

Intelligent Systems on the World Wide Web

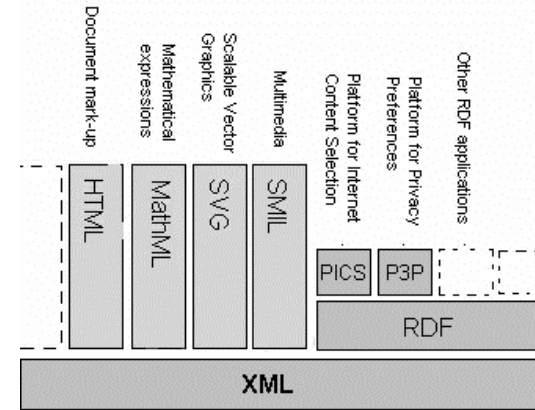
RDF

Lecture Slides
Steffen Staab

Institute for Applied Computer Science and Formal
Description Methods (AIFB)
Karlsruhe University

RDF

Applications and Technologies on top of XML



RDF

Outline

- Motivation: Why XML is not enough
- Introduction to RDF
 - Requirements for KR on the Web
 - The RDF Data Model
 - RDF Schema
- Extensions of RDF(S)
- Tools for RDF and RDF Schema
 - Parser, Query, and Inference Engines

RDF

Why The Shift Towards More Semantics?

- Information Overload
 - Information on the Web currently aiming at Human Consumption
 - Information Consumption is too time consuming
- Search Engines fail more and more
 - combined coverage is less than 42% of the HTML-Web
- Data Interchange growing (eg. B2B)
 - needs a common semantics

RDF

XML?

Extensible Markup Language (XML) Revisited

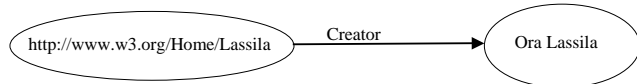
- Key idea: separate structure from presentation
- XML DTDs or Schema define document structure
- Replace HTML with two things
 - A domain specific markup language (defined in XML)
 - A map from that markup language to HTML (defined using XSL)
- DTD enables document recipients to tell whether they've received a well-formed document
 - Gives a minimal level of validation

Why XML is not enough

- Only advantage of using XML is reusing the parser and document validation
 - Many different possibilities to encode a domain of discourse
 - Leads to difficulties when understanding of foreign documents is required
- ➔ Next step: separate content from structure!

Encoding of Knowledge: Example

“The Creator of the Resource “http://www.w3.org/Home/Lassila” is Ora Lassila



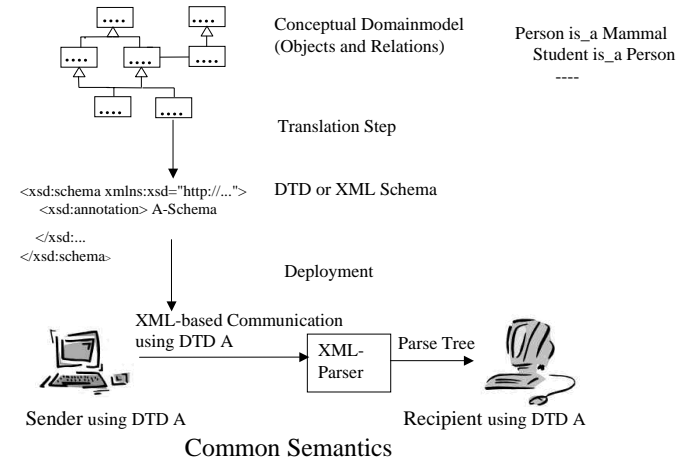
Endless encoding possibilities in XML:

```
<Creator>
  <uri> http://www.w3.org/Home/Lassila </uri>
  <name>Ora Lassila</name>
</Creator>
```

```
<Document uri="http://www.w3.org/Home/Lassila"
  <Creator>Ora Lassila</Creator>
</Document>
```

```
<Document uri="http://www.w3.org/Home/Lassila" Creator="Ora Lassila"/>
```

