat">
02   <rdfs:domain rdf:resource="#LivingThing"/>
03   <rdfs:range rdf:resource="#owl:Thing"/>
04 </owl:ObjectProperty>
05 <owl:DatatypeProperty rdf:ID="age">
06   <rdf:type rdf:resource="#owl:FunctionalProperty" />
07   <rdfs:domain rdf:resource="#Animal" />
08   <rdfs:range rdf:resource="#xsd:positiveInteger" />
09 </owl:DatatypeProperty>

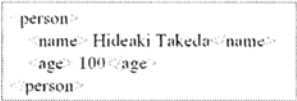
```

Fig. 15 An OWL Example (2)

acteristics of relationship like transitivity and symmetry. `FunctionalProperty` is another characteristics of relationship that indicates the value of the property should be unique.

Figure 15 shows two examples of property definitions. The first property `eat` is an `ObjectProperty` because the its range is Class `Thing`. The second is a `DatatypeProperty` of which range is an integer. In addition, if some class has the property, the value should be unique.

We can now define an instance of Class `Person` shown in Fig. 16. From the class definition, it should hold a name property with a value and values of `eat` should be instances of `LivingThing` class. From the definition of Property `age`, its value should be a positive integer and functional. An instance defined in Fig. 16 is a valid because it satisfies the above requirement.



```

01 <Person rdf:ID="Hideaki">
02   <rdfs:label>Hideaki</rdfs:label>
03   <rdfs:comment>
04     Hideaki is a person.
05     His name is Hideaki Takeda.
06     His age is 100.
07   </rdfs:comment>
08   <name>Hideaki Takeda</name>
09   <age rdf:datatype="#xsd:positiveInteger">100</age>
10 </Person>

```

Fig. 16 An OWL Example (3)

Now we go back to the simple problem shown at the beginning of this section (Fig. 2). The description in Fig. 16 is almost identical to that in Fig. 2. The difference lies in definitions behind it. As I mentioned above, we can check satisfiability of the description as an instance of Class `Person`, e.g., whether necessary properties exist or not. Furthermore I can infer more from this description. For example, since Class `Person` is a subclass of Class `Animal`, any property whose range is `Animal` is also applicable to this instance. Suppose that a Semantic Web Agent is looking for doctors nearby for your emergent need. If it cannot find appropriate doctors, it may recommend an animal doctor instead by inferring this subclass relationship.*"

There are three sub-languages in OWL, i.e., OWL Lite, OWL DL, and OWL Full. What I explained is based on OWL DL. The difference between OWL (Lite and DL) and RDFS is that the former restricts descriptions with

*" Of course, it depends on you whether you accept this recommendation or not. :-)

only the defined constructs. It is more like other computer languages. Thanks to this feature, OWL can offer syntactical checking like OWL Validator and reasoning services, which are actually reasoning with Description Logics. The separation between OWL Lite and OWL DL is a practical reason. OWL DL offers a full set of expression in Description Logics. On the other hand, we should bear computational complexity to realize reasoning systems. OWL Lite is a subset of OWL DL that is suitable to describe classification hierarchies and simple constraints. For example, cardinality is only 0 or 1, union of classes is only allowed as class composition, and so on. Its benefit is that tools to support OWL Lite are simpler than OWL DL. OWL Full allows standard RDF statements with OWL DL descriptions. Since it has maximum expressiveness but serious problems in formal semantics, it is not a practical solution at this moment.

§4 Semantic Web and Knowledge Representation

From AI perspective, Semantic Web languages mentioned in the previous section do not look new, rather may look like “reinventing the wheel”. You may conclude that there is nothing for AI researchers to investigate in Semantic Web and in its languages. I think that it is a little bit nearsighted conclusion which restricts our scope to the problems we used to treat. Although languages themselves seem the same, the environment is totally different. It gives new challenges for us.

4.1 Openness of Knowledge

The first aspect is *openness* of knowledge. Knowledge in WWW is *intrinsically* open. It does not mean simply that descriptions are incomplete, which itself is difficult for AI, but that meaning of descriptions are dependent on our real life that can not be totally described in WWW. Since WWW has penetrated our daily life so much, huge and various information in our life is already represented on WWW. However, it is impossible to suppose that every piece of information in our life is represented on WWW. Information in our life is still partially on WWW and partially just in our real life, although the balance of the two parts are rapidly being changed. Totality of information is never achieved only on WWW. In other words, information on WWW should be eventually grounded on our real life (see Fig. 17).

