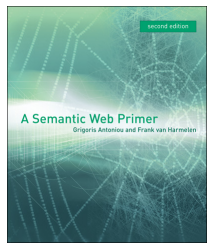
Semantic Web
2 - Metadata

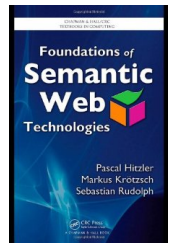
GEIST Research Group
http://geist.agh.edu.pl

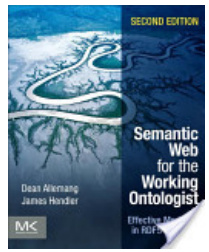
AGH University of Science and Technology, POLAND

Using slides according to license from:

- P. Hitzler – "Knowledge Representation for the Semantic Web" *course based on*
- P. Hitzler, M. Krötzsch, S. Rudolph – Foundations of Semantic Web Technologies
- e-Lite: 01LHVIU - Semantic Web: Technologies, Tools, Applications

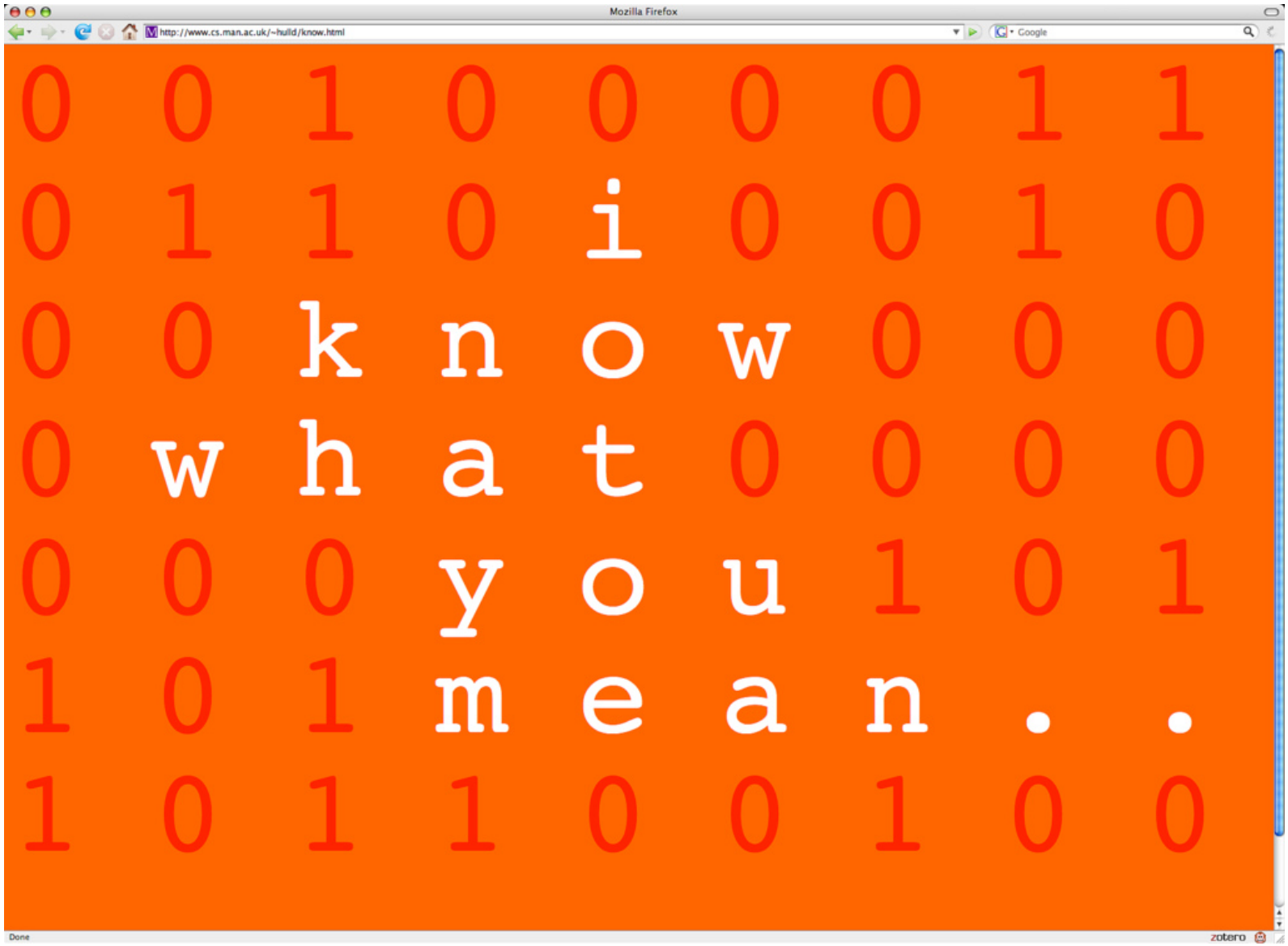




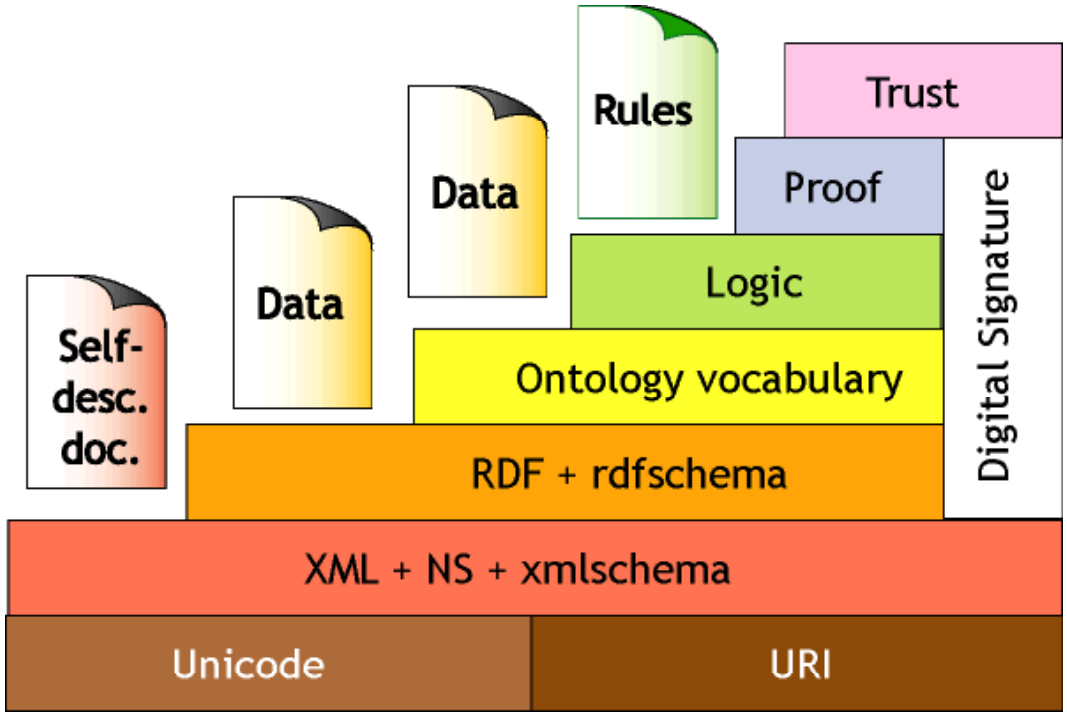


Outline

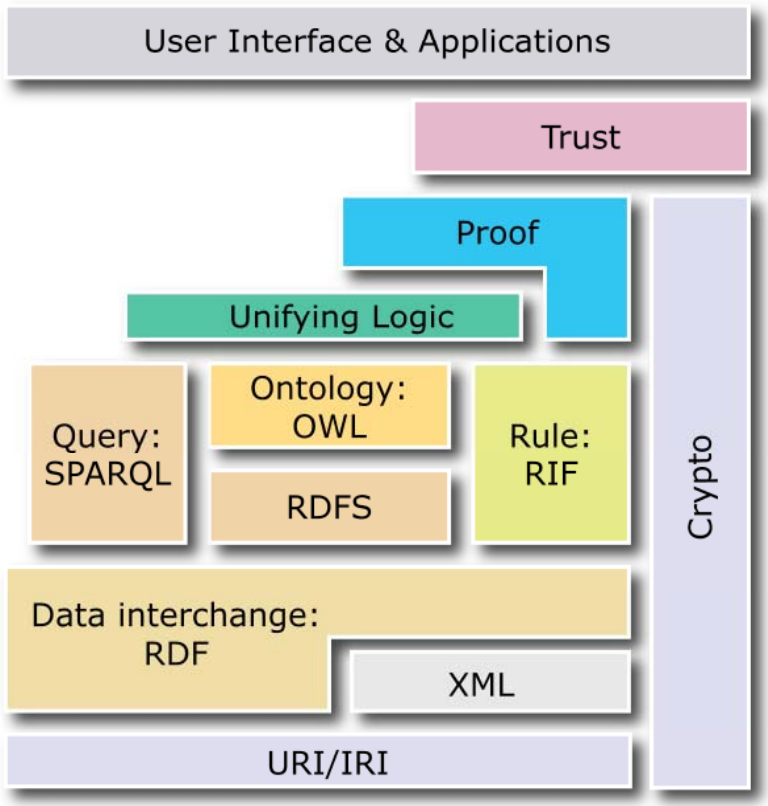
Outline



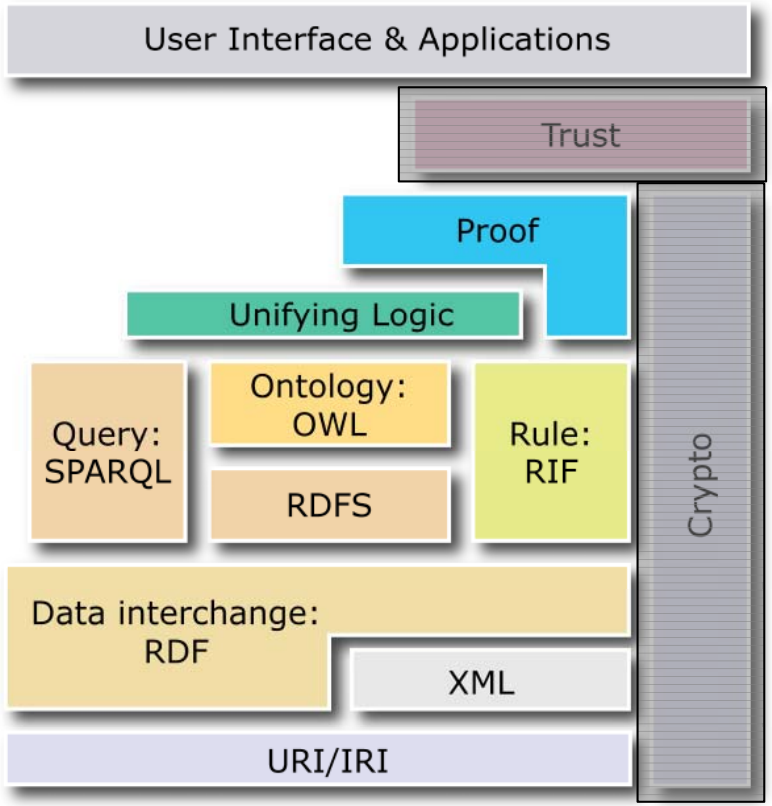
Technology stack (old: pre-2008)



Technology stack (current: 2008)

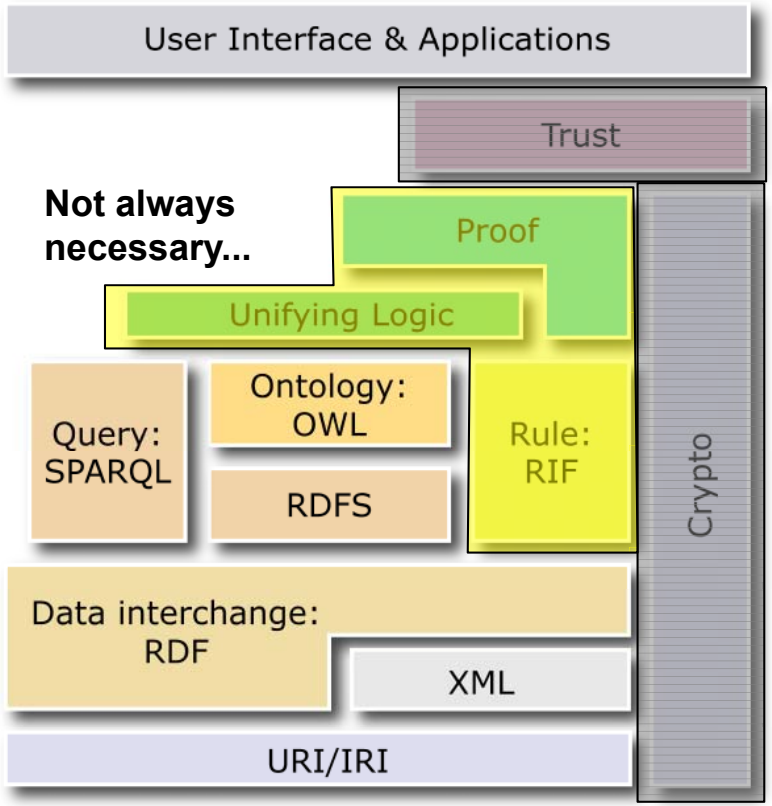


The real world



Not yet...!

