

Semantic Web

Web as a communication infrastructure between human and machines

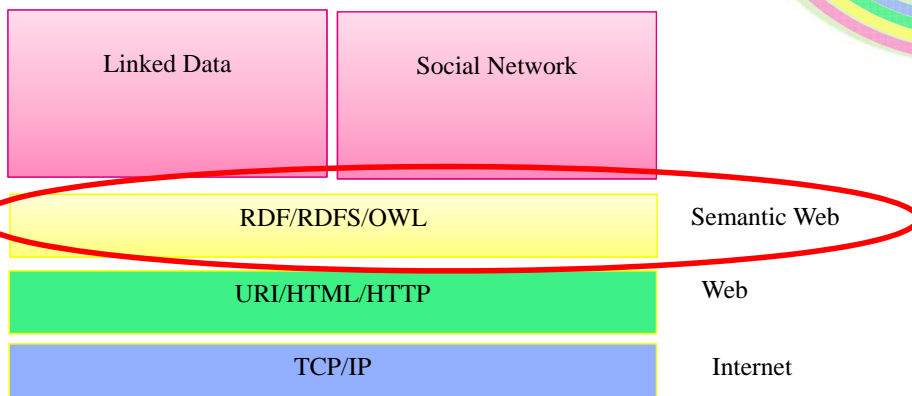
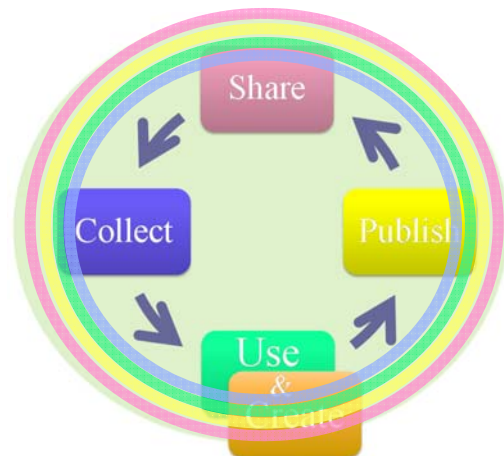
Hideaki Takeda
National Institute of Informatics
takeda@nii.ac.jp

Hideaki Takeda / National Institute of Informatics



Layers for Information Cycle

- Internet
- Web
- Semantic Web
- Linked Data
- Social Network



Hideaki Takeda / National Institute of Informatics



Outline

- What is Semantic Web?
 - XML
- Realization of Semantic Web
 - Metadata
 - Dublin Core
 - RDF
 - RDF Schema
 - OWL

The Aim of The Semantic Web

- "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."

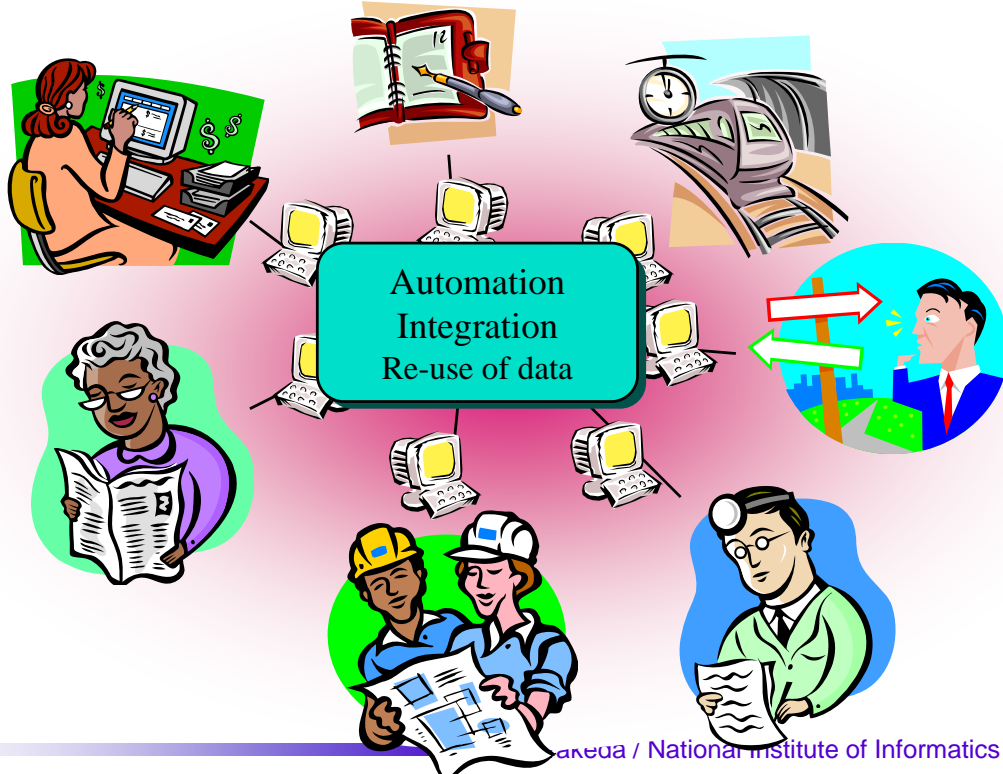
[The Semantic Web](#), Scientific American, May 2001, Tim Berners-Lee, James Hendler and Ora Lassila

- 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**.

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

Semantic Web

- Realization of various information exchanging via Web



Next Generation Web?

- Evolution of Web
 - HTML: Web for Display
 - XML: Web with Syntax
 - ?? : Web with **Semantics**
 - Why should we embed semantics into Web?
 - From
 - Web for Human
 - To
 - **Web for human and machines**
- cf. Web for machines

A brief introduction of XML

- Limitation of HTML
 - Chaos by mixture of displaying and text structures
 - ◆ e.g.,
 - `<h3></h3>` should be used for “the third-level heading”, but are often used just for bigger fonts
 - `` is specifying “bold” , not “emphasis”.
 - Fixed Structure
 - ◆ e.g.,
 - If you need `<h7></h7>....`
 - I need a structure just for my data

```
<h1> A list of lectures</h1>
<h2> Knowledge Sharing Systems</h2>
<h3> Lecturer: Hideaki Takeda</h3>
<h3>Wednesday 3rd</h3>
```

XML

- XML(eXtensible Markup Language)
 - Can define original tags
 - Represent logical structures of data
 - ◆ DTD
 - Do not include style information
 - ◆ XST

```
<lecturelist>
<lecture>
<title id=1234> Knowledge Sharing Systems</title>
<lecturer> Hideaki Takeda</lecturer>
<schedule>
  <week>Wednesday</week>
  <time>3rd</time>
</lecture>
...
</lecturelist>
```