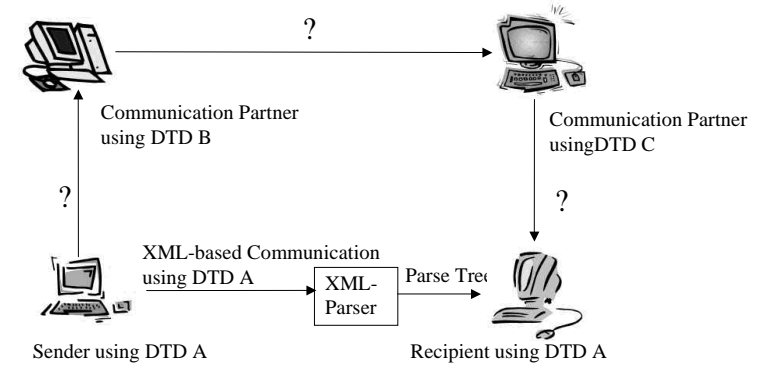
Point to Point Communication for Machine-Understandable Data



Many previously unknown Communication Partners



New Partner don't understand each other



Merging Steps between Models

Steps	DTD A	DTD B
Reengineering of the conceptual model	<pre><xsd:schema xmlns:xsd="http://..."> <xsd:annotation>A-Schema </xsd:... </xsd:schema-</pre>	<pre><xsd:schema xmlns:xsd="http://..."> <xsd:annotation>B-Schema </xsd:... </xsd:schema></pre>
Matching		
XML Document Translation Generation (e.g. in XSLT)	<pre><xsl:stylesheet version="1.0" xmlns:xsl="http://...Transform" <xsl:template match="/"> </xsl:template> </xsl:stylesheet></pre>	<pre><xsl:stylesheet version="1.0" xmlns:xsl="http://...Transform" <xsl:template match="/"> </xsl:template> </xsl:stylesheet></pre>
XML Document Translation from DTD A to DTD B (and B to A)		

Merging/Aligning Models

- Reengineering step is costly and unnecessary, when a conceptual language is use
- Generation document translation procedures is again complicated and unnecessary
- ➔ use a level on top of XML
- What are Requirements for such a level?

Postulates: Fundamental Requirements for KR on the Web

RDF

- 1.) Knowledge on the Web is distributed (link Knowledge on the Web)
- 2.) Knowledge on the Web is biased - there is no universal truth
it must be possible to dispute statements
- 3.) Many different user communities:
Extensibility and Simplicity

➔ Resource Description Framework (RDF)

Slide 13

Introduction to RDF

RDF

- RDF (Resource Description Framework)
 - Beyond Machine readable to *Machine understandable*
- RDF unites a wide variety of stakeholders:
 - Digital librarians, content-raters, privacy advocates, B2B industries, AI...
 - Significant (but less than XML) industrial momentum, lead by W3C
- RDF consists of two parts
 - RDF Model (a set of triples)
 - RDF Syntax (different XML serialization syntaxes)
- RDF Schema for definition of Vocabularies (simple Ontologies) for RDF (and in RDF)

Slide 14

RDF Data Model

RDF

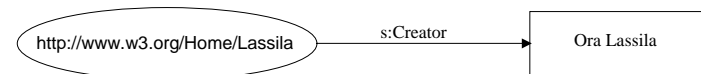
- **Resources**
 - A resource is a thing you talk about (can reference)
 - Resources have URI's
 - RDF definitions are itself Resources (linkage)
- **Properties**
 - slots, defines relationship to other resources or atomic values
- **Statements**
 - “Resource has Property with Value”
 - (Values can be resources or atomic XML data)
- **Similar to Frame Systems**

Slide 15

A simple Example

RDF

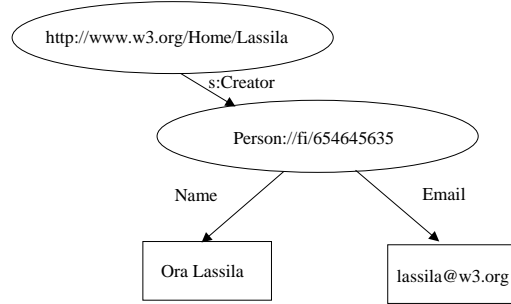
- **Statement**
 - “Ora Lassila is the creator of the resource
<http://www.w3.org/Home/Lassila>”
- **Structure**
 - Resource (subject) <http://www.w3.org/Home/Lassila>
 - Property (predicate) <http://www.schema.org/#Creator>
 - Value (object) “Ora Lassila”
- **Directed graph**



Slide 16

Another Example

- To add properties to Creator, point through a intermediate Resource.