The real world



Outline

Goal of the semantic Web

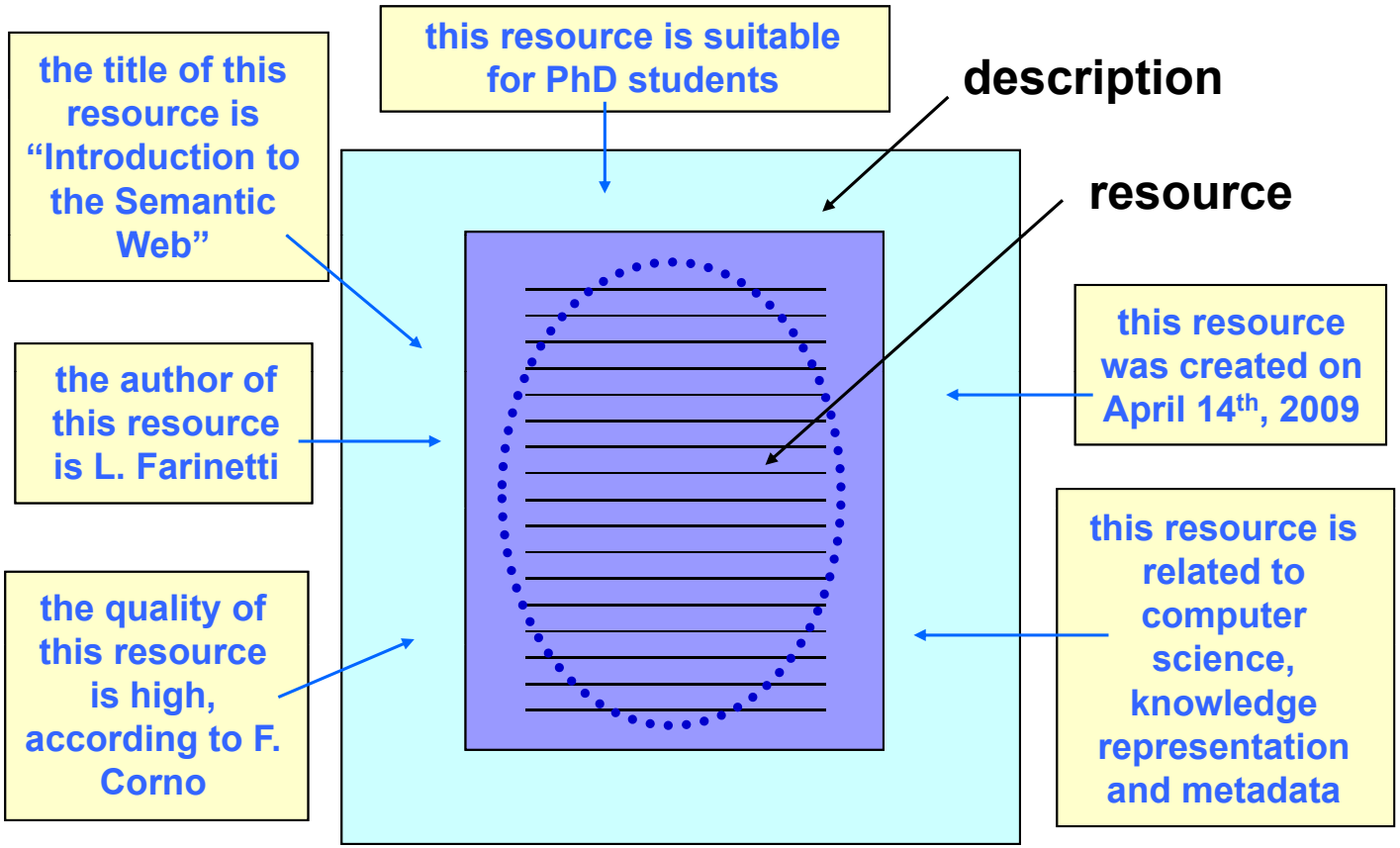
The Semantic Web will enable machines to COMPREHEND semantic documents and data, NOT human speech and writing

- Then, how???
  - Semantic Web foundation: metadata

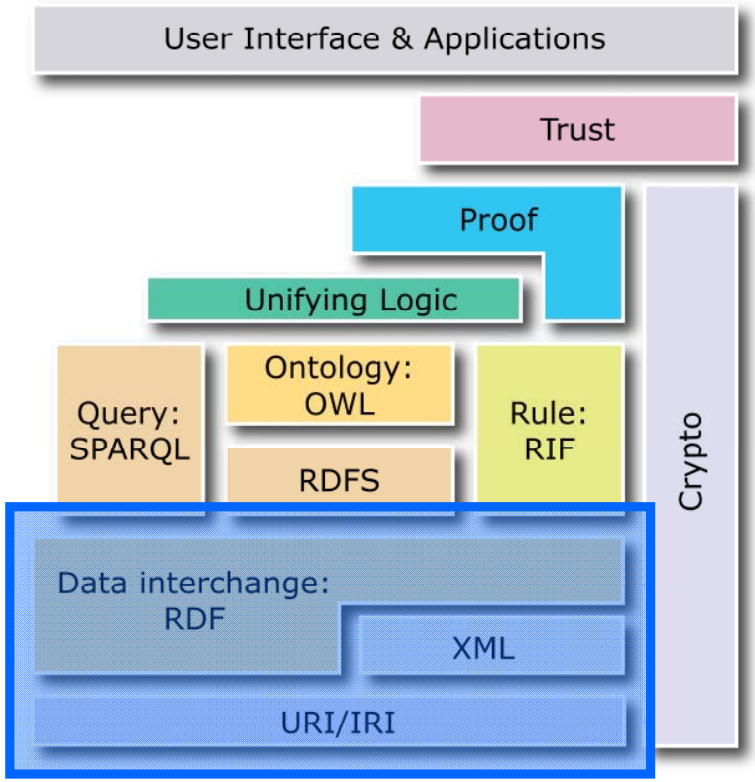
Resource and description

- Resource
  - Content, format, …
  - Access method dependent on format (I can read it if I "know" its language)
- Resource description
  - Independent of the format (I can read "people's comments" about the resource… provided that I know the language in which the comment is written)

Resource and description



Common language for describing resources



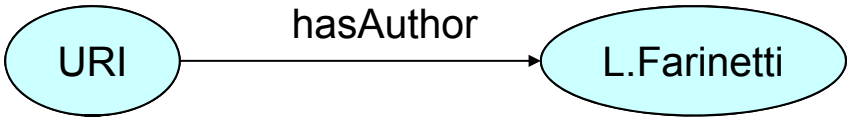
Common language for describing resources

- Resource Description Framework (RDF)
  - Resource = URI (retrievable, or not)
  - RDF is structured in statements
- A statement is a triple
  - Subject – predicate – object
  - Subject: a resource
  - Predicate: a verb / property / relationship
  - Object: a resource, or a literal string

Common language for describing resources

Author =
L. Farinetti

- Diagram:

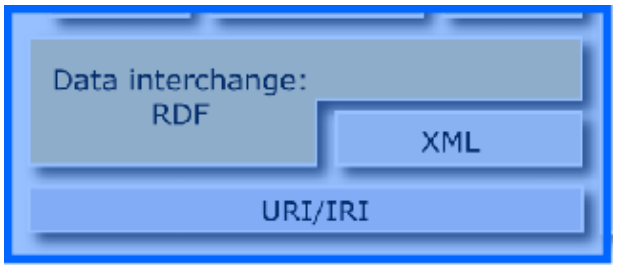


- Simple RDF assertion (triple):

```
triple (hasAuthor, URI, L.Farinetti)
```

Common language for describing resources

Author =
L. Farinetti



- RDF in XML syntax:

```
<RDF  xmlns="http://www.w3.org/TR/ … " >
  <Description about="http://www.polito.it/semweb/intro">
       <Author>L.Farinetti</Author>
  </Description>
</RDF>
```

Metadata standards examples

	MARC	Dublin Core	CDWA	VRA Core	CSDGM	Z39.87	LOM	DIG35	METS	JPX	SMPTE Metadata Dictionary
Standardization Body	Library of Congress	Dublin Core Metadata Initiative (DCMI)	Art Information Task Force (AITF)	Visual Resource Association	Federal Geographic Data Committee (FGDC)	National Information Standard Organization (NISO)	IEE (LTSC)	Digital Imaging Group (DIG of I3A)	Digital Library federation (DLF)	Joint Photographic Experts Group (JPEG)	Society of Motion Picture and Television Engineers (SMPTE)
Year	Current version MARC 21 since 1999	Current version 1.1 since 1999	Current version 2.0 since 2000	Current version 3.0 since 2002	Update version since from 1998	2002	2002	Current version 1.1 April 2001	Last review 2001	2000	Last review 2004
MM Type	Any	Any	Any	Images	Any	Images	Any	Images	Any	Images	Any
Domain	Bibliographic media description	Bibliographic media description	Description of Art works	Description of images of Art works	Description of Geographic media	Description of still images	Description of educational media	Description of digital images	Description of digital objects	Description of digital images	Description of audio/video documents
Level	Largely semantic	Largely semantic	Largely semantic	Largely semantic	Semantic and technical	Technical	Largely semantic	Semantic and technical	Semantic and technical	Semantic and technical	Semantic and technical
Producibility	Mainly manual	Mainly manual	Mainly manual	Mainly manual	Manual and Automatic	Mainly automatic	Mainly manual	Mainly manual	Mainly manual	Mainly manual	Manual and Automatic