In the field of Knowledge Representation Research, under a great influence of formal logic, knowledge *should* be consistent and complete. With such a premise, data and knowledge are not substantially different, because knowledge always satisfies data and vice versa. Since WWW is an open world as mentioned above, we cannot expect the complete data or knowledge, rather, data and knowledge are always incomplete. As WWW is a truly distributed world, we cannot expect consistent data and knowledge, rather, data and knowledge are *always* inconsistent. Then difference between data and knowledge is substantial for inference. Semantic Web languages are aiming knowledge representation with this condition.

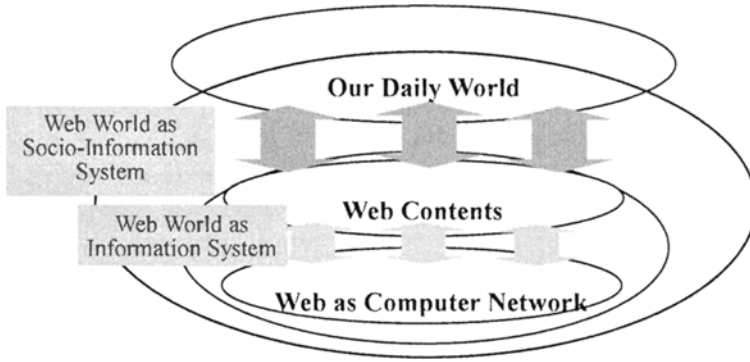


Fig. 17 The Three Layers for WWW

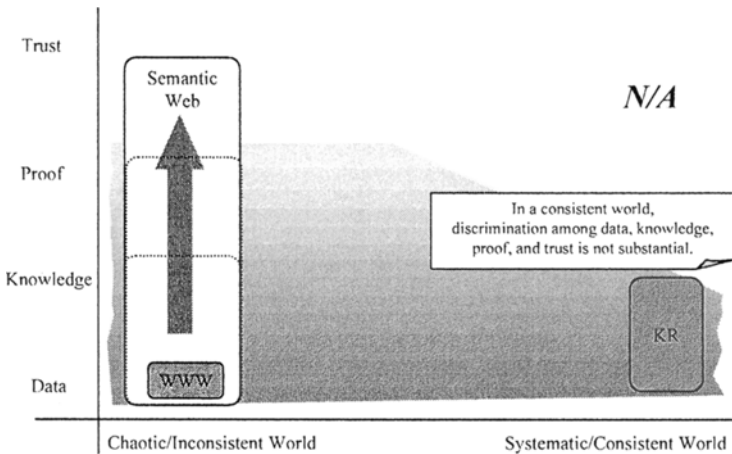


Fig. 18 KR and Semantic Web

The key issue to deal with inconsistency and incompleteness of knowledge and data is *trust*. Resolving inconsistency among data or knowledge, and compensating incomplete data or knowledge depends on which data or knowledge is more reliable than others. It is not simply a degree of trust, but it is more complicated, i.e., trust is context-dependent.

A feasible direction is involvement of human affairs like human networks of acquaintance. Since trust is matured with human network, trust can be modelled by analysing human network and communities. An increasing number of researchers are now interested in knowledge with human networks, e.g., References.^{13~15)}

Thus Semantic Web expands fields of Knowledge Representation from data to trust (see Fig. 18). The vertical axis corresponds to layers in the Layer Cake for Semantic Web. Data and knowledge with trust is a new challenge for Knowledge Representation.

4.2 Human-in-the-Loop in Knowledge Creation

The other aspect concerns knowledge creation process. Constructing a very large knowledge base has been a challenge for knowledge engineering. Semantic Web can be considered as a world-wide social experiment for knowledge creation. Semantic Web provides Knowledge Representation languages for public and expects that people in various domains would construct and distribute knowledge in their domains.

In the past years, there exist knowledge bases *in-house* or made in laboratories like Cyc¹⁶⁾ and EDR.¹⁷⁾ They formed a group of specialists and asked them to construct knowledge bases. It is obvious that tremendous efforts are required to construct a large knowledge base like a common-sense knowledge base. It is therefore not surprising that these knowledge bases are partially useful but still need to be revised or expanded even further.