Slide 17

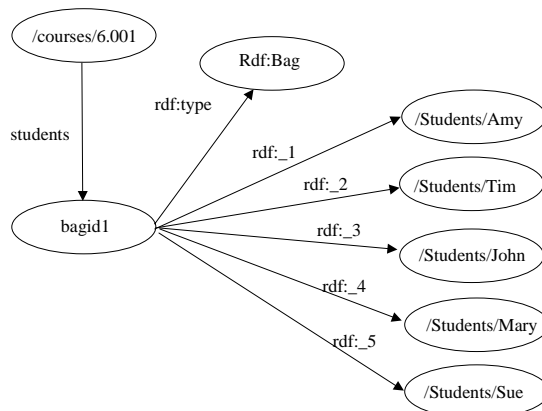
Collection Containers

- Multiple occurrences of the same PropertyType doesn't establish a relation between the values
 - The Millers own a boat, a bike, and a TV set
 - The Millers need (a car or a truck)
 - (Sarah and Bob) bought a new car
- RDF defines three special Resources:
 - Bag** unordered values rdf:Bag
 - Sequence** ordered values rdf:Seq
 - Alternative** single value rdf:Alt
 - Core RDF does not enforce 'set' semantics amongst values

Slide 18

Example: Bag

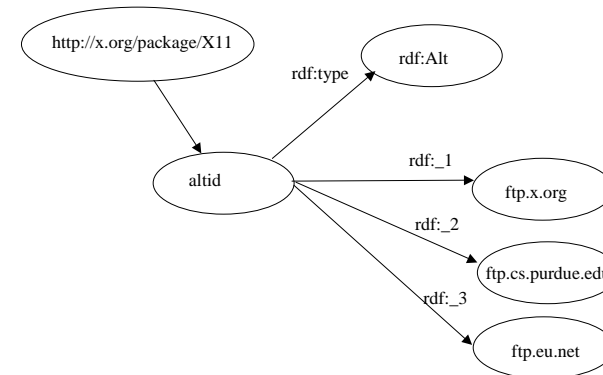
- The students in course 6.001 are Amy, Tim, John, Mary, and Sue



Slide 19

Example: Alternative

- The source code for X11 may be found at ftp.x.org, ftp.cs.purdue.edu, or ftp.eu.net



Slide 20

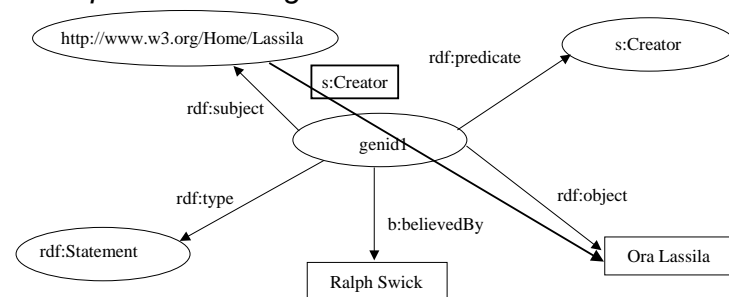
Statements about Statements (Requirement 2: Dispute Statements)

- Making statements about *statements* requires a process for transforming them into Resources
 - **subject** the original referent
 - **predicate** the original property type
 - **object** the original value
 - **type** rdf:Statement

Slide 21

Example: Reification

- *Ralph Swick believes that*
 - *the creator of the resource*
http://www.w3.org/Home/Lassila is Ora Lassila



Slide 22

A Formal Model of RDF

- RDF itself is mathematically straightforward:
 - Basic Definitions
 - Resources.
 - Properties \subset Resources called
 - Literals
 - Statements = Resources \times Properties \times {Resources \cup Literals}
 - Typing
 - `rdf:type` \in Properties
 - {`RDF:type`, sub, obj} \in Statements \Rightarrow obj \in Resources