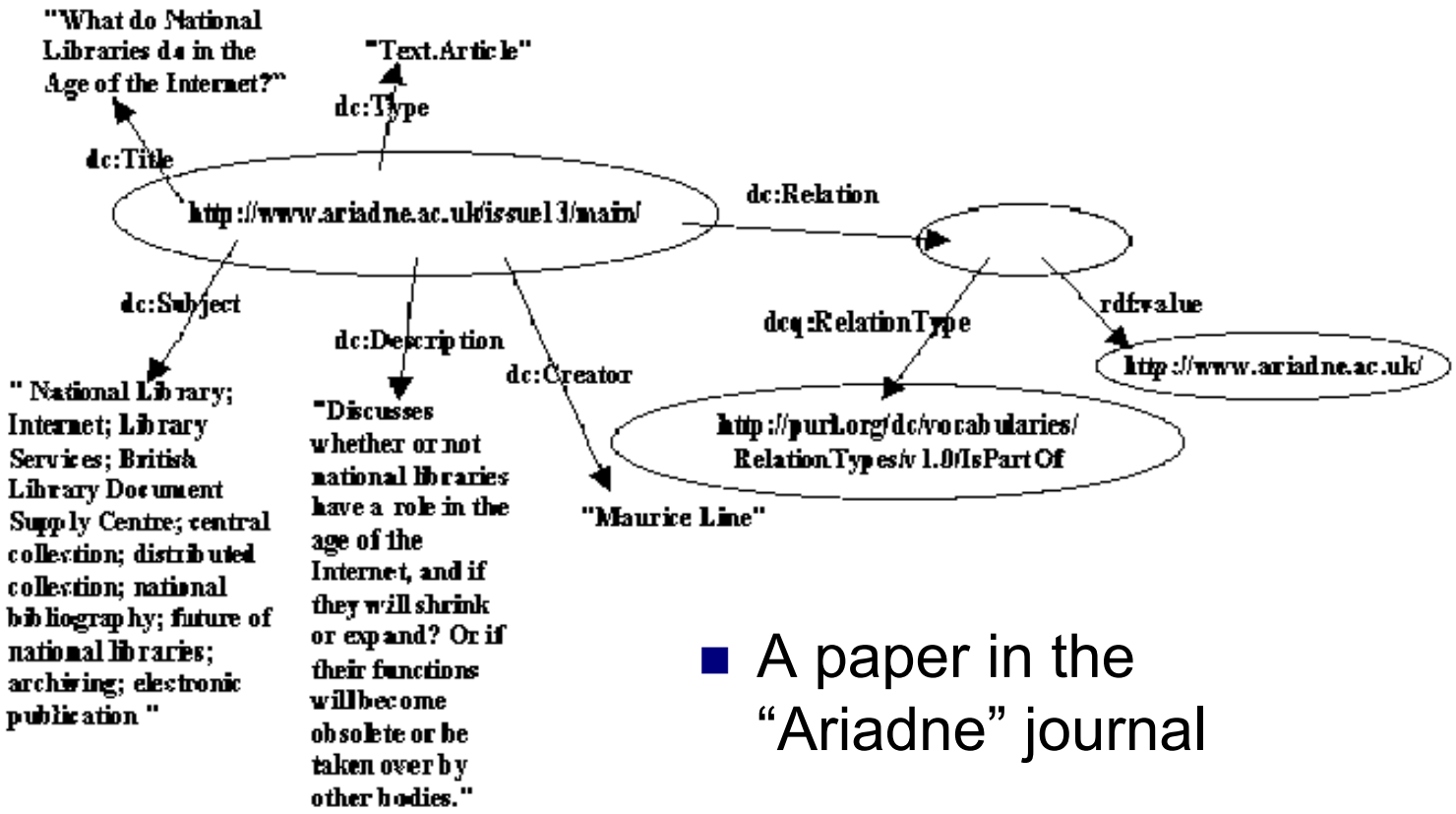
Dublin Core

- Dublin Core Metadata Element Set (DCMES)
  - Building blocks to define metadata for the Semantic Web
  - 15 elements, or categories, general enough to describe most of the published resources
  - Extra elements and element refinements

DC metadata element set

accrualMethod	creator	language	source
accrualPeriodicity	date available created dateAccepted dateCopyrighted dateSubmitted issued modified valid	provenance	subject
accrualPolicy	description abstract tableOfContents	publisher	title alternative
audience educationLevel mediator	format extent medium	relation conformsTo hasFormat hasPart hasVersion isFormatOf isPartOf isReferencedBy isReplacedBy isRequiredBy isVersionOf references replaces requires	type
contributor	identifier bibliographicCitation	rights accessRights license	
coverage spatial temporal	instructionalMethod	rightsHolder	

Example of description using Dublin Core (in RDF)



- A paper in the "Ariadne" journal

Common language for field values

- Problems
  - Value type

Date = 2009-04-14

type = date

Title = "Introduction to the Semantic Web"

type = string

Author = L. Farinetti

type = string
"standard" format?
Laura Farinetti, Farinetti Laura, Farinetti L., …

Common language for field values

- Problems
  - Value type
  - Value restrictions?
    - freedom vs shared understanding

Level = PhD students

any value?
list of possible values?

Quality = high

High, medium, low?
1 to 5?
any value?

Topic = {computer science, knowledge representation, metadata}

any value?
any number of values?

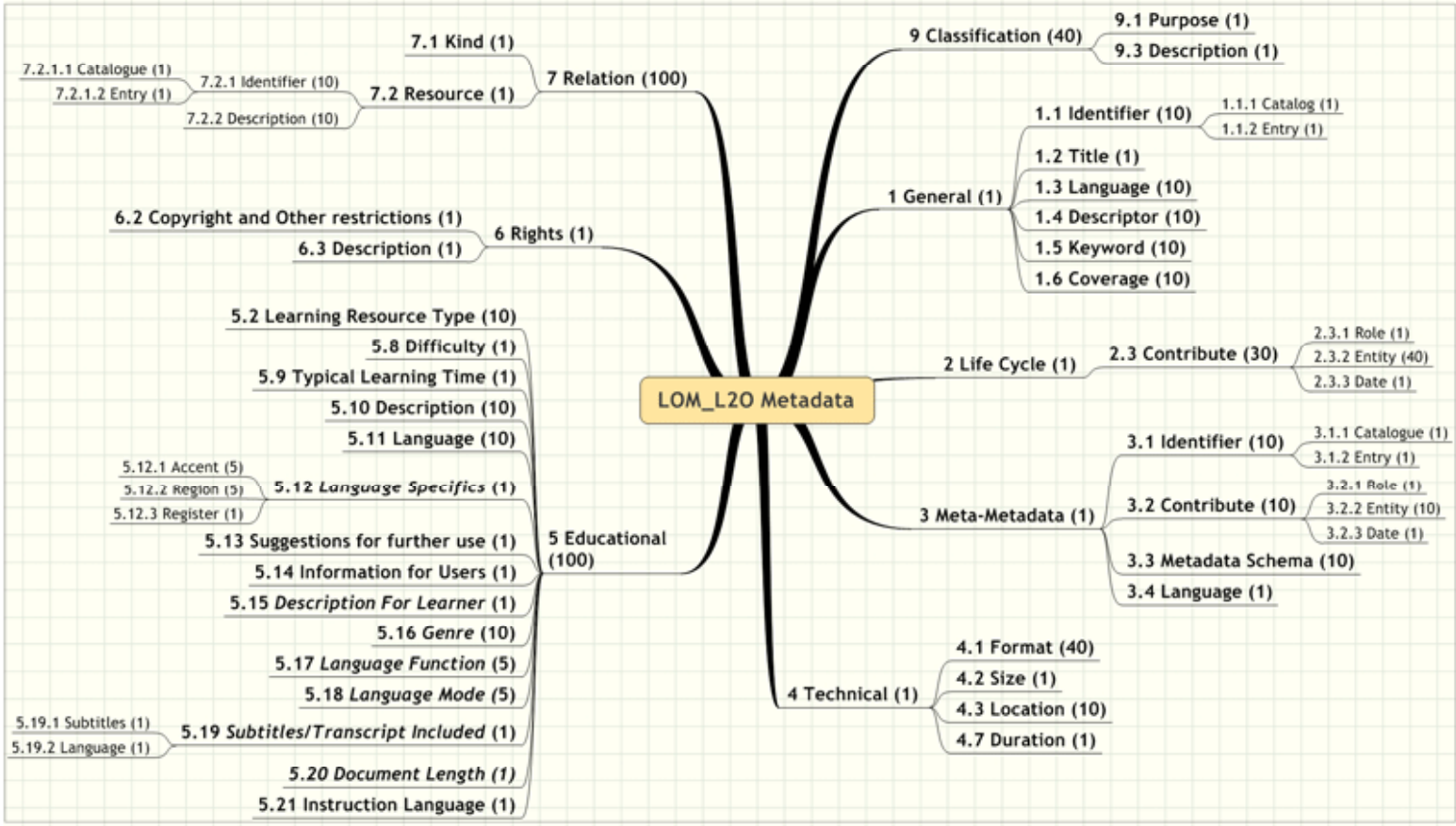
Common language for field values

- Solution: metadata standards + controlled vocabularies
- Metadata standards
  - Only some, and partially
- Controlled vocabularies
  - Explicit list of possible values

Examples from IEEE LOM

- 1484.12.1 - 2002 Learning Object Metadata (LOM) Standard
  - Developed by the IEEE Learning Technology Standards Committee (LTSC)
- Standard to describe the “Learning Objects” in order to guarantee their interoperability

Examples from IEEE LOM



Examples from IEEE LOM

Nr	Name	Explanation	Size	Order	Value space	Datatype	Example
1.2	Title	Name given to this learning object.	1	unspecified	-	LangString (smallest permitted maximum: 1000 char)	("en", "The life and works of Leonardo da Vinci")

Nr	Name	Explanation	Size	Order	Value space	Datatype	Example
1.3	Language	The primary human language or languages used within this learning object to communicate to the intended user. NOTE 1:--An indexation or cataloging tool may provide a useful default. NOTE 2:--If the learning object had no lingual content (as in the case of a picture of the Mona Lisa, for example), then the appropriate value for this data element would be "none". NOTE 3:--This data element concerns the language of the learning object. Data element 3.4:Meta-Metadata.Language concerns the language of the metadata instance.	smallest permitted maximum: 10 items	unordered	LanguageID = Langcode ("-"Subcode)* with Langcode a language code as defined by the code set ISO 639:1988 and Subcode (which can occur an arbitrary number of times) a country code from the code set ISO 3166-1:1997.	CharacterString (smallest permitted maximum: 100 char)	"en", "en-GB", "de", "fr-CA", "it" "grc" (ancient greek, until 1453) "en-US-philadelphia" "eng-GB-cockney" "map-PG-buin" (Austronesian –Papua New Guinea – buin) "gem-US-pennsylvania"

Examples from IEEE LOM

Nr	Name	Explanation	Size	Order	Value space	Datatype	Example
2.3.1	Role	Kind of contribution. NOTE 1:--Minimally, the Author(s) of the learning object should be described.	1	unspecified	author publisher unknown initiator terminator validator editor graphical designer technical implementer content provider technical validator educational validator script writer instructional designer subject matter expert NOTE 2:--"terminator" is the entity that made the learning object unavailable.	Vocabulary (State)	-
2.3.2	Entity	The identification of and information about entities (i.e., people, organizations) contributing to this learning object. The entities shall be ordered as most relevant first.	smallest permitted maximum: 40 items	ordered	vCard, as defined by IMC vCard 3.0 (RFC 2425, RFC 2426).	CharacterString (smallest permitted maximum: 1000 char)	"BEGIN:VCARD\nFN:Joe Friday\nTEL:+1-919-555-7878\nTITLE:Area Administrator\, Assistant\n EMAIL\;TYPE=INTERN\nET:jfriday@host.com\nEND:VCARD\n"
2.3.3	Date	The date of the contribution.	1	unspecified	-	DateTime	"2001-08-23"