In contrast, everyone can participate to construct knowledge bases in Semantic Web. We hope that by far larger knowledge bases would be created by people than such in-house knowledge bases. On the other hand, people may worry about the quality of knowledge bases. Anyway it is very interesting how different knowledge bases would be constructed with different processes.

In addition, it is really a collaborative process among people. It is partially a social process and partially an engineering process. We need a new model for such socio-technical developing processes.

There are some promising ongoing projects in which common-sense knowledge has been developed by a large number of non-expert people. A good example is the Open Directory Project^{*10} which provides a large hierarchy of web directory. There are currently 0.6 million categories and 4.6 million sites in the directory edited by 60,000 people. The directory lacks definitions but can be regarded as a kind of light-weight ontologies.^{*11} The other example is Wikipedia^{*12} which is an encyclopedia edited by a lot of people. There are some others like OpenMind Project^{*13} and MusicBrainz.^{*14} The quality of knowledge in these projects is maintained just by peer review. But as long as I know, their quality has been maintained well. These projects show possibility of knowledge creation in this way. I expect that ontologies for Semantic Web will be built and maintained in a similar but more sophisticated fashion supported by Artificial Intelligence techniques.

§5 For Further Information

I skipped descriptions of some features in the languages intentionally to simplify explanation. The best way to know languages explained in this article in detail is to read specification documents. I recommend RDF Primer¹⁸⁾ to understand core concepts of RDF and RDF Schema, and OWL Guide¹⁹⁾ for OWL. There are some other good articles to know the outline of Semantic Web

^{*10} <http://dmoz.org>

^{*11} The directory is available not only as HTML pages but also as data by RDF format.

^{*12} http://en.wikipedia.org/wiki/Main_Page

^{*13} <http://www.openmind.org/>

^{*14} <http://www.musicbrainz.org>

and its languages.^{20,21)} Since specifications of languages have been changed, you should refer to the language specifications to know their current status in detail.

There are several research projects strongly related to Semantic Web. DAML (The DARPA Agent Markup Language) is a US Government-funded project in which languages like DAML^{*15} and their applications have been intensively investigated, and OnToKnowledge,^{*16} OntoWeb,^{*17} WonderWeb^{*18} and SWAD-Europe^{*19} are European projects in which research on OIL and other ontology-related themes has been conducted.

A conference series called International Semantic Web Conference (ISWC)^{*20} has been held since 2002. The proceedings of the first and second ISWC conferences are published as books in Springer LNCS series.^{22,23)}

§6 Conclusion

In this article, I described the aims, technologies, and challenges of Semantic Web briefly. It is just a starting point, because technologies for Semantic Web are still under development and I cannot foresee their final form.

From AI perspective, Semantic Web can be a gateway between the research field and the real world. Through Semantic Web, not only Knowledge Representation techniques but also other AI techniques like machine learning and agent technologies would be applied to the real world. In other words, AI can potentially contribute much more to Semantic Web. I hope that many researchers and engineers would be interested in and contribute to Semantic Web.

References

- 1) Berners-Lee, T., Hendler, J. and Lassila, O., "The Semantic Web," *Scientific American*, May 2001.
- 2) Berners-Lee, T., "Semantic Web." Keynote Speech, *XML 2000*, 2000, <http://www.w3.org/2000/Talks/1206-xml2k-tbl/>
- 3) Lassila, O., "Web Metadata: A Matter of Semantics," *IEEE Internet Computing*, 2, 4, pp. 30–37, 1998.
- 4) Berners-Lee, T., Fielding, R. and Masinter, L., "RFC 2396 - Uniform Resource Identifiers (URI): Generic Syntax," *tech. rep., IETF*, 1998.