XML

- XML(eXtensible Markup Language)
 - A standard by W3C(The World Wide Web Consortium) (1998)
 - A subset of SGML
 - Features
 - ◆ Fixed character set
 - ◆ DTD is not mandatory
 - ◆ Should not omit end-tags
 - ◆ Style can be specified
- SGML(Standard Generalized Markup Language)
 - ISO8879 (1986)
 - Features
 - ◆ Can specify character set
 - ◆ DTD is mandatory
 - ◆ Can omit end-tags
 - ◆ Style can be specified

DTD

- DTD(Document Type Definition):
 - Specify elements and attributes for XML logical structures
 - ◆ Defining a specific markup language
 - XHTML
 - ◆ Logical structure can be specified
 - Should share DTD between document authors and users

```
<!DOCTYPE lecturelist [  
<!ELEMENT lecturelist (lecture*)>  
<!ELEMENT lecture (title, lecturer*, schedule)>  
<!ELEMENT title (#PCDATA)>  
<!ATTLIST title id ID #REQUIRED >  
<!ELEMENT lecturer (#PCDATA)>  
<!ELEMENT schedule (week, time)>  
<!ELEMENT week (#PCDATA)>  
<!ELEMENT time (#PCDATA)>  
>
```

DTD

- Element type declaration
 - <!ELEMENT element-name (element-content) >
 - Element name
 - Element content
 - ◆ Data type
 - ◆ Children
 - ◆ Order
 - ◆ Number of appearance: *(0-), +(1-),?(0-1)
- Attribute-list declaration
 - <!ATTLIST element-name attribute-name attribute-type default-value>
- Entity declaration
 - <!ENTITY entity-name "string to be replaced">
- Notation declaration

Style Sheet

- XSLT (eXtensible Stylesheet Language Transformations)
 - A language to describe rules to translate structures in XML
 - Originally a part of XML (eXtensible Stylesheet Language) which is a language to display or print XML data
 - Procedure
 - ◆ A template for each node in the source tree
 - ◆ Specify which node should be applied to a template with *match* attribute of *xsl:template* element
 - ◆ XSLT processor analyzes from the top-most node to down
 - ◆ Find a pattern of appearance of elements and attributes which is specified with *match* attribute of *xsl:template* element
 - ◆ Apply a rule of a template applied

```

<?xml version="1.0" encoding="shift_jis"?><xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html" encoding="shift_jis"/>
  <xsl:template match="/">
    <html><body>
      <xsl:apply-templates/>
    </body></html>
  </xsl:template>
  <xsl:template match="lecturelist">
    <table border="1">
      <tr><td>Lecture name</td><td>Lecturer</td><td>Time</td></tr>
      <xsl:apply-templates /></table>
    </xsl:template>
  <xsl:template match="lecture">
    <tr><xsl:apply-templates/></tr>
  </xsl:template>
  <xsl:template match="title">
    <td><xsl:value-of select="." /></td>
  </xsl:template>
  <xsl:template match="lecturer">
    <td><xsl:value-of select="." /></td>
  </xsl:template>
  <xsl:template match="schedule">
    <td> <xsl:apply-templates /> </td>
  </xsl:template>
  <xsl:template match="week">
    <xsl:value-of select="." />
  </xsl:template>
  <xsl:template match="time">
    <xsl:value-of select="." />
  </xsl:template> </xsl:stylesheet>

```

```

<?xml version="1.0" encoding="shift_jis"?>
<?xml-stylesheet href="lecturelist.xml" type="text/xsl" ?>
<!DOCTYPE lecturelist [
<!ELEMENT lecturelist (lecture*)>
<!ELEMENT lecture (title, lecturer*, schedule)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT lecturer (#PCDATA)>
<!ELEMENT schedule (week, time)>
<!ELEMENT week (#PCDATA)>
<!ELEMENT time (#PCDATA)>
]>
<lecturelist>
<lecture>
<title id=1234> Knowledge Sharing
Systems</title>
<lecturer> Hideaki Takeda</lecturer>
<schedule>
  <week>Wednesday</week>
  <time>3rd</time>
</lecture>
</lecturelist>

```



```

<html>
<body>
<table border="1">
<tr>
<td>Lecture name</td>
<td>Lecturer</td>
<td>Time</td>
</tr>
<tr>
<td> Knowledge Sharing Systems </td>
<td> Hideaki Takeda</td>
<td>
  Wednesday/3rd
</td>
</tr>
</table>
</body>
</html>

```

Why is XML not sufficient?

```
<person>
  <name> Hideaki Takeda</name>
  <age> 20</age>
</person>
```

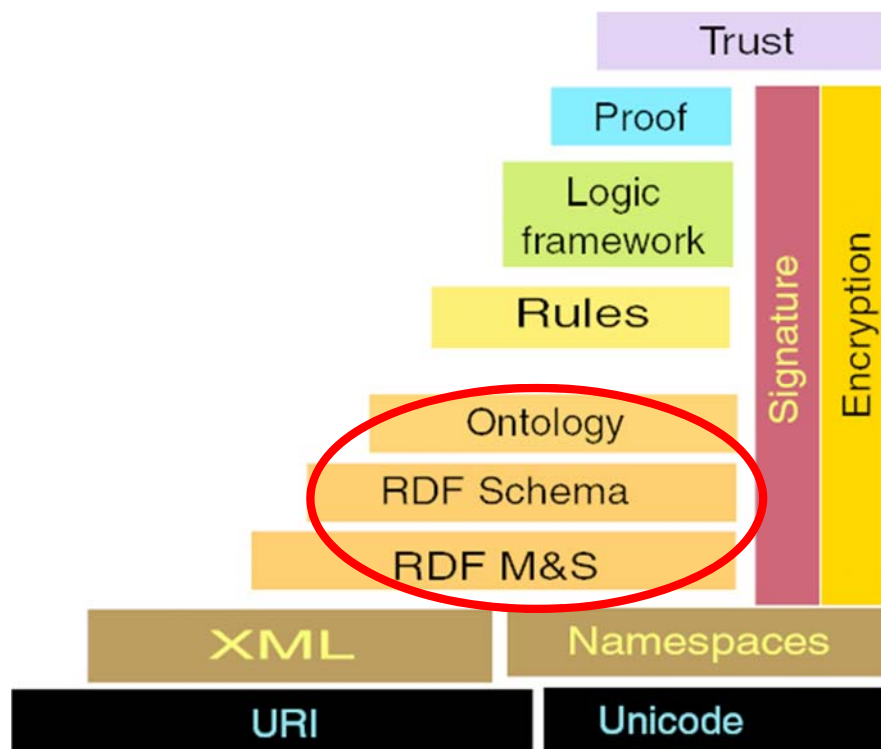
```
<個人>
  <名前>Hideaki Takeda</名前>
  <年齢> 20</年齢>
</個人>
```