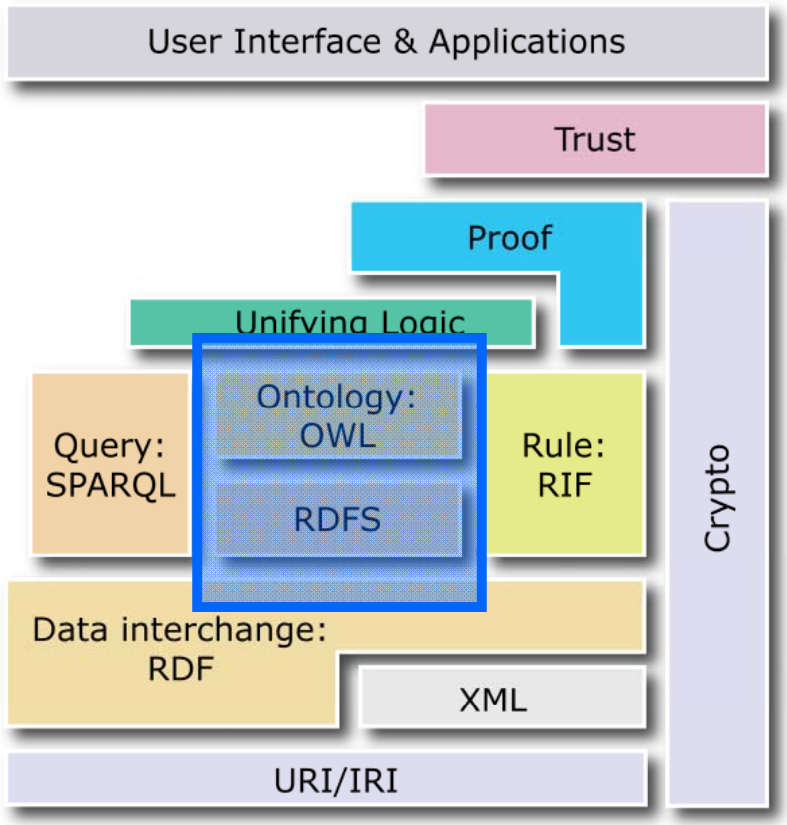
… + controlled vocabularies

Nr	Name	Explanation	Size	Order	Value space	Datatype	Example
1.5	Keyword	A keyword or phrase describing the topic of this learning object. This data element should not be used for characteristics that can be described by other data elements.	smallest permitted maximum: 10 items	unordered	-	LangString (smallest permitted maximum: 1000 char)	("en", "Mona Lisa")

- A closed list of named subjects, which can be used for classification
- Metadata field values are restricted to a list of terms (selected by experts)

Topic = {computer science, ~~informatics~~, knowledge representation, metadata}

Semantically rich descriptions to support search



Outline

Outline

A common language for describing resources

- The Resource Description Framework (RDF) is a language for **representing information about resources** in the World Wide Web
- Particularly intended for representing **metadata** about Web resources
- RDF can also be used to represent information about things that can be **identified** on the Web, even when they cannot be directly **retrieved** on the Web

RDF Design goals

- having a **simple data model**
- having **formal semantics** and **provable inference**
- using an extensible **URI-based vocabulary**
- using an XML-based syntax
- supporting use of XML schema **datatypes**
- allowing **anyone** to make statements about **any** resource

Simple yet powerful

- RDF has an **abstract syntax** that reflects a simple graph-based data model
- RDF has **formal semantics** with a rigorously defined notion of entailment providing a basis for well founded deductions

Basic principles (1/2)

- Clearly separate
  - **Model** structure (RDF graph)
  - Interpretation **Semantics** (Entailment)
  - Concrete **Syntaxes** (XML, TN, N3, …)
- Only two datatypes
  - URI/URIref: everything is a URI
  - Literal
    - String or other XSD datatype

Basic principles (2/2)

- Integrated with the Web
  - Uses XMLSchema datatypes
  - May reference http-retrievable resources
- **Open world** assumption
  - Allows anyone to make statements about any resource
  - No guaranteed completeness
  - No guaranteed consistency

Outline

Outline

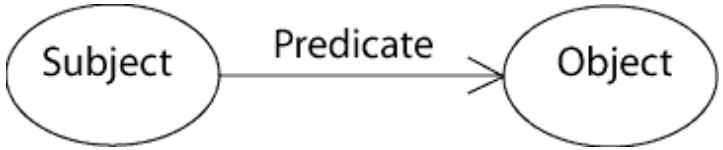
- RDF Design objectives
- **RDF General structure**
- RDF Vocabularies
- Serialization: XML
- Semantic features
- RDF Schema

Key concepts

- Graph data model
- URI-based vocabulary
- Datatypes
- Literals
- XML serialization syntax
- Expression of simple facts
- Entailment

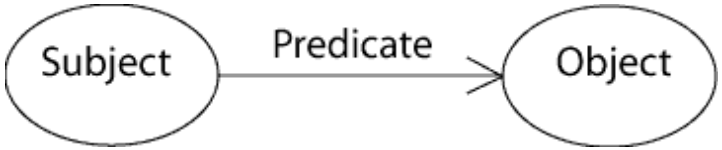
Graph data model

- Triple: subject, predicate, object
- Expression: collection of triples
  - RDF graph



Terminology and constraints

- *Subject* and *Object* are called *Nodes*
- *Predicate* and *Property* are synonyms
- Special unnamed nodes: Blank Nodes
- Subject may be: URI reference or blank node
- Predicate must be: URI reference
- Object may be: URI reference, literal or blank node



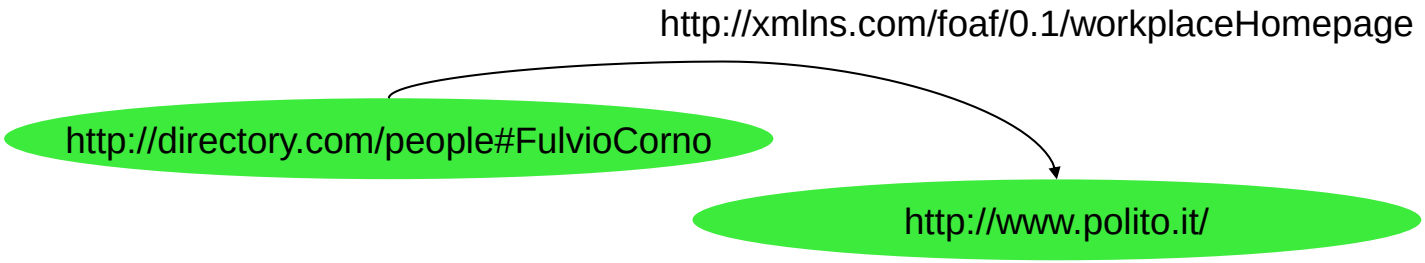
The Triples and the Graph

- The *assertion of an RDF triple* says that some relationship, indicated by the predicate, holds between the things denoted by subject and object of the triple.
- The *assertion of an RDF graph* amounts to asserting all the triples in it, so the meaning of an RDF graph is the conjunction (logical AND) of the statements corresponding to all the triples it contains.

Expression of Simple Facts

- Some simple facts indicate a relationship between two things → one triple
  - the predicate names the relationship
  - the subject and object denote the two things

Information in triples



RDF

CompanyHomePage

PersonID	Homepage
FulvioCorno	http://www.polito.it/

Relational database

First order logic predicate

HasCompanyHomePage(
'FulvioCorno',
'http://www.polito.it/') ;

But...

- Relational database tables may have an arbitrary number of columns
- First order logic predicates may have an arbitrary number of places (arguments)
- RDF triples may only have **one** subject and **one** object
  - Complex statements have to be decomposed for representation as RDF triples

Example

- Represent in RDF the following statement
- "there is a Person identified by http://www.w3.org/People/EM/contact#me, whose name is Eric Miller, whose email address is em@w3.org, and whose title is Dr."

Example

URIs represent (almost) everything

- Nodes (subject or object)
  - individuals: Eric Miller, identified by `http://www.w3.org/People/EM/contact#me`
  - kinds of things: Person, identified by `http://www.w3.org/2000/10/swap/pim/contact#Person`
  - values of properties: `mailto:em@w3.org` as the value of the mailbox property
- Predicates
  - properties of things: mailbox, identified by `http://www.w3.org/2000/10/swap/pim/contact#mailbox`

Non-URI information

- Literals (only as objects, never as subjects)
  - The name `"Eric Miller"`
  - The title `"Dr."`
  - May be localized
    - `"Dr."@en`
    - `"Dott."@it`
  - May be typed with XMLSchema data types
    - `"27"^^<http://www.w3.org/2001/XMLSchema#integer>`
    - `"37"^^xsd:integer`
    - `"1999-08-16"^^xsd:date`

URIs are more than URLs

- URL = uniform resource *locator*
  - Designed to locate, and retrieve, resources on the web
- URI = uniform resource *identifier*
  - More general
  - Identifies also resources that do not have a network location
  - Every person or organization can independently create URIs, and use them to identify “things” (either concrete or abstract)

URIref = URI#fragmet

- URIref = URI reference
- A single URI may define many different resources
  - E.g., the URI references an RDF file with many definitions
- To identify a single **fragment** inside the URI, we use the '#' notation
  - E.g., http://example.org/index#person

“Triple” or “Turtle” notation

```
<http://www.w3.org/People/EM/contact#me>
<http://www.w3.org/2000/10/swap/pim/contact#fullName>
"Eric Miller" .

<http://www.w3.org/People/EM/contact#me>
<http://www.w3.org/2000/10/swap/pim/contact#mailbox>
<mailto:em@w3.org> .

<http://www.w3.org/People/EM/contact#me>
<http://www.w3.org/2000/10/swap/pim/contact#personalTitle>
"Dr." .

<http://www.w3.org/People/EM/contact#me>
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://www.w3.org/2000/10/swap/pim/contact#Person> .
```

“Triple” or “Turtle” notation (abbreviated)

```
w3people:EM#me contact:fullName "Eric Miller" .

w3people:EM#me contact:mailbox <mailto:em@w3.org> .

w3people:EM#me contact:personalTitle "Dr." .

w3people:EM#me rdf:type contact:Person .
```

More details on the turtle syntax and further abbreviations will be shown in the SPARQL chapter

Example

```
@prefix rdf: http://www.w3.org/1999/02/22-rdf-syntaxns# .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix : <http://example.org/#> .

<http://www.w3.org/TR/rdf-syntax-grammar>
  dc:title "RDF/XML Syntax Specification (Revised)" ;
  :editor [
    :fullName "Dave Beckett";
    :homePage <http://purl.org/net/dajobe/>
  ] .
```

RDF/XML Syntax

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
             xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#">

  <contact:Person rdf:about="http://www.w3.org/People/EM/contact#me">
    <contact:fullName>Eric Miller</contact:fullName>
    <contact:mailbox rdf:resource="mailto:em@w3.org"/>
    <contact:personalTitle>Dr.</contact:personalTitle>
  </contact:Person>

</rdf:RDF>
```

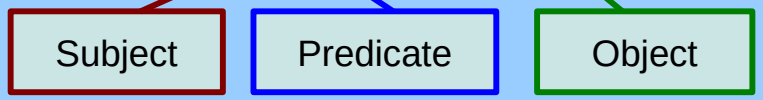
RDF/XML Syntax

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#">

  <contact:Person rdf:about="http://www.w3.org/People/EM/contact#me">
    <contact:fullName>Eric Miller</contact:fullName>
    <contact:mailbox rdf:resource="mailto:em@w3.org"/>
    <contact:personalTitle>Dr.</contact:personalTitle>
  </contact:Person>

</rdf:RDF>
```

Name space shortcut.
Equivalent to
`http://www.w3.org/2000/10/swap/pim/contact#fullName`

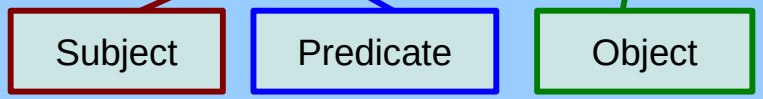
RDF/XML Syntax

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#">

  <contact:Person rdf:about="http://www.w3.org/People/EM/contact#me">
    <contact:fullName>Eric Miller</contact:fullName>
    <contact:mailbox rdf:resource="mailto:em@w3.org"/>
    <contact:personalTitle>Dr.</contact:personalTitle>
  </contact:Person>

</rdf:RDF>
```



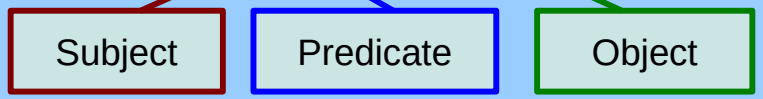
RDF/XML Syntax

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#">

  <contact:Person rdf:about="http://www.w3.org/People/EM/contact#me">
    <contact:fullName>Eric Miller</contact:fullName>
    <contact:mailbox rdf:resource="mailto:em@w3.org"/>
    <contact:personalTitle>Dr.</contact:personalTitle>
  </contact:Person>

</rdf:RDF>
```