Slide 23

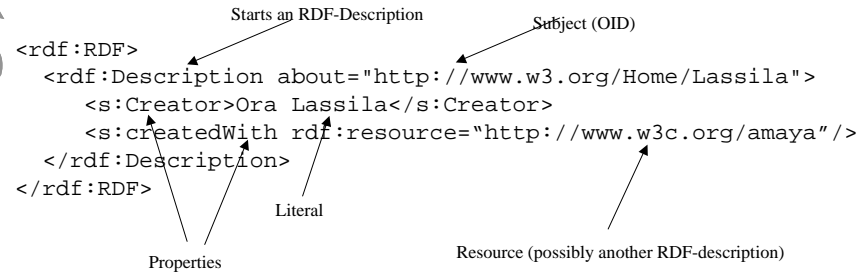
Formal Model of RDF II

- Reification
 - `rdf:Statement` \in Resource-Properties
 - {`rdf:predicate`, `rdf:subject`, `rdf:object`} \subset Properties
 - Reification of a triple { pred, sub, obj } of Statements is an element `r` of Resources representing the reified triple and the elements `s1`, `s2`, `s3`, and `s4` of Statements such that
 - `s1`: {`RDF:predicate`, `r`, pred}
 - `s2`: {`RDF:subject`, `r`, subj}
 - `s3`: {`RDF:object`, `r`, obj}
 - `s4`: {`RDF:type`, `r`, [`RDF:Statement`]}
- Collections
 - { `RDF:Seq`, `RDF:Bag`, and `RDF:Alt` } \subset Resource-Properties
 - There is a subset of Properties corresponding to the ordinals (1, 2, 3, ...) called Ord. We refer to
 - elements of Ord as `RDF:_1`, `RDF:_2`, `RDF:_3`, ...

Slide 24

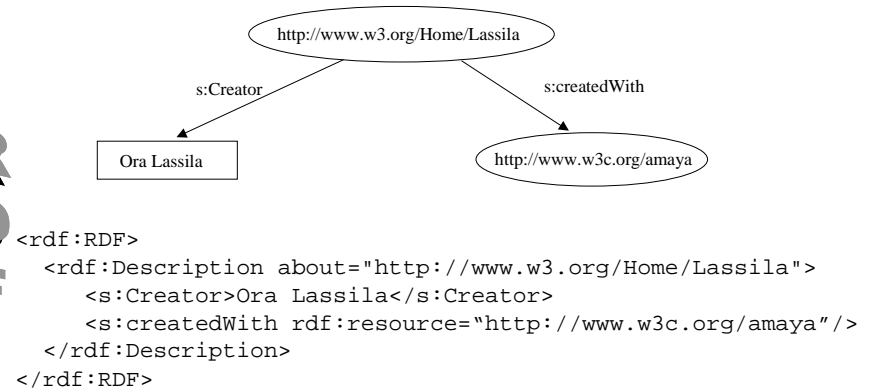
RDF Syntax I

- Datamodel does not enforce particular syntax
- Specification suggests many different syntaxes based on XML
- General form:



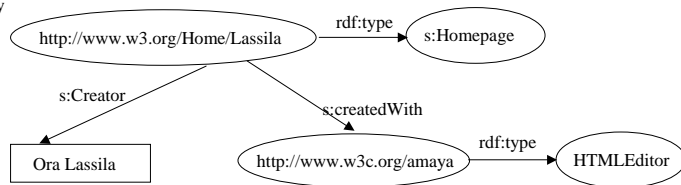
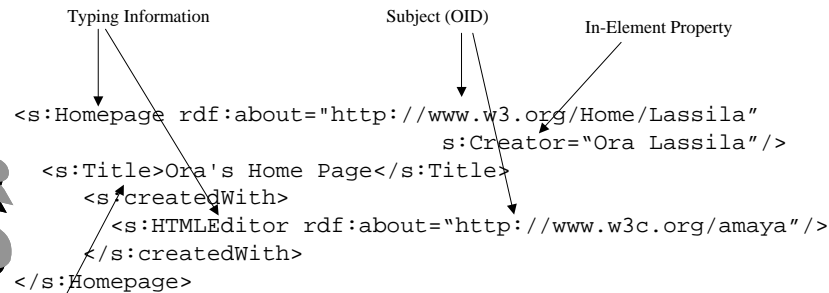
Slide 25

Resulting Graph



Slide 26

RDF Syntax II: Syntactic Varieties



Slide 27

RDF Schema (RDFS)

- RDF just defines the datamodel
- Need for definition of vocabularies for the datamodel - an Ontology Language!
- RDF schemas are Web resources (and have URIs) and can be described using RDF