- What are specified by “person” and “name” ?
- Is “name” and “名前” the same?
- Is this description sufficient as a description for “person”?
- ...
- In short, syntax alone cannot solve these problems

Hideaki Takeda / National Institute of Informatics

NII

Architecture for the Semantic Web



Tim Berners-Lee <http://www.w3.org/2002/Talks/09-lcs-sweb-tbl/>

Hideaki Takeda / National Institute of Informatics

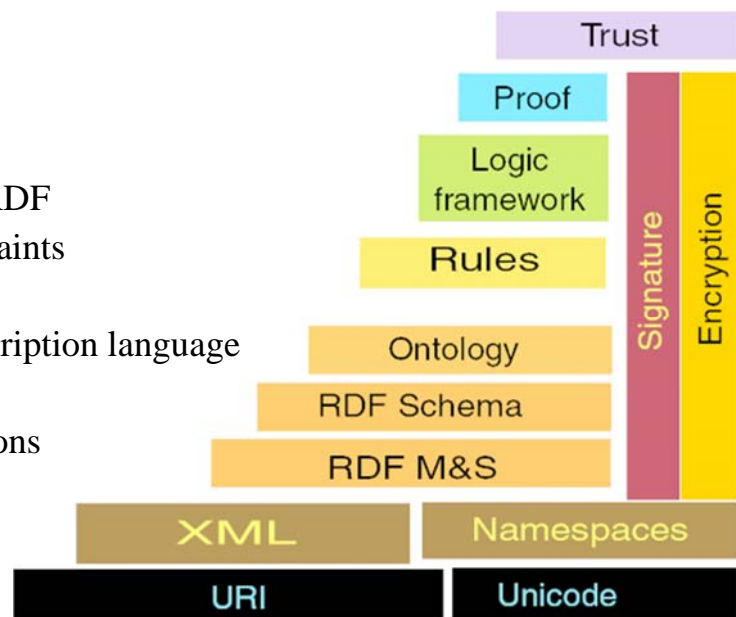
NII

How to describe “meaning”?

- Need to describe “information on information”
 - “Meaning of something” is a description (“meaning”) to a description (“something”) in computers
 - Metadata
 - ◆ Data about data
- Need to architecture for common understanding
 - Syntax (language or scheme)
 - Vocabulary (ontology)

A Layer model for Semantic Web

- RDF (Resource Description Framework)
 - The most primitive model for metadata description
 - ◆ SVO model
 - ◆ Entity-Relation Model
 - ◆ Semantic net
- RDF Schema
 - Addition of “concept” to RDF
 - ◆ class-subclass, constraints
- OWL
 - More general concept description language
 - ◆ Logical consistency
 - ◆ Various class expressions
 - ◆ Various constraints
- DAML-S
 - Descriptions on processes



Two origin of Semantic Web

- From the viewpoint of Web Community
 - More meaning, more knowledge
 - HTML -> XML -> RDF -> RDF Schema -> OWL
->XML schema ->
- From the viewpoint of Knowledge Sharing Community (AI)
 - From closed knowledge presentation to open KR
 - ◆ Not for depth but for width
 - ◆ Not for completeness, but for possibility
 - KIF, Ontolinga -> OKBC -> OWL

TOC

- What is Semantic Web?
- Realization of Semantic Web
 - Metadata
 - Dublin Core
 - RDF
 - RDF Schema
 - OWL

Metadata

- What is metadata?
 - Data about data
 - What one can say about any information object
- What is described as metadata?
 - **Content** relates to what the object contains or is about, and is *intrinsic* to an information object.
 - **Context** indicates the who, what, why, where, how aspects associated with the object's creation and is *extrinsic* to an information object.
 - **Structure** relates to the formal set of associations within or among individual information objects and can be *intrinsic* or *extrinsic*

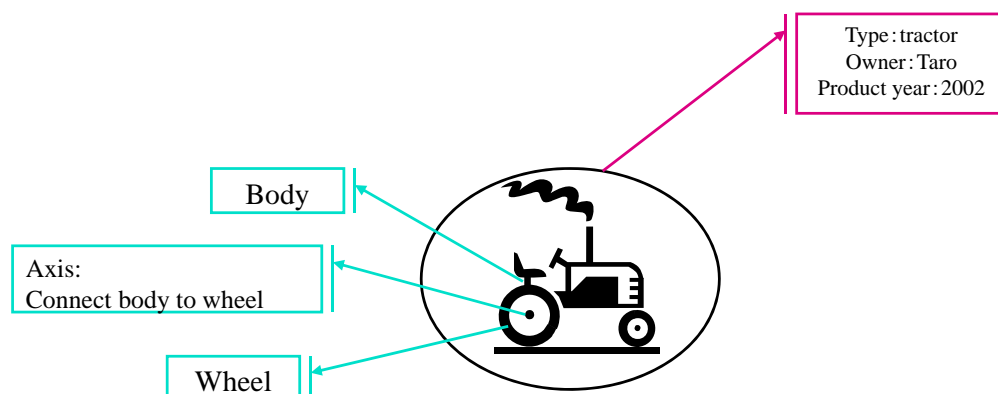
Setting the State, Anne J.Gilliand-Swetland, Introduction to Metadata – Pathways to Digital Information, Murthsa Baca (ed.), Getty Information Institute.

Hideaki Takeda / National Institute of Informatics

NII ■

Metadata

- Metadata to individual information objects
 - Bibliography, Dublin Core
- Metadata to part or structure of information objects
 - Drawings, RDF, RDFS, OWL