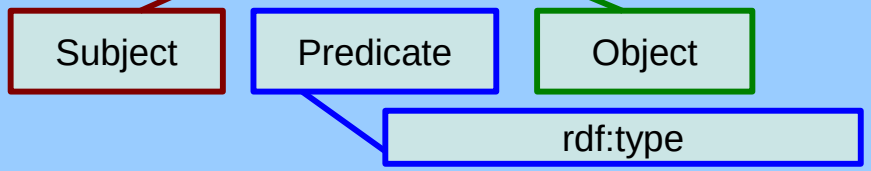
RDF/XML Syntax

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#">

  <contact:Person rdf:about="http://www.w3.org/People/EM/contact#me">
    <contact:fullName>Eric Miller</contact:fullName>
    <contact:mailbox rdf:resource="mailto:em@w3.org"/>
    <contact:personalTitle>Dr.</contact:personalTitle>
  </contact:Person>

</rdf:RDF>
```



RDF/XML Syntax

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#">

  <contact:Person rdf:about="http://www.w3.org/People/EM/contact#me">
    <contact:fullName>Eric Miller</contact:fullName>
    <contact:mailbox rdf:resource="mailto:em@w3.org"/>
    <contact:personalTitle>Dr.</contact:personalTitle>
  </contact:Person>

</rdf:RDF>
```



Outline

RDF vocabularies

- A set of URIref is called *vocabulary*
- Common vocabularies collect URIrefs under the same *name space*, so that all nodes may be reached with QNames such as:
  - prefix:nodeName
- The name space is chosen to represent the organization responsible for the definitions
- Every elaboration in RDF must first *resolve all prefixes*, so that only **absolute URIs** are used by the algorithms

Common prefixes

- prefix rdf:, namespace URI: http://www.w3.org/1999/02/22-rdf-syntax-ns#
- prefix rdfs:, namespace URI: http://www.w3.org/2000/01/rdf-schema#
- prefix dc:, namespace URI: http://purl.org/dc/elements/1.1/
- prefix owl:, namespace URI: http://www.w3.org/2002/07/owl#
- prefix xsd:, namespace URI: http://www.w3.org/2001/XMLSchema#
- prefix ex:, namespace URI: http://www.example.org/ (or http://www.example.com/)

Vocabulary reuse

- Extremely easy to re-use other vocabularies in our RDF graph... just define a prefix to point to the proper name space
- When using a predicate, **always** check if its semantics is already satisfied by some property defined in well-known vocabularies
  - Never re-define, with a different URIref, some already existing predicate
- The same applies for names, but with somewhat less importance.

Hands-on: let's explore some useful vocabularies...

- Dublin Core
  - Specification: http://dublincore.org/documents/dces/
  - Namespace: xmlns:dc="http://purl.org/dc/elements/1.1/"
- FOAF
  - Specification: http://xmlns.com/foaf/spec/
  - Namespace: xmlns:foaf="http://xmlns.com/foaf/0.1/"

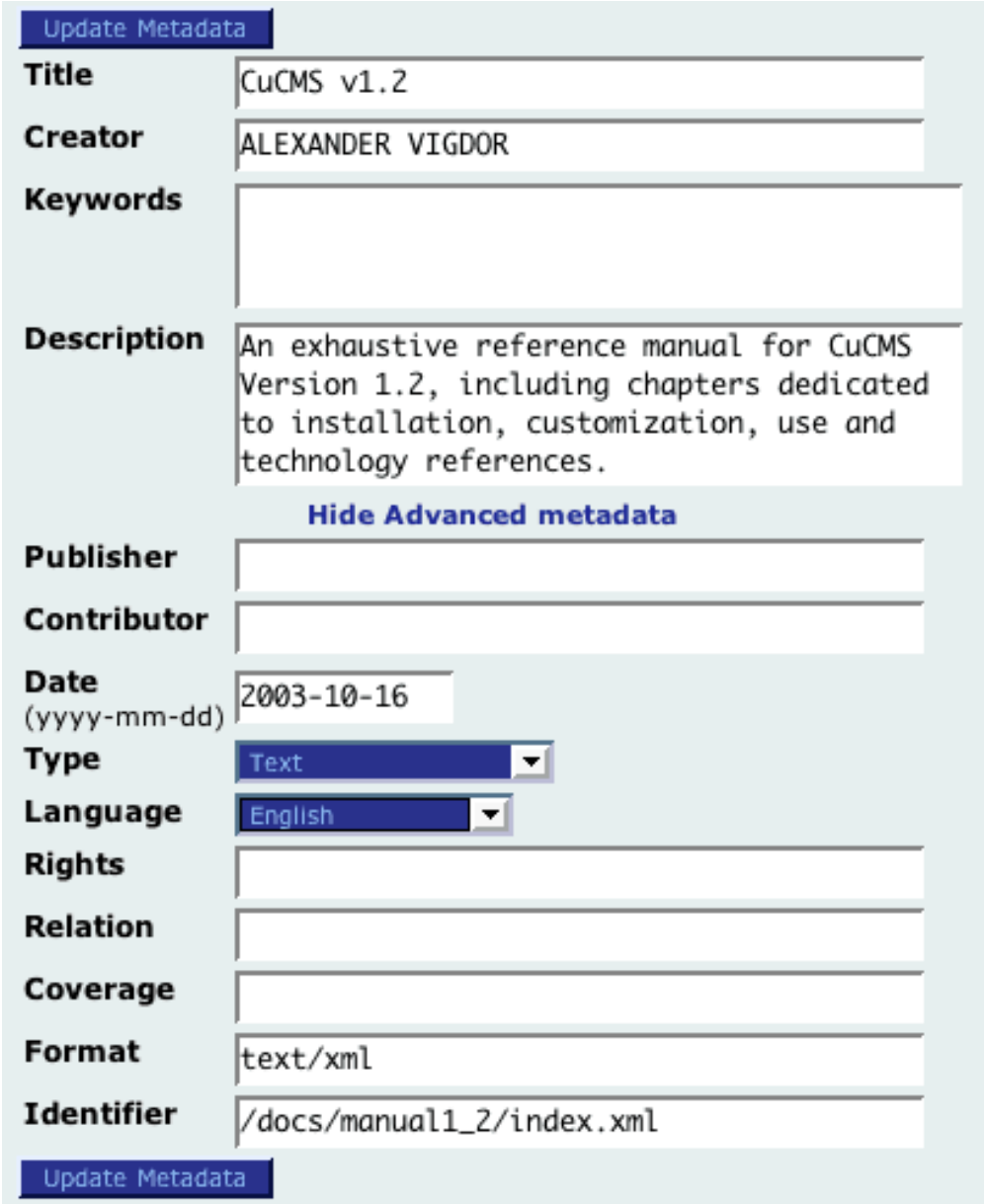
Hands-on: let's explore some useful vocabularies...

- Recent Dublin Core enhancement: DCMI Metadata Terms
  - Specification: http://dublincore.org/documents/dcmi-terms/
  - Namespace: xmlns:dcterms="http://purl.org/dc/terms/"
- RSS 1.0
  - Information: http://en.wikipedia.org/wiki/RSS_(file_format)

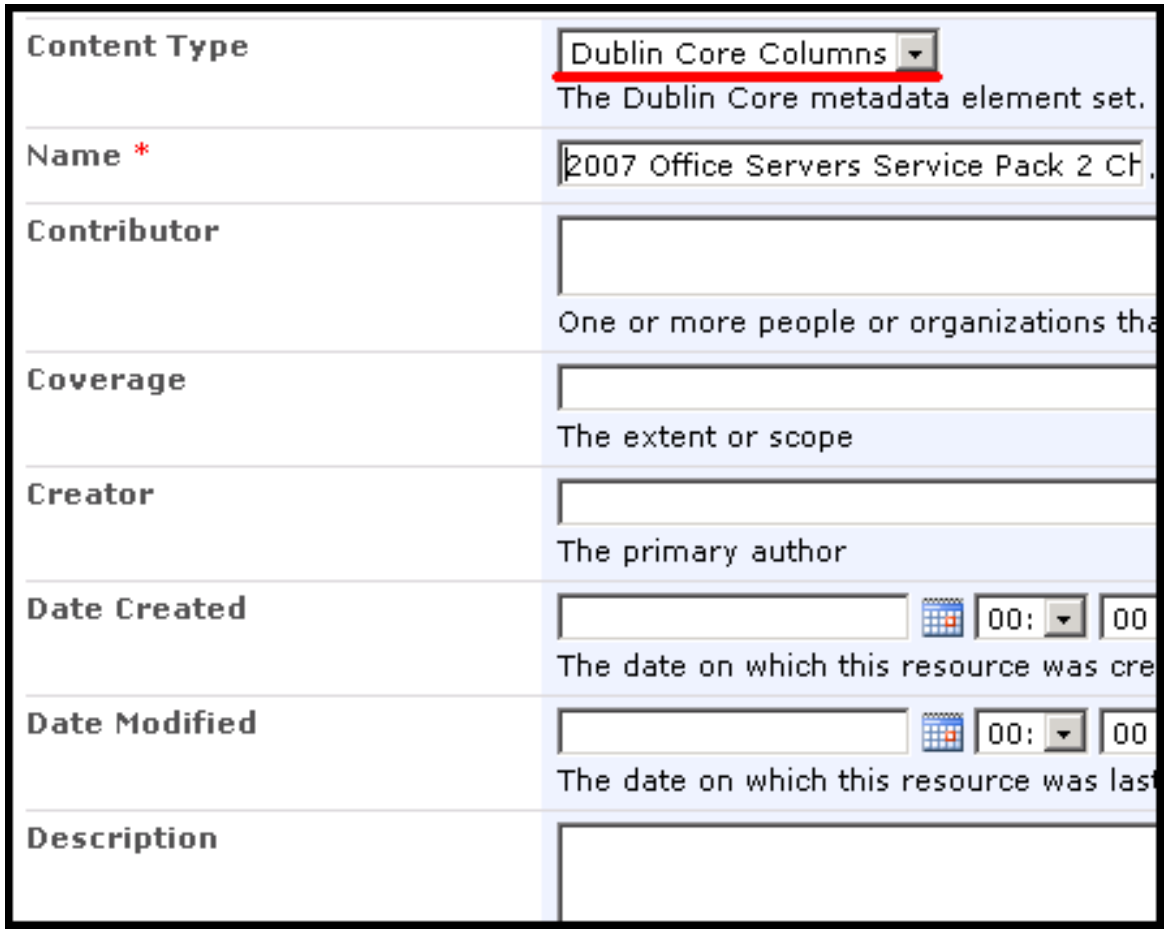
Dublin Core example

```
<?xml version="1.0"?>

<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:dc= "http://purl.org/dc/elements/1.1/">

<rdf:Description rdf:about="http://www.w3schools.com">
  <dc:description>
    W3Schools - Free tutorials
  </dc:description>
  <dc:publisher>Refsnes Data as</dc:publisher>
  <dc:date>2008-09-01</dc:date>
  <dc:type>Web Development</dc:type>
  <dc:format>text/html</dc:format>
  <dc:language>en</dc:language>
</rdf:Description>

</rdf:RDF>
```

