Chapter 13
Metadata and 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."
– Tim Berners-Lee, James Hendler, Ora Lassila, The Semantic Web, Scientific American, May 2001

- application-level description of Web sources and facts
- metadata (e.g. keywords in header section of Web pages)
  problem: which keywords to use?
- "Ontologies": describe and relate concepts
  (e.g., geographics, bioinformatics, ...)
- Portals that provide information and services on a given topic.

Literature: e.g. Eckstein/Eckstein: "XML und Datenmodellierung" (dpunkt), Ch. 6,7

ONTOLOGIES

Description of concepts

- what concepts
- properties
- relationships (hierarchy, uses-relationship, any others)

Example: Biology

biological *taxonomy*: animals, plants, mammals, fishes, insects, carnivores, felidae, canidae, bovinae

- taxonomy: hierarchy
- sometimes also: typical properties (base for taxonomy)
- Ontology: additionally describe all kinds of relationships (e.g. that carnivores eat animals)
- more than schema: schema with annotations *what* notions mean

WEB PORTALS

Provide services/access for a given ontology, e.g.

- tourism, traveling
- bioinformatics

Communication with clients, servers, and users via notions of an ontology:

- information about the information/services provided by resources ("Yellow-pages", represented e.g. in RDF (Resource Description Framework))
- queries and results/answers according to a given format (data in XML, according to a DTD/XML Schema)
- mapping from individual DTD/XML Schemas to global ontology
- Portal: maps different schemas to the common one (based on ontology).

USE ONTOLOGIES FOR DESCRIBING NON-STRUCTURED CONTENTS

- describe content of pictures and films
  - pictures: description of the picture (where some item is found on the picture), and of its "message" ("what is depicted there" (application level))
  - films: additionally, temporal aspects

important: common ontologies for each of the aspects.

Portals then use these annotations:

Example

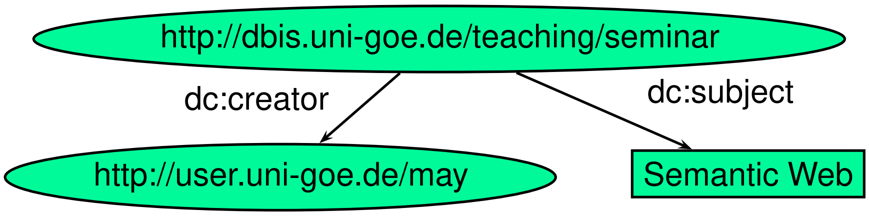
An annotated information video on the "BSc/MSc studies in Applied Computer Science at Göttingen University" could contribute to answers:

- how are german BSc/MSc studies structured, what are ECTS points?
- what is the contents of "Informatik I" in GOE?
- where can I study bioinformatics in Germany?
- (for local students:) how many CP do I need in the "Application Area"?

13.1 RDF: Resource Description Framework

Extension of the ideas of conceptual modeling (ER-Model, UML)

- first Working Draft: 1997
- triple-based "RDF statements": (*subject*, *predicate*, *object*)
- graphical representation (⇒ RDF graph), XML representation
- *subject* and *object* are *resources*
  (cf. terminology for XLink)
- a resource is everything ... that can be described by a URI
  (graphical notation: an ellipse)
  - a Web page, a book, a lecture ...



Example: Dublin Core

"Dublin Core": a set of properties for describing (HTML Web) documents (defined at a metadata workshop in Dublin/Ohio 1995).

- Title: A name given to the resource.
- Creator: An entity primarily responsible for making the content of the resource.
- Subject: The topic of the content of the resource.
- Description: An account of the content of the resource.
- Publisher: An entity responsible for making the resource available.
- Contributor: An entity responsible for making contributions to the content of the resource.
- Date
- Type: The nature or genre of the content of the resource.
- Language: A language of the intellectual content of the resource.

etc ... see `http://dublincore.org/documents/dces/`

RDF Model and Triples

- Initial goal: semantic description of things on the Web (i.e., Web pages and parts of them) author, keywords, *annotation* of links
  goal: semantics-driven Web indexing and search engines
- Information about real resources (*Triples, Statements*):

  (<http://dbis.uni-goe.de> dc:creator "Wolfgang May")
  (<http://dbis.uni-goe.de> dc:subject "Database Group")

... now it's possible to "search for Web pages of database groups".