Hideaki Takeda / National Institute of Informatics

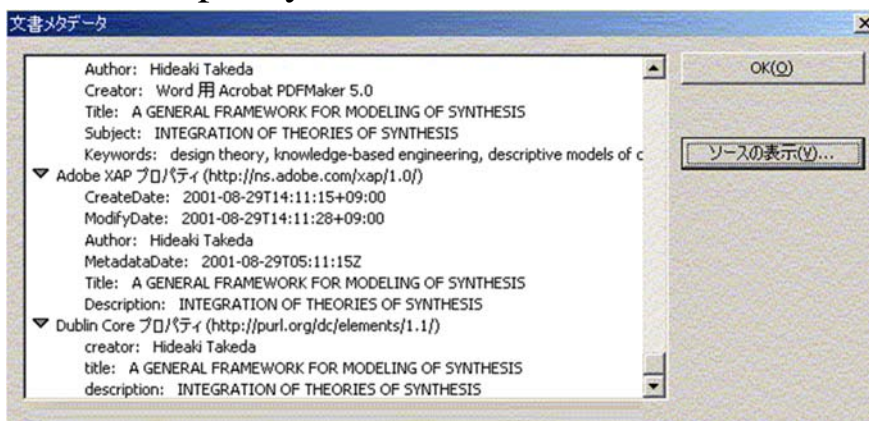
NII ■

Dublin Core

- Framework for “bibliographic” style metadata
- Simplicity “Pidgins”
 - A few vocabulary
 - A simple structure
- 15 elements: Dublin Core Metadata Element Set
 - Creator, Title, Subject/Keywords, Description, Publisher, Contributor, Date, Resource Type, Format, Resource Identifier, Source, , Language, Relation, Coverage, Rights Management
 - All elements can be omitted and duplicated in any order

Dublin Core

- An example by PDF



```
<rdf:Description about="
  xmlns='http://purl.org/dc/elements/1.1/'
  xmlns:dc='http://purl.org/dc/elements/1.1/'>
  <dc:creator>Hideaki Takeda</dc:creator>
  <dc:title>A GENERAL FRAMEWORK FOR MODELING
OF SYNTHESIS</dc:title>
  <dc:description>INTEGRATION OF THEORIES OF
SYNTHESIS</dc:description>
</rdf:Description>
```

RDF (Resource Description Framework)

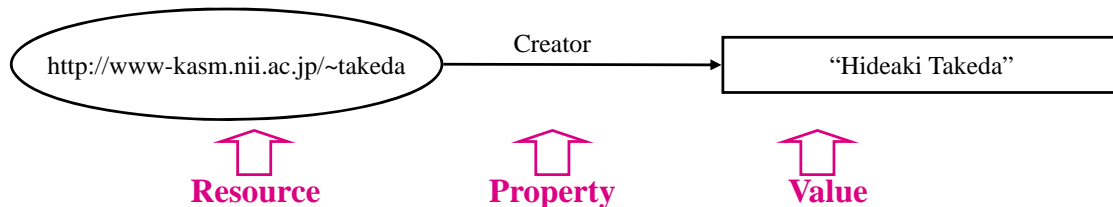
- A framework to describe metadata
- Separation of model and syntax
- W3C Recommendation (2004)

RDF Model

- Element
 - Resource:
 - ◆ URI(Universal Resource Identifier)
 - ◆ Literal(string)
 - No need to be specified by Web
 - Property:
 - ◆ Attribute when describing resources
 - ◆ URI or Literal just as Resource
 - Statement: triad of resource, property, and resource

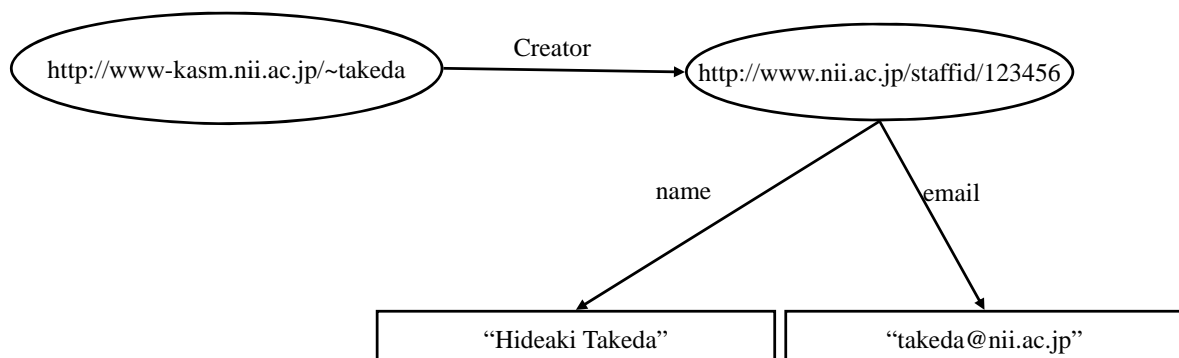
RDF model

- Statement
 - Creator of <http://www-kasm.nii.ac.jp/~takeda> is “Hideaki Takeda”
- Structure
 - Resource (subject): <http://www-kasm.nii.ac.jp/~takeda>
 - Property (predicate): Creator
 - Value (object): “Hideaki Takeda”



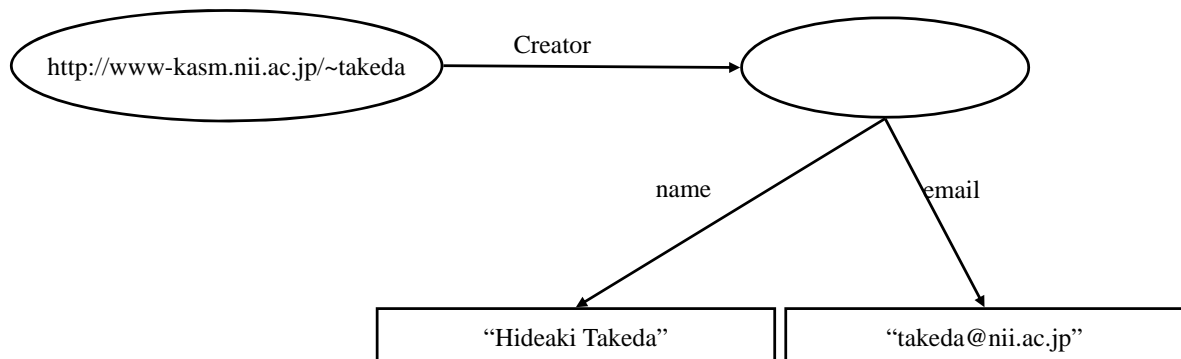
RDF model

- Creator of <http://www-kasm.nii.ac.jp/~takeda> is <http://www.nii.ac.jp/staffid/123456> which has name “Hideaki Takeda” and email “takeda@nii.ac.jp” .



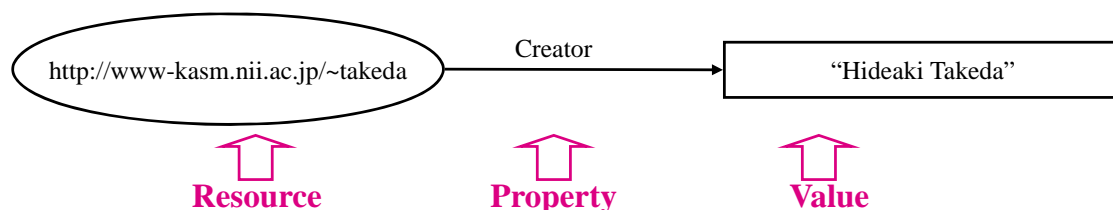
RDF model

- Creator of <http://www-kasm.nii.ac.jp/~takeda> has name “Hideaki Takeda” email “takeda@nii.ac.jp” .