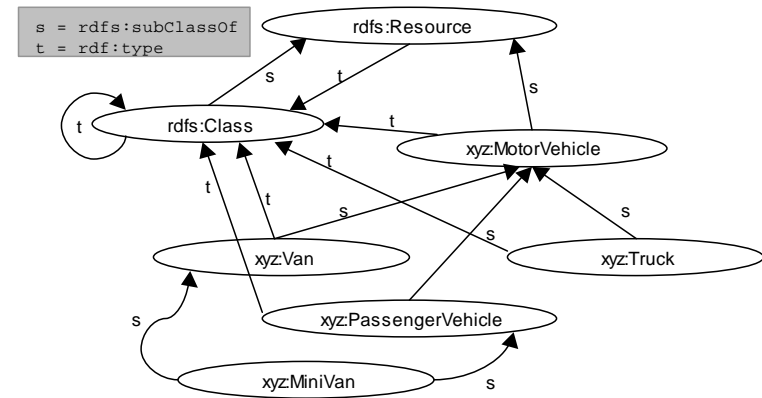
Slide 28

Most Important Modeling Primitives

- Core Classes
 - Root-Class `rdfs:Resource`
 - MetaClass `rdfs:Class`
 - Literals `rdfs:Literal`
- `rdfs:subClassOf`-property
- Inherited from RDF: properties (slots)
- `rdfs:domain` & `rdfs:range`
- `rdfs:label`, `rdfs:comment`, etc.
- Inherited from RDF: InstanceOf (`rdf:type`)

Slide 29

RDF-Schema: Example



Slide 30

Rdfs:subClassOf

```

<rdfs:description about=„Xyz:Minivan">
  <rdfs:subClassOf about=„xyz:Van"/>
</rdfs:description>
<rdfs:description about=„myvan">
  <rdf:type about=„xyz:MiniVan"/>
</rdfs:description>
  
```

Predicate Logic Consequences:

Forall X: `type(X,MiniVan) -> type(X, Van)`.
 Forall X: `subclassOf(X,MiniVan) -> subclassOf(X, Van)`.

Slide 31

Rdf:property

```

<rdf:description about=„possesses">
  <rdf:type about=„....property"/>
  <rdfs:domain about=„person"/>
  <rdfs:range about=„vehicle"/>
</rdf:description>
<rdf:description about=„peter">
  <possesses>petersminivan</possesses>
</rdf:description>
  
```

Predicate Logic Consequences:

Forall X,Y: `possesses (X,Y) -> (type(X,person) & type(Y,vehicle))`.

Slide 32

Proxy Resource

```
<rdf:RDF xmlns:rdf="..." xmlns:xx="..." >
  <rdf:Description ID="John_Cowan">
    <xx:email>cowan@ccil.org</xx:email>
    <xx:email>jcowan@haxmail.com</xx:email>
    <xx:homepage
      resource="http://www.ccil.org/~cowan"/>
  </rdf:Description>
</rdf:RDF>
```

Slide 33

Attributes instead of elements

A property/value pair can be expressed as an attribute instead of a sub-element if:

- The value is a string
- There is only one value for the property

```
<rdf:Description ID="John_Cowan"
  email="cowan@ccil.org">
  <xx:homepage resource="..." />
</rdf:Description>
```

Slide 34

Nested descriptions

- When a value is a proxy resource, it may be placed inside the property element

```
<rdf:Description about="...">
  <xx:author>
    <rdf:Description ID="John_Cowan">
      ..
    </rdf:Description>
  </xx:author>
</rdf:Description>
```

Slide 35

Simplification rules

- People specifying text for arbitrary RDF processors can use any simplification
- Processors of arbitrary RDF therefore must accept all simplifications
- Special-purpose XML formats can be RDF-compliant while disallowing simplifications, requiring them, or exploiting them in specific ways

Slide 36

Containers

Proxy resources that hold one or more values (Web resources or strings)

- Bag: a simple collection with no ordering
- Seq: a collection with implicit ordering
- Alt: a set of alternatives (first one preferred)
- Elements are properties named `_1`, `_2`, `_3`, etc.

Slide 37

Container examples

- Bag: committee members, documents in a folder, checks in a bag
- Seq: book authors (order counts!), chapters in a book, items in an agenda
- Alt: document home and mirrors, mailing-list moderators, translations of a document

Slide 38

A Bag, the hard way

```
<rdf:Description ID="committee">
  <rdf:type
    resource="http://www.w3.org/1999/02/22-rdf-syntax-
    ns#Bag"/>
  <rdf:_1>Jack Robinson</rdf:_1>
  <rdf:_2>John Doe</rdf:_2>
  <rdf:_3>Richard Roe</rdf:_3>
</rdf:Description>
```

Slide 39

What is the “rdf:type” property?