DC: HyperContent system



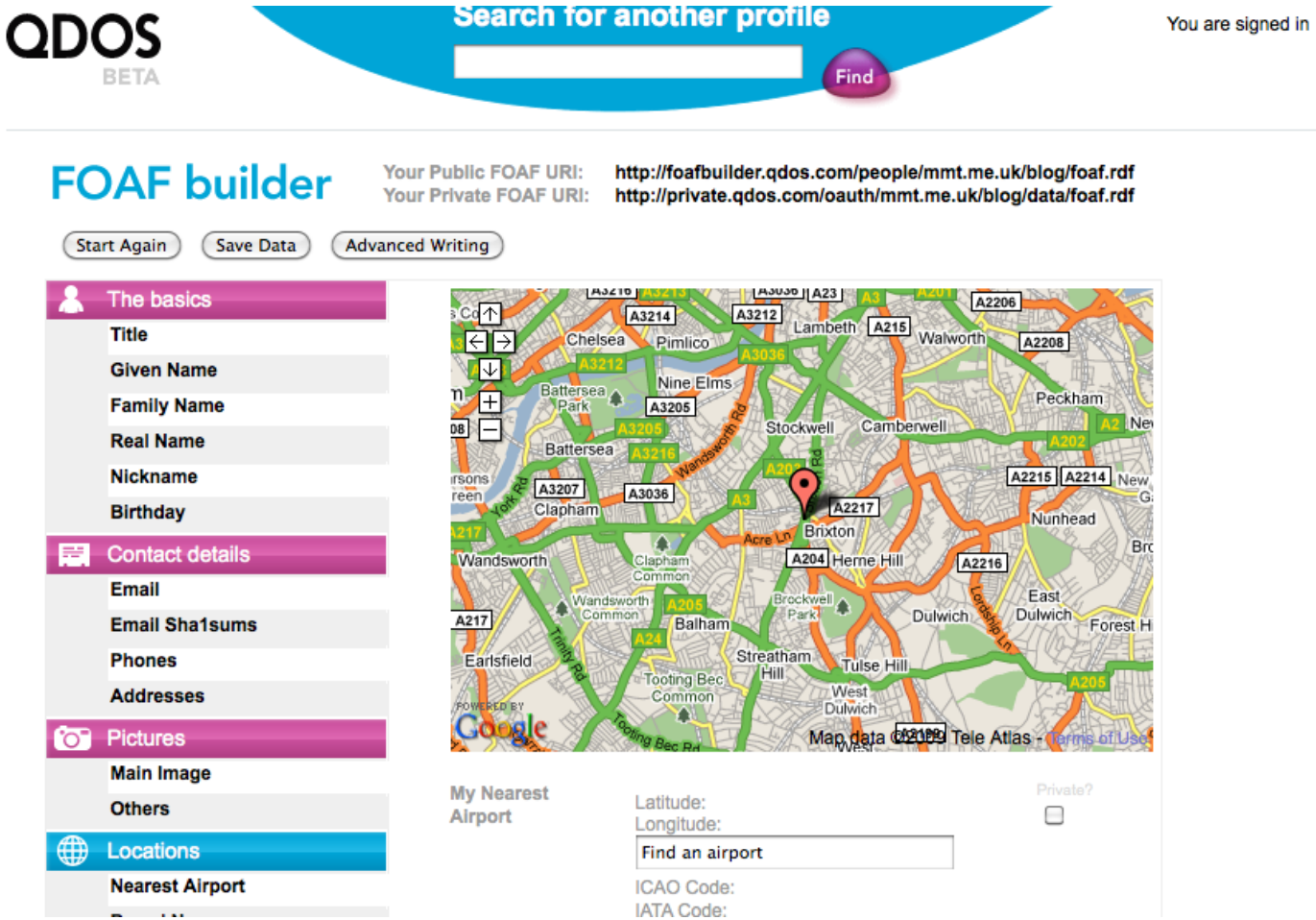
DC: MS SharePoint 2007



FOAF example

```
<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:foaf="http://xmlns.com/foaf/0.1/"

 <foaf:name>Weronika T. Adrian (Furmanska)</foaf:name>
 <foaf:givenname>Weronika T.</foaf:givenname>
 <foaf:family_name>Adrian</foaf:family_name>
 <foaf:nick>wta</foaf:nick>
 <foaf:homepage rdf:resource="http://home.agh.edu.pl/wta"/>
 <foaf:workplaceHomepage
       rdf:resource="http://geist.agh.edu.pl"/>
 <foaf:knows>
   <foaf:Person>
     <foaf:name>Krzysztof Kluza</foaf:name>
     <rdfs:seeAlso rdf:resource="http://home.agh.edu.pl/
                                 ~kluza/kklFoaf.rdf"/>
   </foaf:Person>
 </foaf:knows>
</rdf:RDF>
```

FOAF: Builders



FOAF: Viewers

FOAFer

Please enter a FOAF-Resource: http:// go

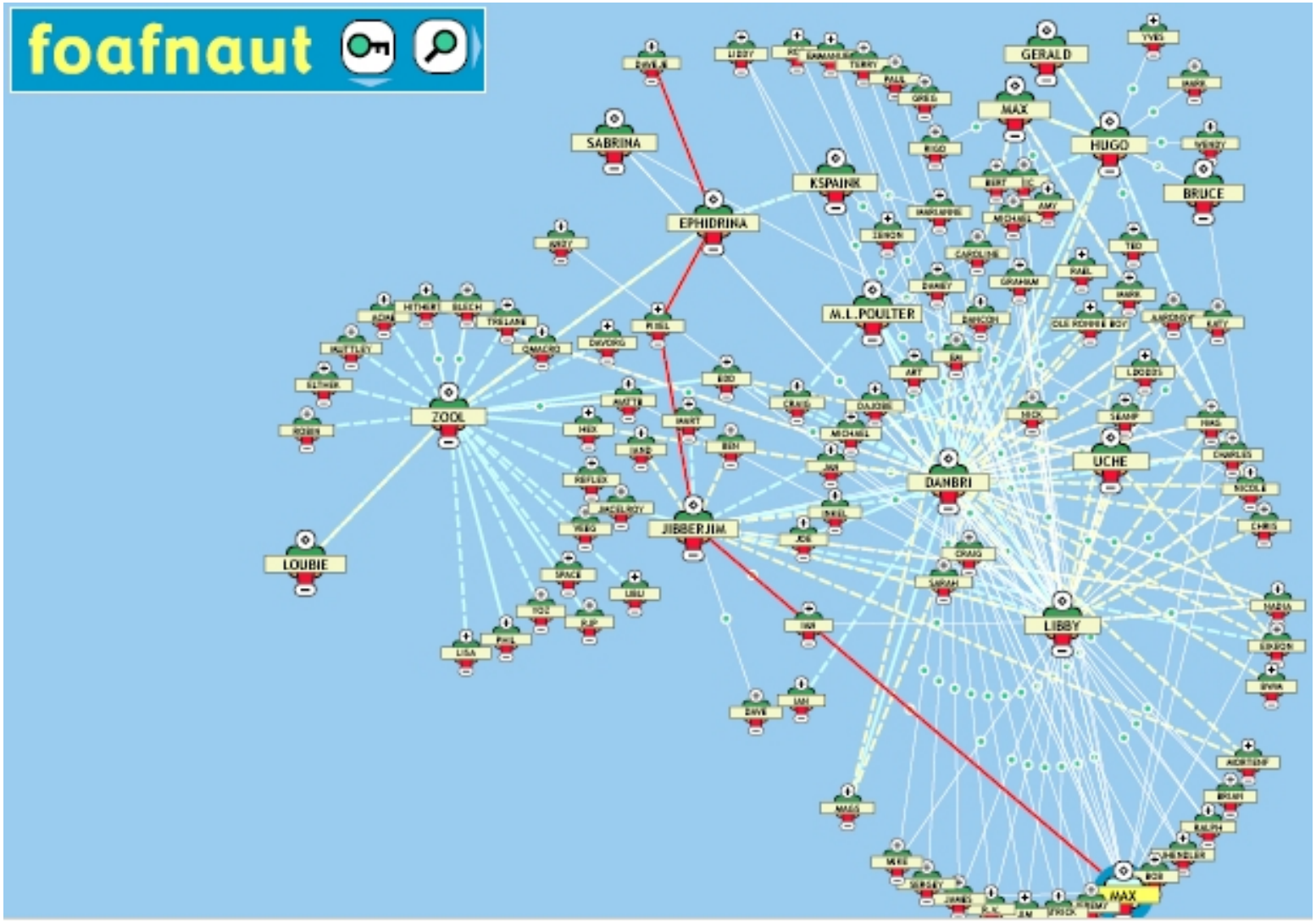
Make your own FOAF-file!
What is FOAF?

-> http://www.infopoint.pl/wordpress/foaf.rdf#me

Name:	Artur Machlarz
Givenname:	Artur
Family Name:	Machlarz
MBOX_SHA1SUM:	808fbd1bac667b3788b30a225a90156e5189e8d0
Weblog:	Info-Blog
Workplace:	Agencja Infobrokerska Infopoint.pl
Workplace:	Uniwersytet Opolski
School:	Uniwersytet Wrocławski
jabberID:	artur.machlarz@jabbim.pl
Languages:	**masters:** pl,en,de

FOAFer supports | What is FOAF? | *FOAFer* contact demo

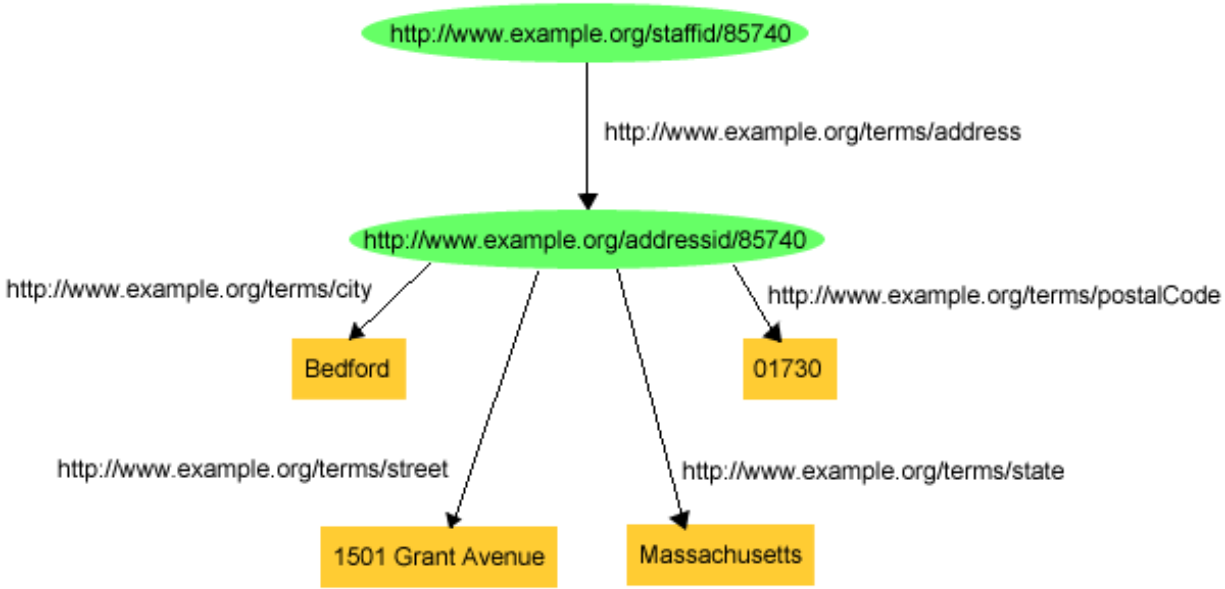
FOAF: browsers



Blank nodes

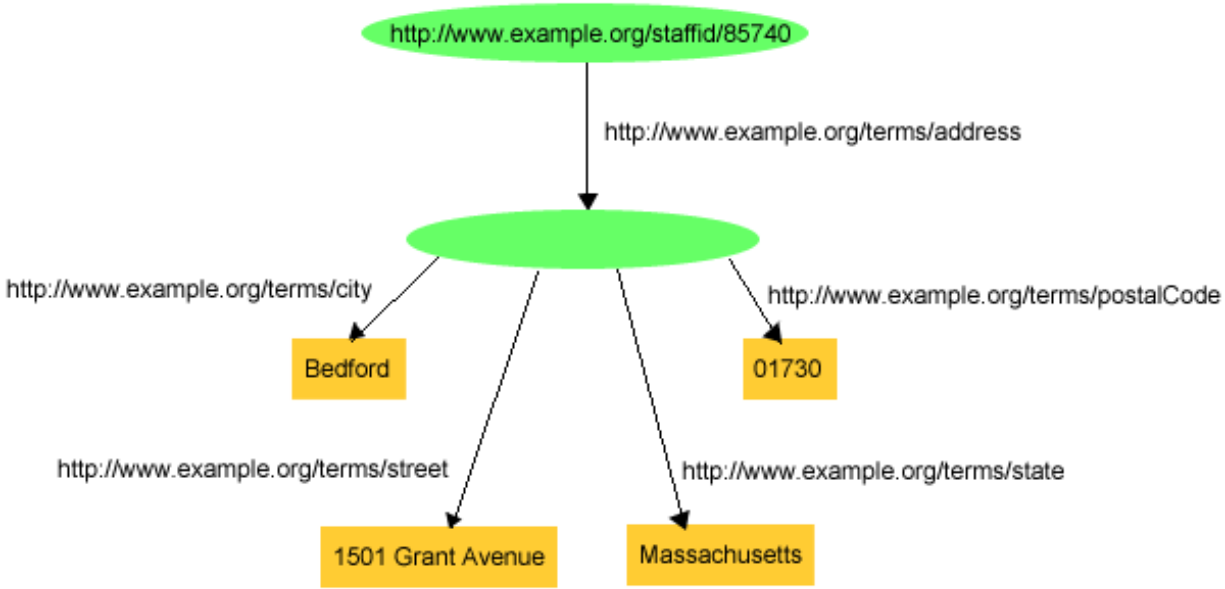
- RDF just supports triples, i.e., binary relationships
- Higher-order relationships must be broken down into many binary pieces
- Breaking down means creating additional nodes
- Such additional nodes will never be referenced from outside the current sub-graph → the don't need a name!
- A subject or object may be left "blank"