RDF syntax

- Creator of <http://www-kasm.nii.ac.jp/~takeda> is “Hideaki Takeda”

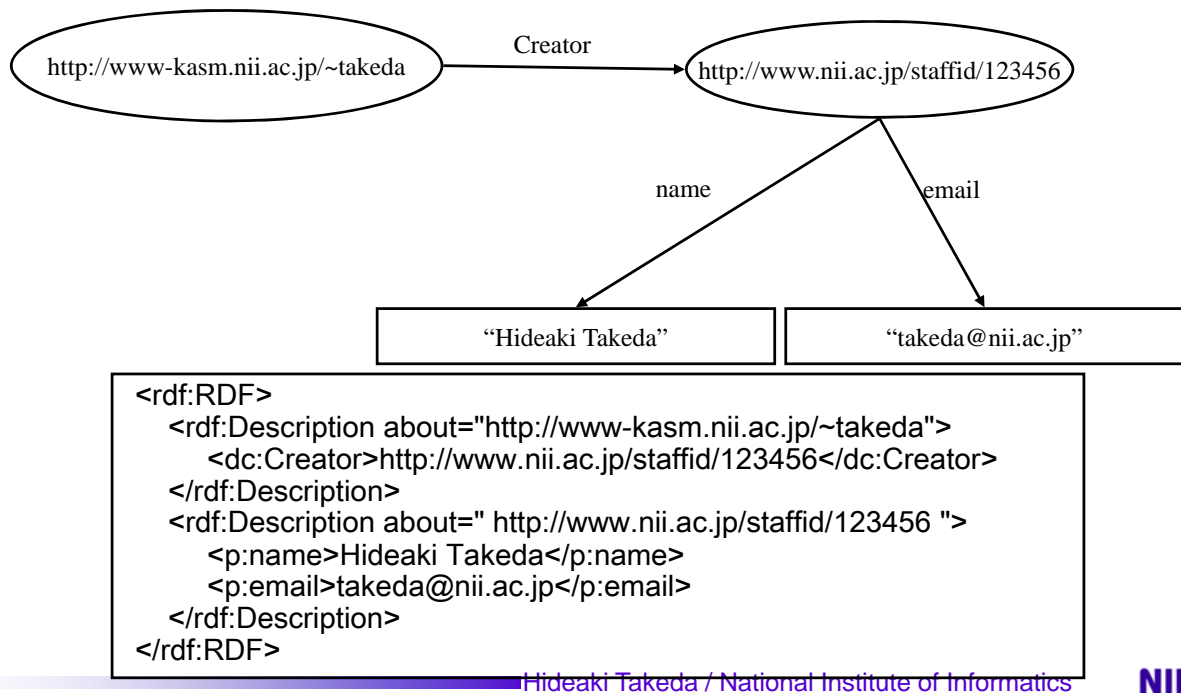


```
<?xml version="1.0"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://dublincore.org/2001/08/14/dces#">
  <rdf:Description about="http://www-kasm.nii.ac.jp/~takeda">
    <dc:Creator>Hideaki Takeda</dc:Creator>
  </rdf:Description>
</rdf:RDF>
```

```
<rdf:RDF>
  <rdf:Description about="http://www-kasm.nii.ac.jp/~takeda">
    <dc:Creator rdf:resource="Hideaki Takeda" />
  </rdf:Description>
</rdf:RDF>
```

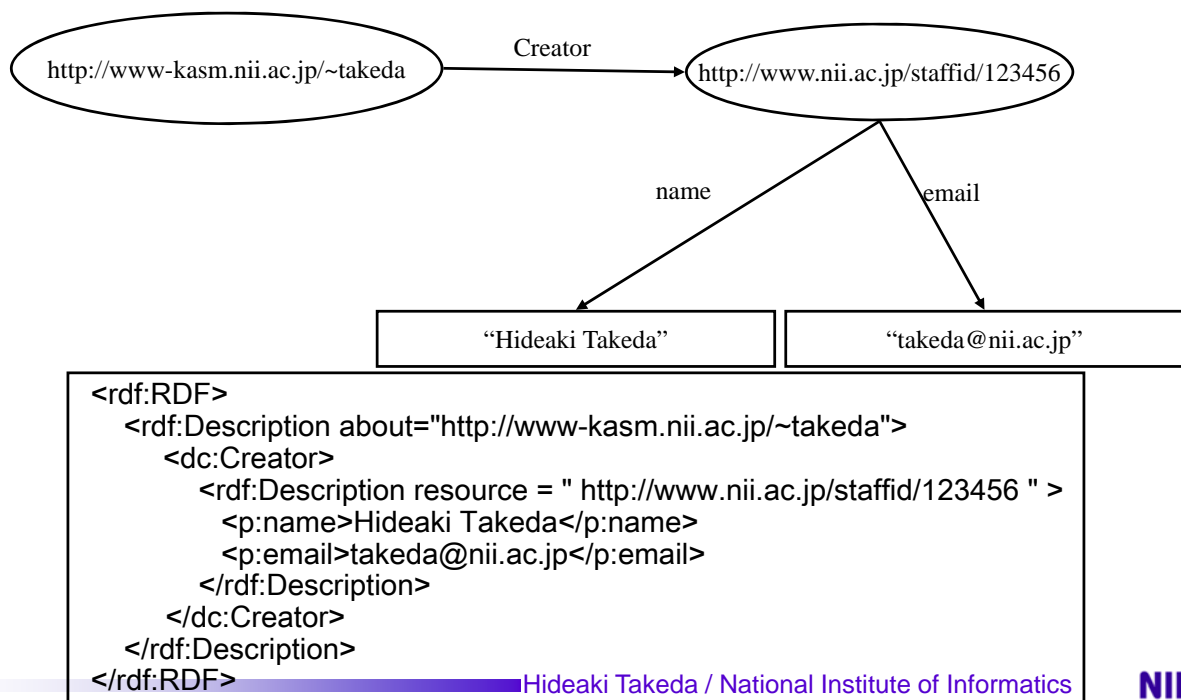
RDF syntax

- Creator of <http://www-kasm.nii.ac.jp/~takeda> is <http://www.nii.ac.jp/staffid/123456> which has name “Hideaki Takeda” and email “takeda@nii.ac.jp”