- It specifies a class (there may be more than one) to which the resource belongs
- Its value is always a Web resource representing the class
- It can be expressed as a “type” attribute on a Description element
- It can also be implied by using a special element instead of a Description element

Slide 40

A Bag, the easy way

```
<rdf:Bag ID="committee">
  <rdf:li>Jack Robinson</rdf:li>
  <rdf:li>John Doe</rdf:li>
  <rdf:li>Richard Roe</rdf:li>
</rdf:Bag>
```

Using an “rdf:Bag” element means the value of “type” is
“http://www.w3.org/1999
/02/22-rdf-syntax-ns#Bag”

Slide 41

Kinds of “about” attributes

- “about”: specifies the URL of the Web resource directly
- “aboutEach”: specifies the URL of a container; the properties apply to the individual members of the container
- “aboutEachPrefix”: specifies an URL prefix; the properties apply to all Web resources with that prefix

Slide 42

Containers vs. multiple values

- A property can appear more than once with different values
- What is true of a container isn’t necessarily true of its contents and vice versa
- “aboutEach” lets us get to the contents when we already have a container
- “aboutEachPrefix” in effect manufactures a container based on URLs

Slide 43

Internationalization

- RDF property names are the same in all locales
- RDF string values take their language from the nearest enclosing “xml:lang” attribute
- Typical values of “xml:lang”: en-US, en-UK, ja-JP, he, i-no-nyn

Slide 44

Reified statements

- We reify statements so that we can talk about them rather than asserting them
- “Charles Dickens is the author of Bleak House” asserts a property of Charles Dickens
- “Jack believes that Charles Dickens is the author of War and Peace” asserts a property of Jack, not Charles Dickens

Slide 45

Reification properties

```
<rdf:Description about="...">
  <xx:creator>Charles Dickens</xx:creator>
</rdf:Description>
```

reifies as:

```
<rdf:Statement>
  <rdf:subject resource="..." />
  <rdf:predicate resource="...#creator/">
  <rdf:object>Charles Dickens</rdf:object>
</rdf:Statement>
```

Slide 46

Bags of reified statements

- A Description element can be seen as specifying a Bag of reified statements, one for each property element or attribute
- This Bag can be given a name using the “bagID” attribute
- Referring to a “bagID” attribute from an “aboutEach”-type Description element allows statements about statements

Slide 47

RDF Schemas

- Describe rules for using RDF properties
- Are expressed in RDF
- Are not to be confused with XML Schemas (eventual DTD replacements)
- Are still in Proposed Recommendation stage (therefore 90-95% stable)

Slide 48

RDF RDF Classes

- Are groups of Web resources
- Have URLs to identify them
- The special class “rdfs:Literal” consists of all possible RDF string values

Slide 49

RDF Property-centric classes

- In typical OO classes, each class specifies completely what properties it has and what their types are
- In RDF classes, each property specifies what classes of subjects and objects it relates
- Therefore, new properties can be added to a class without modifying the class

Slide 50

RDF Specifying classes

To specify a class, create an RDF resource of type rdfs:Class

```
<rdfs:Class id="MyClass">  
  <rdfs:label>My Class</rdfs:label>  
  <rdfs:comment>John Cowan's demonstration  
    Class</rdfs:comment>  
</rdfs:Class>
```

Slide 51

RDF Specifying properties

To specify a property, create an RDF resource of type rdfs:Property

```
<rdfs:Property id="myProperty">  
  <rdfs:comment>John Cowan's demo  
    property</rdfs:comment>  
  <rdfs:domain resource="#MyClass"/>  
  <rdfs:range resource="..#Literal"/>  
</rdfs:Property>
```

Slide 52

Schema URIs

- Ordinary XML namespace URIs are just to guarantee uniqueness: there is no assumption that the URI refers to anything useful (or even refers at all)
- URIs for namespaces used in RDF, though, should refer to an RDF schema document

Slide 53

Useful properties

- “rdf:type” relates any resource to its class
- “rdfs:subClassOf” relates a subclass to its superclass (multiple inheritance is OK)
- “rdfs:subPropertyOf” relates a subproperty to its superproperty