- 5) Berners-Lee, T. and Connolly, D., "Primer: Getting into RDF & Semantic Web using N3," <http://www.w3.org/2000/10/swap/Primer.html>
- 6) Brickley, D. and Guha, R., "RDF Vocabulary Description Language 1.0: RDF Schema," *W3C Recommendation* 10 Feb. 2004, <http://www.w3.org/TR/rdf-schema/>
- 7) McGuinness, D. L. and Harmelen, F. van, "OWL Web Ontology Language Overview," *W3C Recommendation* 10 Feb. 2004, <http://www.w3.org/TR/owl-features/>

*15 <http://www.daml.org/>

*16 <http://www.ontoknowledge.org/>

*17 <http://ontoweb.aifb.uni-karlsruhe.de/>

*18 <http://wonderweb.semanticweb.org/>

*19 <http://www.w3.org/2001/sw/Europe/>

*20 <http://iswc.semanticweb.org/>

- 8) Connolly, D., Harmelen, F. van Horrocks, I. McGuinness, D. L., Patel-Schneider, P. F. and Stein, L. A., "DAML+OIL (March 2001) Reference Description," *W3C Note* 18 Dec. 2001, <http://www.w3.org/TR/daml+oil-reference>
- 9) Horrocks, I., "DAML+OIL: a description logic for the semantic web," *IEEE Data Engineering Bulletin*, 25, 1, pp. 4–9, 2002.
- 10) Fensel, D., Harmelen, F. van, Horrocks, I., McGuinness, D. and Patel-Schneider, P., "Oil: An Ontology Infrastructure for the Semantic Web," *Intelligent Systems*, 16, 2, pp. 38–44, 2001.
- 11) Baader, F., Calvanese, D., McGuinness, D. Nardi, D. and Patel-Schneider, P. (eds.), *The Description Logic Handbook: Theory, Implementation and Applications*, Cambridge University Press, 2003.
- 12) Dean, M. and Schreiber, G., "OWL Web Ontology Language Reference," *W3C Recommendation* 10 Feb. 2004, <http://www.w3.org/TR/owl-ref/>
- 13) Golbeck, J., Parsia, B. and Hendler, J., "Trust networks on the semantic web," in *Proc. of Cooperative Intelligent Agents 2003*, Helsinki, Finland, Aug. 2003.
- 14) Adamic, L. A., Buyukkotken, O. and Adar, E., "A Social Network Caught in the Web," *First Monday*, 8, 6, 2003.
- 15) Ohmukai, I., Numa, K. and Takeda, H., "Egocentric Search Method for Authoring Support in Semantic Weblog," *Workshop on Knowledge Markup and Semantic Annotation (Semannot2003)*, Held in conjunction with the Second Intern. Conf. on Knowledge Capture (K-CAP2003), 2003.
- 16) Lenat, D. B. and Guha, R. V., *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project*, Addison-Wesley Longman Publishing Co., Inc., 1989.
- 17) Yokoi, T., "The EDR Electronic Dictionary," *Communications of ACM*, 38, 11, pp. 42–44, 1995.
- 18) Manola, F. and Miller, E., "RDF Primer," *W3C Recommendation*, 10 Feb. 2004, <http://www.w3.org/TR/rdf-primer/>
- 19) Smith, M. K., Weltyand, C. and McGuinness, D. L., "OWL Web Ontology Language Guide," *W3C Recommendation*, 10 Feb. 2004, <http://www.w3.org/TR/owl-guide/>
- 20) Klein, M. C. A., Broekstra, J., Fensel, D. Hermelen, F. van and Horrocks, I., "Ontologies and Schema Languages on the Web," in *Spinning the Semantic Web* (D. Fensel, J. A. Hendler, H. Lieberman, and W. Wahlster, eds.), pp. 95–139, MIT Press, 2003.
- 21) Broekstra, J., Klein, M. C. A., Decker, S., Fensel, D., Harmelen, F. van and Horrocks, I., "Enabling Knowledge Representation on the Web by Extending RDF Schema," *Computer Networks*, 39, 5, pp. 609–634, 2002.
- 22) Horrocks, I. and Hendler, J. (eds.), "The Semantic Web - ISWC 2002," of *Lecture Notes in Computer Science*, 2342, Springer, 2002.
- 23) Fensel, D., Sycara, K. and Mylopoulos, J. (eds.), "The Semantic Web - ISWC 2003," of *Lecture Notes in Computer Science*, 2870, Springer, 2003.



Hideaki Takeda: He is a professor at National Institute of Informatics (NII) and a professor in Department of Informatics at the Graduate University of Advanced Studies (Sokendai). He received his Ph.D. from the University of Tokyo in 1991. His research interest in computer science includes ontology engineering, community informatics and knowledge sharing systems.