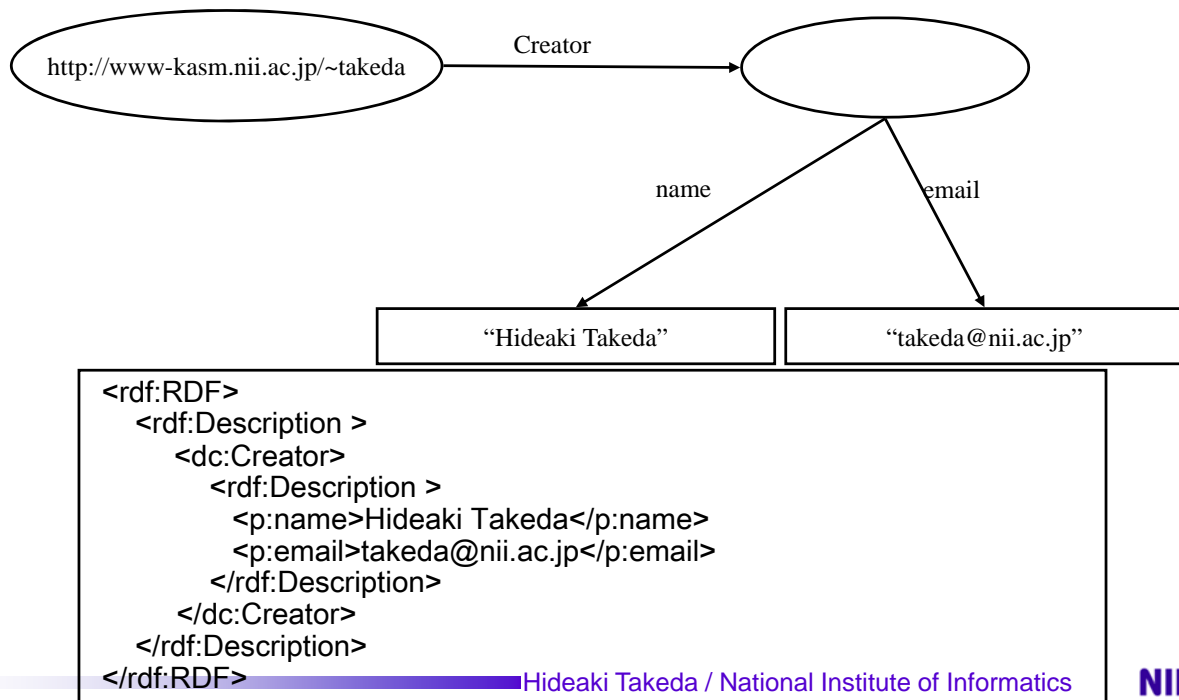
RDF syntax

- Creator of <http://www-kasm.nii.ac.jp/~takeda> is <http://www.nii.ac.jp/staffid/123456> which has name “Hideaki Takeda” and email “takeda@nii.ac.jp” .



RDF syntax

Creator of <http://www-kasm.nii.ac.jp/~takeda> has name “Hideaki Takeda” email “takeda@nii.ac.jp” .

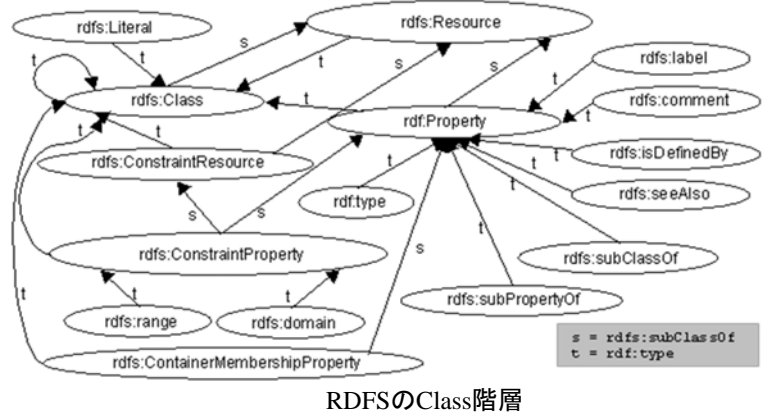
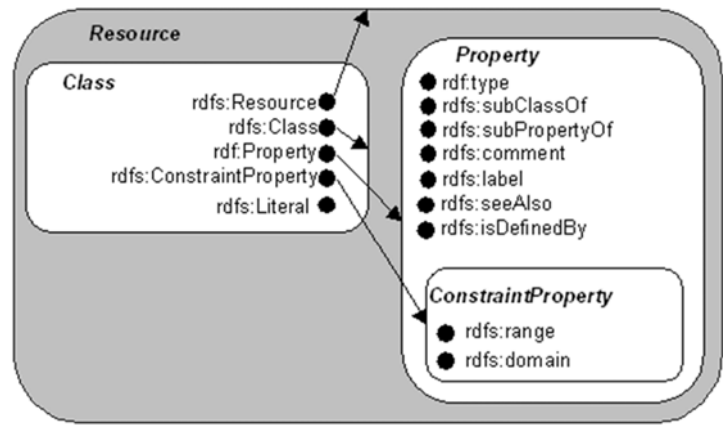


RDFS (RDF Schema)

- Stronger knowledge representation model
 - RDF: ER model, semantic net
 - RDF Schema: Frame model, object-oriented paradigm
 - ◆ Minimal definition
 - ◆ Property-centered approach
- RDFS is defined as extension of RDF
- RDFS gives definitions of RDF descriptions

RDFS

- Class Definition
 - rdfs:Resource
 - rdfs:Class
 - rdf:Property
 - rdfs:ConstraintProperty
 - rdfs:Literal
- Property Definition
 - rdf:type
 - rdfs:subClassOf
 - rdfs:subPropertyOf
 - rdfs:comment
 - rdfs:label
 - rdfs:seeAlso
 - rdfs:isDefinedBy
- ConstraintProperty Definition
 - rdfs:range
 - rdfs:domain



Resource Description Framework(RDF) Schema Specification 1.0
<http://www.w3.org/TR/2000/CR-rdf-schema-20000327/>