Slide 54

Useful properties

- “rdfs:seeAlso” relates a resource to another resource explaining it (use a subproperty to specify the nature of the explanation)
- “rdfs:isDefinedBy” is a subproperty of “rdfs:seeAlso” and relates a resource to its definition, typically an RDF schema

Slide 55

Useful properties

- “rdfs:domain” specifies the domain of a property (the classes of its subjects); if unknown, anything can be a subject
- “rdfs:range” specifies the range of a property (the single class of its objects); if unknown, anything can be an object

Slide 56

Useful properties

- “rdf:subject” is the property relating a reified statement to its subject (resource)
- “rdf:predicate” is the property relating a reified statement to its predicate (property)
- “rdf:object” is the property relating a reified statement to its object (value)

Slide 57

Useful properties

- “rdfs:label” specifies a human-readable name for this Class, Property, or whatever
- “rdfs:comment” specifies human-readable documentation
- Multiple values are useful for specifying multiple languages

Slide 58

Useful classes

- “rdfs:Resource” is the class of all resources
- “rdfs:Literal” is the class of all strings
- “rdfs:Class” is the class of all classes
- “rdfs:Property” is the class of all properties
- “rdf:Statement” is the class of all asserted RDF statements

Slide 59

Useful classes

- “rdfs:Container” is the superclass of all container classes
- “rdf:Bag”, “rdf:Seq”, “rdf:Alt” are the classes of Bags, Seqs, and Alts
- (Any other class that is a subclass of “rdfs:Container” can be used in RDF syntax in place of a standard container)

Slide 60

Dublin Core

- A set of fifteen basic properties for describing generalized Web resources
- The “obvious” mapping of Dublin Core properties into RDF properties has not yet been approved by the Dublin Core initiative, but is generally a good example

Slide 61

Dublin Core

- “Title”: the name given to the resource
- “Creator”: the person or organization primarily responsible for the resource
- “Subject”: what the resource is about
- “Description”: a description of the content

Slide 62

Dublin Core

- “Publisher”: the person or organization responsible for making the resource available
- “Contributor”: someone who has provided content to the resource other than the creator
- “Date”: date of creation or publication

Slide 63

Dublin Core

- “Type”: type of resource, such as home page, technical report, novel, photograph...
- “Format”: data format of the resource
- “Identifier”: URL, ISBN number, ...
- “Source”: another resource that this resource is derived from

Slide 64

Dublin Core

- “Language”: the language of the content
- “Relation”: another resource and its relationship to this one
- “Coverage”: the portion of time or space described by this resource (atlases, histories, etc.)

Slide 65

Dublin Core

- “Rights”: the intellectual property rights adhering to this resource, or a pointer to them

Slide 66

Where to look next

- RDF Syntax:
<http://www.w3.org/TR/1999/REC-rdf-syntax-19990222>
- RDF Schema:
<http://www.w3.org/TR/1999/PR-rdf-schema-19990303>
- Public comments archive:
<http://w3.org/Archives/Public/www-rdf-comments>

Slide 67

Example: RDF Schema in RDF-Schema

- Namespace-URL:
<http://www.w3.org/2000/01/rdf-schema#>

```
<rdfs:Class rdf:ID="Resource">
  <rdfs:label xml:lang="en">Resource</rdfs:label>
  <rdfs:label xml:lang="fr">Ressource</rdfs:label>
  <rdfs:comment>The most general class</rdfs:comment>
</rdfs:Class>
<rdfs:Class rdf:ID="Class">
  <rdfs:label xml:lang="en">Class</rdfs:label>
  <rdfs:label xml:lang="fr">Classe</rdfs:label>
  <rdfs:comment>The concept of Class</rdfs:comment>
  <rdfs:subClassOf rdf:resource="#Resource"/>
</rdfs:Class>
<rdf:Property ID="subClassOf">
  <rdfs:label xml:lang="en">subClassOf</rdfs:label>
  <rdfs:label xml:lang="fr">sousClasseDe</rdfs:label>
  <rdfs:comment>Indicates membership of a class</rdfs:comment>
  <rdfs:range rdf:resource="#Class"/>
  <rdfs:domain rdf:resource="#Class"/>
</rdf:Property>
```