Example




```
exstaff:85740       exterms:address        exaddressid:85740 .
exaddressid:85740   exterms:street         "1501 Grant Avenue" .
exaddressid:85740   exterms:city           "Bedford" .
exaddressid:85740   exterms:state          "Massachusetts" .
exaddressid:85740   exterms:postalCode     "01730" .
```

Example – with blank node




```
exstaff:85740   exterms:address          _:johnaddress .
_:johnaddress   exterms:street           "1501 Grant Avenue" .
_:johnaddress   exterms:city             "Bedford" .
_:johnaddress   exterms:state            "Massachusetts" .
_:johnaddress   exterms:postalCode       "01730" .
```

Outline

Details on the XML serialization

- The XML document has a root node <**rdf:RDF**>
- Specifying the subject:
  - <**rdf:Description rdf:about**=”SubjectURIref”>
- Specifying properties, in the body of the rdf:Description tag
  - <ex:propertyName>ObjectLiteral</ex:propertyName>
  - <ex:otherProperty **rdf:resource**=”ObjectURIref” />
- Several triples sharing the same subject may be collected in the same rdf:Description body

Examples

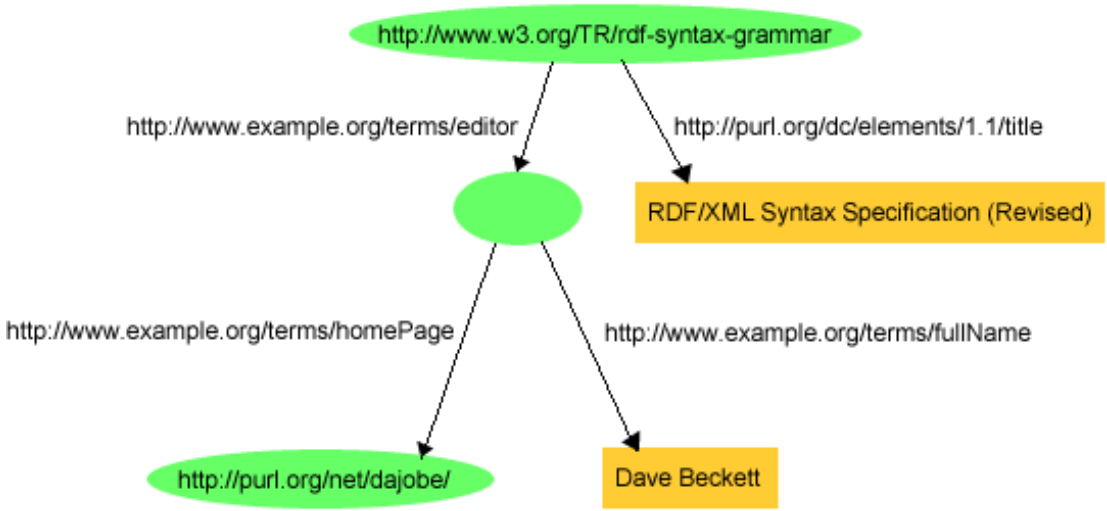
```
1. <?xml version="1.0"?>
2. <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3.           xmlns:exterms="http://www.example.org/terms/">

4.   <rdf:Description rdf:about="http://www.example.org/index.html">
5.       <exterms:creation-date>August 16, 1999</exterms:creation-date>
6.   </rdf:Description>

7. </rdf:RDF>
```

Examples

```
1. <?xml version="1.0"?>
2. <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3.            xmlns:exterms="http://www.example.org/terms/">

4.   <rdf:Description rdf:about="http://www.example.org/index.html">
5.       <exterms:creation-date>August 16, 1999</exterms:creation-date>
6.   </rdf:Description>

7. </rdf:RDF>
```

```
1.  <?xml version="1.0"?>
2.  <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3.             xmlns:dc="http://purl.org/dc/elements/1.1/"
4.             xmlns:exterms="http://www.example.org/terms/">

5.    <rdf:Description rdf:about="http://www.example.org/index.html">
6.         <exterms:creation-date>August 16, 1999</exterms:creation-date>
7.         <dc:language>en</dc:language>
8.         <dc:creator rdf:resource="http://www.example.org/staffid/85740"/>
9.    </rdf:Description>

10. </rdf:RDF>
```

Blank nodes in XML: rdf:nodeID




```
5.    <rdf:Description rdf:about="http://www.w3.org/TR/rdf-syntax-grammar">
6.      <dc:title>RDF/XML Syntax Specification (Revised)</dc:title>
7.      <exterms:editor rdf:nodeID="abc"/>
8.    </rdf:Description>

9.    <rdf:Description rdf:nodeID="abc">
10.       <exterms:fullName>Dave Beckett</exterms:fullName>
11.       <exterms:homePage rdf:resource="http://purl.org/net/dajobe/"/>
12.   </rdf:Description>
```

Typed literals in XML

```
ex:index.html   exterms:creation-date   "1999-08-16"^^xsd:date .
```

```
4.   <rdf:Description rdf:about="http://www.example.org/index.html">
5.     <exterms:creation-date rdf:datatype=
         "http://www.w3.org/2001/XMLSchema#date">1999-08-16
       </exterms:creation-date>
6.   </rdf:Description>
```

Outline

RDF Data structures

- Containers (unbounded)
  - `rdf:Bag` (unordered)
  - `rdf:Seq` (ordered)
  - `rdf:Alt` (one-of)
  - Semantically equivalent, the different beween Bag/Seq/Alt is only in its “intended usage”
  - Does not limit the member elements to the ones declared
- Collections (bounded)
  - `rdf:List`
  - Only the mentioned elements are part of the collection

Reification

- It may be sometimes useful to assert a statement *about* another statement.
  - For example, I want to say *who* added a fact (a triple) to my set of statements
- In this case, instead of writing the triple, we *describe* the triple by
  - **Giving a name** to the statement (`rdf:Statement`)
  - Giving the **elements of the triple** with `rdf:subject`, `rdf:predicate`, `rdf:object`

Example

```
exproducts:item10245   exterms:weight   "2.4"^^xsd:decimal .
```


```
exproducts:triple12345   rdf:type         rdf:Statement .
exproducts:triple12345   rdf:subject      exproducts:item10245 .
exproducts:triple12345   rdf:predicate    exterms:weight .
exproducts:triple12345   rdf:object       "2.4"^^xsd:decimal .
```

… and now the statement has a URIref: `this.rdf#triple12345`

Example (cont.)

```
exproducts:triple12345   rdf:type         rdf:Statement .
exproducts:triple12345   rdf:subject      exproducts:item10245 .
exproducts:triple12345   rdf:predicate    exterms:weight .
exproducts:triple12345   rdf:object       "2.4"^^xsd:decimal .
```

```
exproducts:triple12345   dc:creator       exstaff:85740 .
```

We expressed the `dc:creator` of the previous statement!

Entailment

- An RDF expression A is said to **entail** another RDF expression B if every possible arrangement of things in the world that makes A true also makes B true. On this basis, if the truth of A is presumed or demonstrated then the truth of B can be inferred.
- The mechanism for defining formal semantics for RDF
- The ultimate mechanism for creating reasoning engines in the semantic web
- Never asserts anything about “the things in the world”, only about the propagation of truth in RDF statements/assertions

More on this in the RDF Semantics chapter!

Outline

RDF Schema

- Special RDF vocabulary for describing the properties and the content of... RDF vocabularies
- Think of a definition (**schema**) of the nodes and predicates used in an RDF document.
  - However, this definition is expressed in RDF, too, by using the **RDFS vocabulary**
- With RDFS we may restrict the usage of RDF nodes and predicates, by introducing coherency and a sort of data types
- RDF Schema provides a **type system** for RDF

RDFS nature

- RDFS does **not** specify a vocabulary of *descriptive* properties such as “author”
- RDFS specifies **mechanisms** that may be used to name and describe properties and the classes of resource they describe
- Similar to the type systems of object-oriented programming languages, but:
  - OO languages define a class in terms of the properties its instances may have
  - RDFS describes properties in terms of the classes of resource to which they apply (domain & range)

Example

- OO language
  - define a class `eg:Book`
  - with an attribute called `eg:author`
  - of type `eg:Person`
- RDFS
  - define the `eg:author` property
  - to have a domain of `eg:Document`
  - and a range of `eg:Person`
- Why?
  - Easy for others to subsequently define additional properties with a domain of `eg:Document` or a range of `eg:Person`
  - This can be done without the need to re-define the original description of these classes
  - It allows anyone to extend the description of existing resources, one of the architectural principles of the Web

Defining Classes in RDFS

- `rdf:type`
  - Defines the 'type' of the subject node
  - The object of 'type' must be a class
- `rdfs:Class`
  - The set of all possible classes
  - A class is any resource having an `rdf:type` property whose value is the resource `rdfs:Class`