RDF Schema

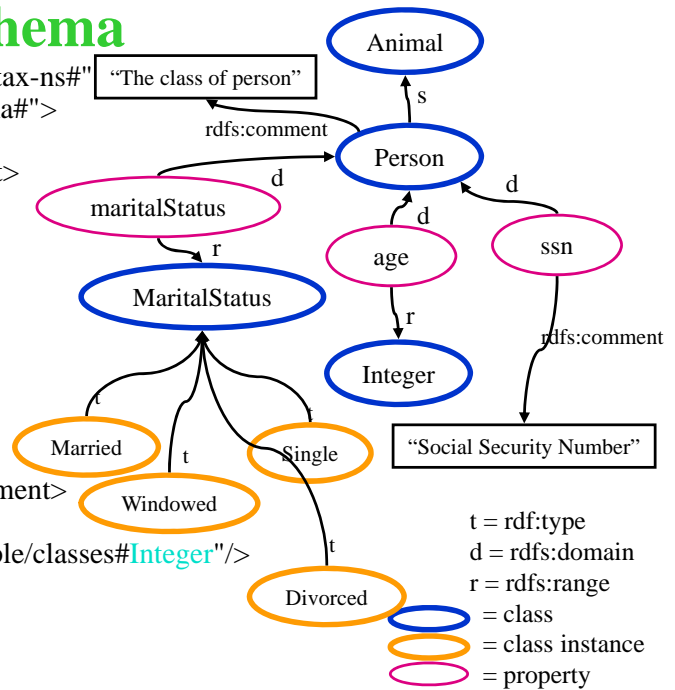
- rdfs:Class
 - Detailed class
 - Multiple
 - Transitivity
- rdf:type
 - Indicate an instance of a class
- rdf:property
 - Attribute
- rdfs:subPropertyOf
 - Detailed property
 - Transitivity
- Range
 - Only one
 - ◆ No cardinality
- Domain
 - Multiple (or)

RDF Schema

```

<rdf:RDF xml:lang="en"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
<rdfs:Class rdf:ID="Person">
<rdfs:comment>The class of people.</rdfs:comment>
<rdfs:subClassOf rdf:resource="http://www.w3.org/
  2000/03/example/
  classes#Animal"/>
</rdfs:Class>
<rdf:Property ID="maritalStatus">
<rdfs:range rdf:resource="#MaritalStatus"/>
<rdfs:domain rdf:resource="#Person"/>
</rdf:Property>
<rdf:Property ID="ssn">
<rdfs:comment>Social Security Number</rdfs:comment>
<rdfs:range
  rdf:resource="http://www.w3.org/2000/03/example/classes#Integer"/>
<rdfs:domain rdf:resource="#Person"/>
</rdf:Property>
<rdf:Property ID="age">
<rdfs:range
  rdf:resource="http://www.w3.org/2000/03/example/classes#Integer"/>
<rdfs:domain rdf:resource="#Person"/>
</rdf:Property>
<rdfs:Class rdf:ID="MaritalStatus"/>
<MaritalStatus rdf:ID="Married"/>
<MaritalStatus rdf:ID="Divorced"/>
<MaritalStatus rdf:ID="Single"/>
<MaritalStatus rdf:ID="Widowed"/>
</rdf:RDF>

```



Resource Description Framework(RDF) Schema Specification 1.0
<http://www.w3.org/TR/2000/CR-rdf-schema-20000327/>

Hideaki Takeda / National Institute of Informatics

NII

OWL(Web Ontology Language)

- More general knowledge representation
- Based on Description Logics
- Features
 - Class
 - ◆ Necessary condition / necessary and sufficient condition
 - ◆ Class expression:
 - Constraint by property
 - Like slot definition of a class
 - Type constraint (all/some), cardinality, typed cardinality
 - Logical operation of classes: union, intersection, negation
 - Property
 - ◆ Multiple ranges and domains
 - ◆ Specifying meta-property
 - Import of definitions

Hideaki Takeda / National Institute of Informatics

NII

Class descriptions

- 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: Enumeration

- **owl:oneOf** [instance]+

```
<owl:oneOf parseType="owl:collection">
  <owl:Thing rdf:about="#Eurasia"/>
  <owl:Thing rdf:about="#Africa"/>
  <owl:Thing rdf:about="#North_America"/>
  <owl:Thing rdf:about="#South_America "/>
  <owl:Thing rdf:about="#Australia"/>
  <owl:Thing rdf:about="#Antarctica"/> </oneOf>
```

OWL

- Property restriction
 - A special kind of **class description**. It describes an anonymous class, namely a class of all individuals that satisfy the restriction

OWL: Value constraints

- A restriction class to either a class description or a data range
- **owl:allValuesFrom**
 - All individuals must satisfy the specific condition (class or data range)

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#hasParent" />
  <owl:allValuesFrom rdf:resource="#Human" />
</owl:Restriction>
```
- **owl:someValuesFrom**
 - At least one property value must satisfy the specific condition

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#hasParent" />
  <owl:someValuesFrom rdf:resource="#Physician" />
</owl:Restriction>
```
- **owl:hasValue**
 - Constrain the value as the specific individual

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#hasParent" />
  <owl:hasValue rdf:resource="#Clinton" />
</owl:Restriction>
```

OWL: Cardinality constraints

- to allow only a specific number of values
- **owl:maxCardinality**
 - A restriction containing an **owl:maxCardinality** constraint describes a class of all individuals that have at most N semantically distinct values

```
<owl:Restriction>  
  <owl:onProperty rdf:resource="#hasParent" />  
  <owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger">2</owl:maxCardinality>  
</owl:Restriction>
```

- **owl:minCardinality**
- **owl:cardinality**

OWL: Boolean Operators

- **owl:intersectionOf** [*class-expression*]⁺
 - a class for which the class extension contains precisely those individuals that are members of the class extension of all class descriptions in the list.
- **owl:unionOf** [*class-expression*]⁺
 - an anonymous class for which the class extension contains those individuals that occur in at least one of the class extensions of the class descriptions in the list
- **owl:complementOf** *class-expression*
 - a class for which the class extension contains exactly those individuals that do not belong to the class extension of the class description that is the object of the statement

```
<complementOf>  
  <Class>  
    <unionOf parseType="owl:collection">  
      <Class rdf:resource="#meat"/>  
      <Class rdf:resource="#fish"/>  
    </unionOf>  
  </Class>  
</complementOf>
```

OWL

- Class axioms
 - **owl:Class** *class-name(URI)*
 - **owl:subClassOf** *class-expression*
 - **owl:disjointWith** *class-expression*
 - **owl:equivalentClass** *class-expression*
- } necessary condition
- necessary and sufficient condition

```
<owl:Class rdf:about="#Opera">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasLibrettist" />
      <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

OWL: Property elements

- RDF/RDFS Constraints
 - **rdf:Property** *property-name*
 - **rdfs:subPropertyOf**
 - **rdfs:domain** *class-expression*
 - ◆ If there are multiple descriptions, they express intersections
 - **rdfs:range** *class-expression*
 - ◆ If there are multiple descriptions, they express intersections
- Restrictions to other properties
 - **owl:equivalentProperty** *property-name*
 - **owl:inverseOf** *property-name*
- Global cardinality constraints
 - **owl:FunctionalProperty**
 - **owl:InverseFunctionalProperty**
- Logical characteristics
 - **owl:SymmetricProperty**
 - **owl:TransitiveProperty**