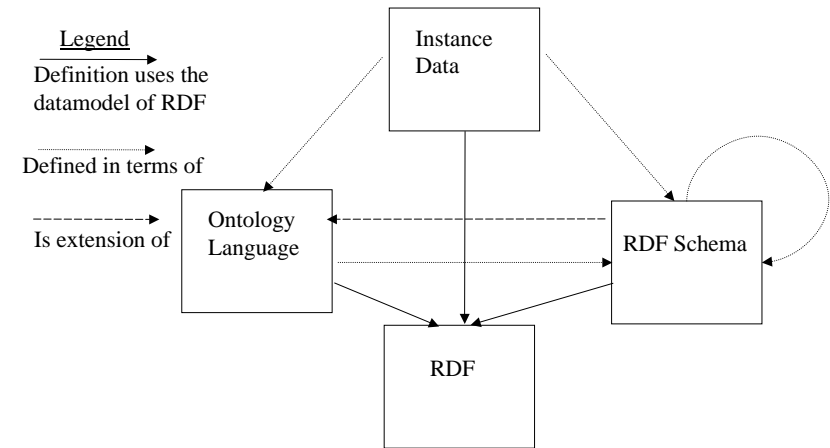
Slide 68

Extensibility of RDF

- Define an Ontology of your Language with RDF Schema (like RDF-Schema itself)
- Describe Instance Data using your new Vocabulary
- Advantage: all Languages use the same Data Model (simplifies Interoperability)

Slide 69

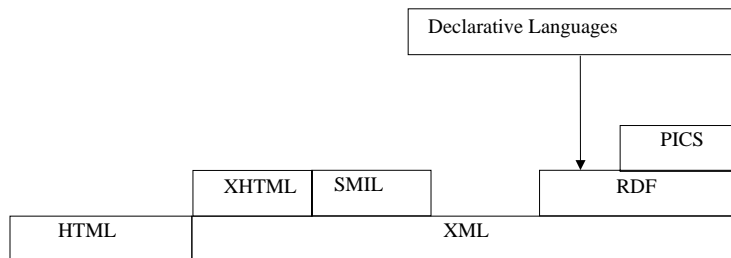
Ontology Languages on Top of RDF: The Principle



Slide 70

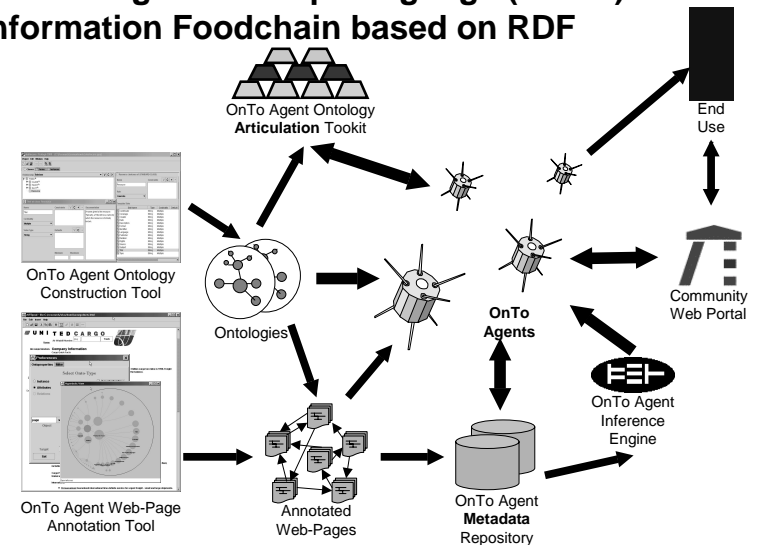
The Semantic Web

- A Web of machine understandable Data, based on declarative languages on top of RDF (all use the same datamodel!)
- Intelligent Agent enabling architecture
- W3C's vision for the Semantic Web Architecture:



Slide 71

DARPA Agent Markup Language (DAML): An Information Foodchain based on RDF



Slide 72

RDF-Resources

- RDF-Editor: Protege
www-smi.stanford.edu/projects/protege
- RDF-Parser and APIs/Query Engines
 - <http://www-db.stanford.edu/~melnik/rdf>
 - <http://www.aifb.uni-karlsruhe.de/~sde/rdf>
- RDF Knowledge Sources
 - DMOZ - Open Directory (largest human created Web-directory) <http://www.dmoz.org>
- General Information:
 - RDF Interest Mailing list: www-rdf-interest@w3.org
Archive: <http://lists.w3.org/Archives/Public/www-rdf-interest/>
 - [SemanticWeb.org](http://www.SemanticWeb.org)