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

exthings:companyCar   rdf:type   ex:MotorVehicle .
```

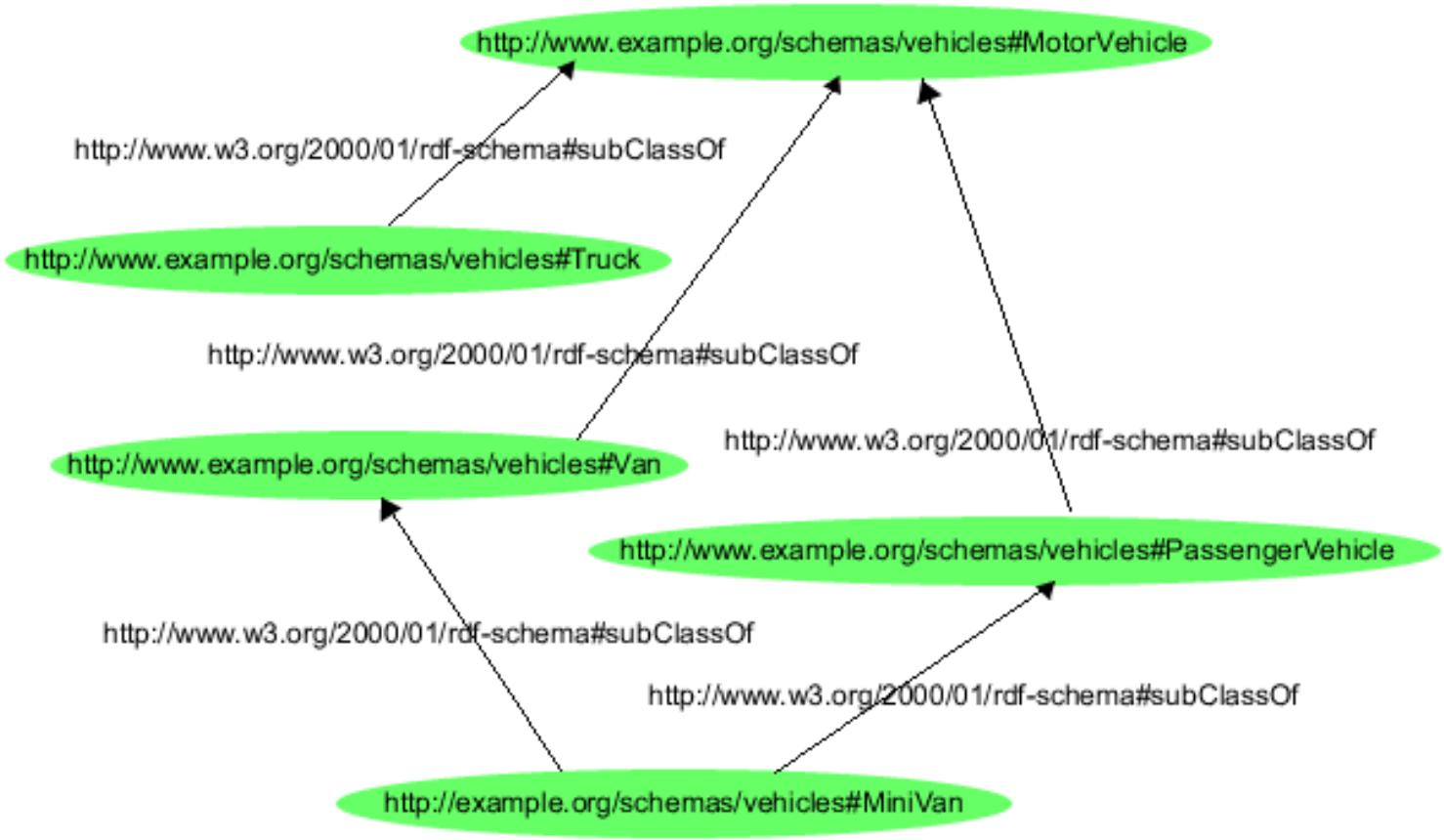
Defining class hierarchies

- `rdfs:subClassOf`
  - Defines a narrower class
  - Any instance of class `ex:Van` is also an instance of class `ex:MotorVehicle`
  - A transitive predicate

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

exthings:companyCar   rdf:type   ex:MotorVehicle .
```

```
ex:Van     rdf:type   rdfs:Class .
ex:Truck   rdf:type   rdfs:Class .

ex:Van   rdfs:subClassOf   ex:MotorVehicle .
```

Class hierarchies



Defining properties in RDFS

- `rdf:Property`
  - Any URIref used as a predicate has an `rdf:type` of `rdf:Property`
- `rdfs:domain`, `rdfs:range`
  - Define the domain and the range of the property
  - Domain and range are Classes
- `rdfs:subPropertyOf`
  - Defines hierarchies of properties

Example

```
<rdf:Property rdf:ID="registeredTo">
  <rdfs:domain rdf:resource="#MotorVehicle"/>
  <rdfs:range rdf:resource="#Person"/>
</rdf:Property>

<rdf:Property rdf:ID="rearSeatLegRoom">
  <rdfs:domain rdf:resource="#PassengerVehicle"/>
  <rdfs:range rdf:resource="&xsd;integer"/>
</rdf:Property>

<rdfs:Class rdf:ID="Person"/>

<rdfs:Datatype rdf:about="&xsd;integer"/>
```

RDF/RDFS Classes

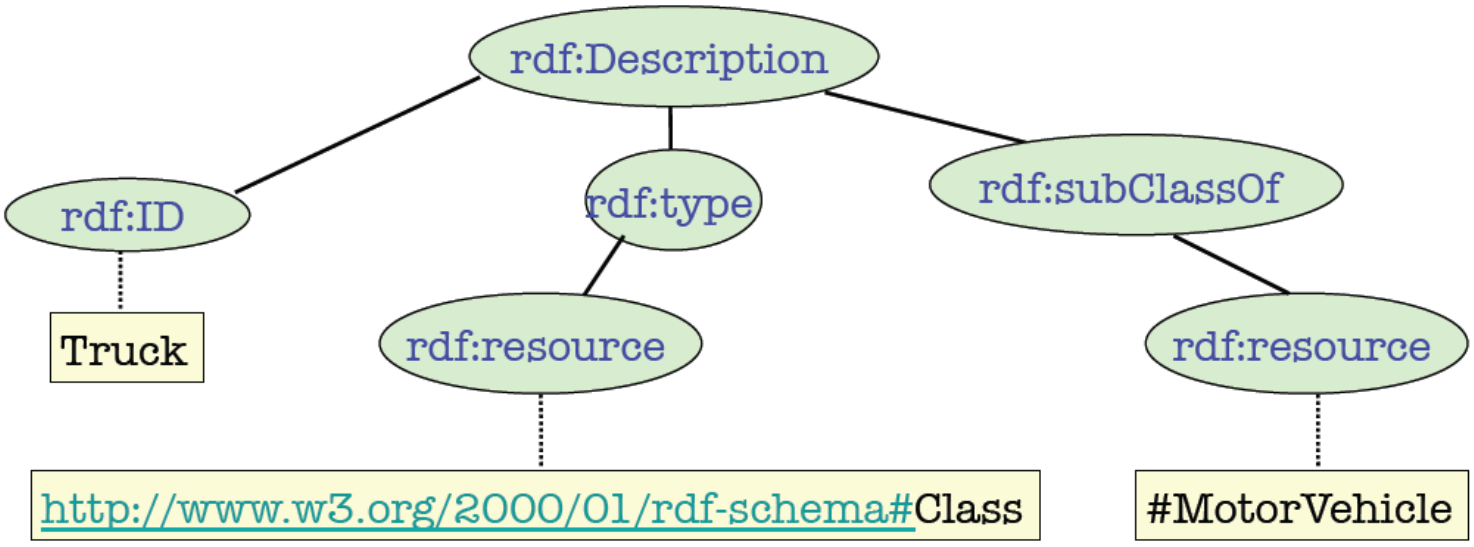
Class name	comment
rdfs:Resource	The class resource, everything.
rdfs:Literal	The class of literal values, e.g. textual strings and integers.
rdf:XMLLiteral	The class of XML literals values.
rdfs:Class	The class of classes.
rdf:Property	The class of RDF properties.
rdfs:Datatype	The class of RDF datatypes.
rdf:Statement	The class of RDF statements.
rdf:Bag	The class of unordered containers.
rdf:Seq	The class of ordered containers.
rdf:Alt	The class of containers of alternatives.
rdfs:Container	The class of RDF containers.
rdfs:ContainerMembershipProperty	The class of container membership properties, rdf:_1, rdf:_2, ..., all of which are sub-properties of 'member'.
rdf:List	The class of RDF Lists.

RDF/RDFS Properties

Property name	comment	domain	range
rdf:type	The subject is an instance of a class.	rdfs:Resource	rdfs:Class
rdfs:subClassOf	The subject is a subclass of a class.	rdfs:Class	rdfs:Class
rdfs:subPropertyOf	The subject is a subproperty of a property.	rdf:Property	rdf:Property
rdfs:domain	A domain of the subject property.	rdf:Property	rdfs:Class
rdfs:range	A range of the subject property.	rdf:Property	rdfs:Class
rdfs:label	A human-readable name for the subject.	rdfs:Resource	rdfs:Literal
rdfs:comment	A description of the subject resource.	rdfs:Resource	rdfs:Literal
rdfs:member	A member of the subject resource.	rdfs:Resource	rdfs:Resource
rdf:first	The first item in the subject RDF list.	rdf:List	rdfs:Resource
rdf:rest	The rest of the subject RDF list after the first item.	rdf:List	rdf:List
rdfs:seeAlso	Further information about the subject resource.	rdfs:Resource	rdfs:Resource
rdfs:isDefinedBy	The definition of the subject resource.	rdfs:Resource	rdfs:Resource
rdf:value	Idiomatic property used for structured values (see the RDF Primer for an example of its usage).	rdfs:Resource	rdfs:Resource
rdf:subject	The subject of the subject RDF statement.	rdf:Statement	rdfs:Resource
rdf:predicate	The predicate of the subject RDF statement.	rdf:Statement	rdfs:Resource
rdf:object	The object of the subject RDF statement.	rdf:Statement	rdfs:Resource

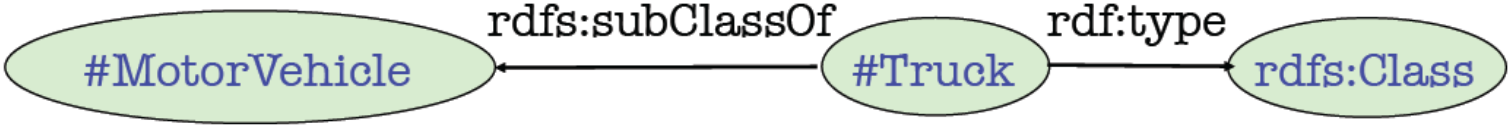
Note the multiple views: XML

```
<rdf:Description rdf:ID="Truck">
  <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
  <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
</rdf:Description>
```

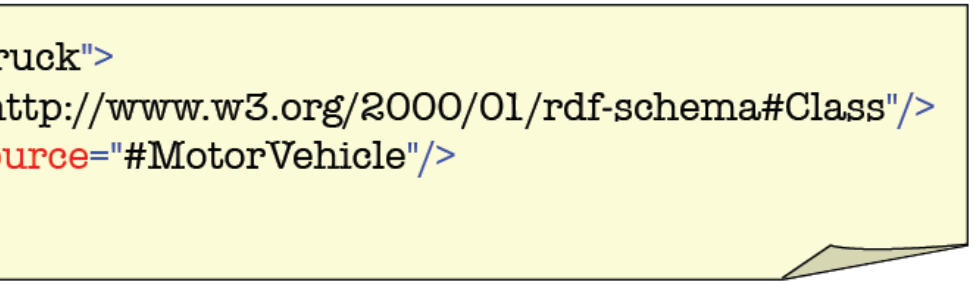


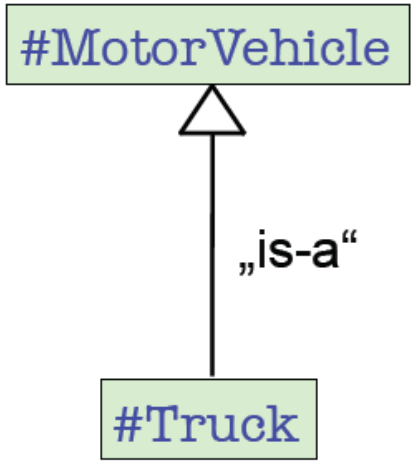
Note the multiple views: RDF

```
<rdf:Description rdf:ID="Truck">
  <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
  <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
</rdf:Description>
```



Note the multiple views: RDF Schema





References

- RDF Primer – W3C Recommendation 10 February 2004
  - http://www.w3.org/TR/rdf-primer/
- Resource Description Framework (RDF): Concepts and Abstract Syntax – W3C Recommendation 10 February 2004
  - http://www.w3.org/TR/rdf-concepts/
- RDF Vocabulary Description Language 1.0: RDF Schema – W3C Recommendation 10 February 2004
  - http://www.w3.org/TR/rdf-schema/

License

- This work is licensed under the Creative Commons **Attribution-Noncommercial-Share Alike** 3.0 Unported License.
- To view a copy of this license, visit http://creativecommons.org/licenses/by-nc-sa/3.0/ or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.

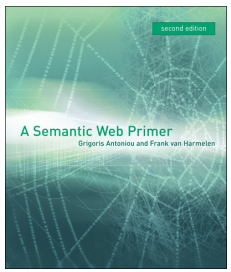
Questions

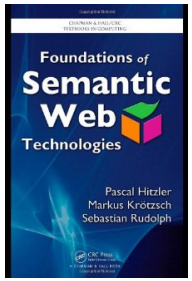
Any questions?

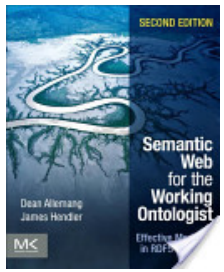
Thank you

Thank you for your attention!

http://geist.agh.edu.pl
GEIST Research Group







Powered by LaTeX