OWL: Instances

- Instance
 - Instance for a class or property
 - **rdf:type**

```
<continent rdf:ID="Asia"/>

<rdf:Description rdf:ID="Asia">
  <rdf:type>
    <rdfs:Class rdf:about="#continent"/>
  </rdf:type>
</rdf:Description>

<rdf:Description rdf:ID="India">
  <is_part_of rdf:resource="#Asia"/>
</rdf:Description>
```

OWL

- Object and datatype
 - datatype domain: defined in XML Schema
 - object domain: defined in OWL

Three OWL Languages

- OWL-Full
 - OWL-DL + all RDFS expressions
 - Logically very complicated
- OWL-DL
 - The basic OWL
 - Full use of OWL language constructs
- OWL-Lite
 - Intended to be used in simple applications
 - Restrictions of OWL language constructs
 - ◆ E.g, No use of these constructs
 - [owl:oneOf](#)
 - [owl:unionOf](#)
 - [owl:complementOf](#)
 - [owl:hasValue](#)
 - [owl:disjointWith](#)
 - [owl:DataRange](#)

OWL example

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
  <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
]>

<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns="http://www.example.com/person.owl#"
  xml:base="http://www.example.com/person.owl">

  <owl:Ontology rdf:about="">
  <rdfs:comment>
    An example ontology by Takeda, with data types taken from XML Schema
  </rdfs:comment>
</owl:Ontology>
```

```
<owl:Class rdf:ID="Animal">
  <rdfs:label>Animal</rdfs:label>
  <rdfs:comment>
    This class of animals is illustrative of
    a number of ontological idioms.
  </rdfs:comment>
</owl:Class>

<owl:ObjectProperty rdf:ID="name">
  <rdfs:domain rdf:resource="#Animal"/>
  <rdfs:range rdf:resource="&xsd:string" />
</owl:ObjectProperty>

<owl:DatatypeProperty rdf:ID="age">
  <rdfs:comment>
    age is a DatatypeProperty whose range is
    xsd:decimal. age is also a UniqueProperty
    (can only have one age)
  </rdfs:comment>
  <rdf:type rdf:resource=
    "&owl;FunctionalProperty" />
  <rdfs:domain rdf:resource="#Animal" />
  <rdfs:range rdf:resource=
    "&xsd;positiveInteger" />
</owl:DatatypeProperty>
```

```
<owl:Class rdf:ID="Person">
  <rdfs:subClassOf rdf:resource="#Animal"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:about="#name" />
      <owl:minCardinality rdf:datatype=
        "http://www.w3.org/2001/XMLSchema#int">
        1
      </owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>

<Person rdf:ID="Hideaki">
  <rdfs:label>Hideaki</rdfs:label>
  <rdfs:comment>Hideaki is a person. His name is
  Hideaki Takeda and his age is
  100.</rdfs:comment>
  <name>Hideaki Takeda</name>
  <age rdf:datatype="&xsd;positiveInteger">
    100</age>
</Person>
</rdf:RDF>
```

The Secret of Semantic Web

- Why can Semantic Web work?
 - URI, URI, URI
 - ◆ The global identifier
 - ◆ If two people refer the single URL, it is assumed that they share concepts *implicitly* and *optimistically*
 - But it may be too optimistic
 - ◆ More various sharing ways are needed

Tasks for Semantic Web

- Developing languages and processing systems (SW people)
- Propagation of SW (SW people, Application People, Users)
- Providing ontologies (SW people, Application People)
- Applying domains (Application People)

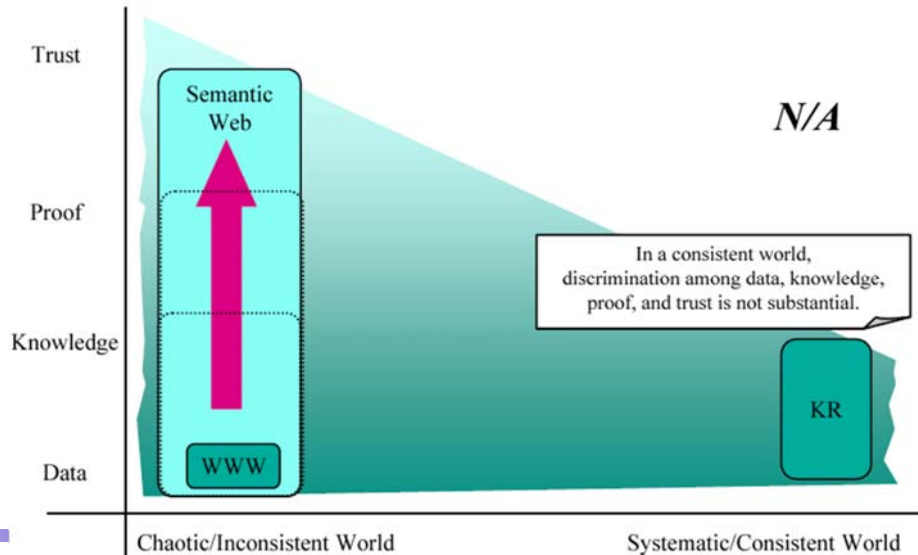
- Loop between specification developers and users is common problem.
- But the problem is crucial in Semantic Web
 - Ontologies are meaningful *if-and-only-if* they are shared
 - ◆ C.f. XML is useful even if it is used in a small community

Semantic Web and Knowledge Representation

- Is Semantic Web just “re-inventing a wheel”?
 - Renewal of existing DL-based languages?
 - No...The Environment is different!
 - ◆ KR is based on the ideal environment
 - ◆ WWW is a real-world environment
- Openness of Knowledge
- Human-in-the-Loop in Knowledge Creation

Openness of Knowledge

- Knowledge on WWW is *always* inconsistent and incomplete
- Traditional KR assumes consistence and completeness ideally
 - Tasks
 - ◆ Incomplete and inconsistent KR



Human-in-the-Loop in Knowledge Creation

- Building very large knowledge bases
 - In-house or made in laboratories
 - ◆ Small experts
 - ◆ e.g, Cyc and EDR
 - ◆ Not so successful
 - Open development
 - ◆ Like Linux
 - ◆ Everyone can contribute
 - ◆ Quality?
 - ◆ e.g, Open Directory Project, OpenMind project, MusicBrainz