RDF as a Knowledge Representation Framework

The Semantic Web intention was not only to annote documents, but also to relate them by annotating e.g. hyperlinks:

Objects of statements can also be (actual) resources:

(<http://dbis.uni-goe.de> uni:seminar <http://dbis.uni-goe.de/teaching/seminar>)

... and to search for seminars of database groups.

(<http://user.cs.uni-goe.de/~may> dc:author "Wolfgang May")

Search for Web pages that have been authored by german computer scientists?
⇒ requires a description of "Wolfgang May".

RDF as a Knowledge Representation Framework

Things that are not "real" resources that have a URL (e.g. in HTTP) get a virtual *Unified Resource Identifier*, e.g.

(<http://user.cs.uni-goe.de/∼may> dc:creator <mail:may@cs.uni-goettingen.de>)
(<http://user.cs.uni-goe.de/∼may> dc:creator <bla:person-D-12345678>)

and can then be described:

(<mail:may@cs.uni-goe.de> uni:profession "Computer Scientist")
(<mail:may@cs.uni-goe.de> bla:lives <geo://country-de/city-goettingen>)

RDF for Data Integration

- information contained in the triples is independent from where they are is located, and from their order.
- RDF files (knowledge bases) in different places that use the same URIs for "things" and the same "names" for properties can be easily combined.
- these "names" are taken from *domain ontologies*.
- for XML data, te ontologies are provided by the namespaces (element and attribute names) and the DTDs.

RDF Notations

- the above *"N-triple"* notation:
  - *subjects* of triples: resources – having a URI
  - *objects* of triples: resources or literals (here: character strings)
  - *predicates* of the statements are resources:
    - *namespace:localpart*, i.e.,
      mon:city with xmlns:mon="http://www.mondial.de/mondial-nmspc#" yields
      http://www.mondial.de/mondial-nmspc#city
      which actually is a (really useful) URI.
  - note: it is –like XML – in ASCII format (but much older).
- XML serialization of RDF is broadly used.
  See `http://www.w3.org/RDF/Validator/` for visualization of XML-serialized RDF:
  - RDF graph
  - RDF triples
- RDF model: Everything that is not a literal value is a resource, including things, properties, classes.

Example (with actual resources)

```xml
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:mon="http://www.mondial.de/names#">
  <rdf:Description rdf:about="http://www.deutschland.de">
    <mon:name>Germany</mon:name>
    <mon:code>D</mon:code>
    <mon:has_province>
      <rdf:Description rdf:about="http://www.niedersachsen.de">
        <mon:name>Niedersachsen</mon:name>
        <mon:has_city>
          <rdf:Description rdf:about="http://www.hannover.de">
            <mon:name>Hannover</mon:name>
            <mon:in_country rdf:resource="http://www.deutschland.de"/>
            <mon:population>
              <rdf:Description> <mon:year>1995</mon:year>
                                <mon:value>525763</mon:value>
              </rdf:Description>
            </mon:population>
          </rdf:Description>
        </mon:has_city>
        <mon:has_capital rdf:resource="http://www.hannover.de"/>
      </rdf:Description>
    </mon:has_province>
  </rdf:Description>
</rdf:RDF>
```

- there can be different RDF files that describe Germany, Lower Saxonia (e.g. more cities) and Hannover in more detail.

Example B (with actual resources)

```xml
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:uni="http://www.education.de/names#">
  <rdf:Description rdf:about="http://dbis.informatik.uni-goettingen.de">
    <uni:name>DBIS</uni:name>
    <uni:uni rdf:resource="http://www.uni-goettingen.de"/>
    <uni:institute rdf:resource="http://www.informatik.uni-goettingen.de"/>
    <uni:person>
      <rdf:Description rdf:about="http://user.informatik.uni-goettingen.de/~may">
        <uni:name>Wolfgang May</uni:name>
      </rdf:Description>
    </uni:person>
    <uni:leader rdf:resource="http://user.informatik.uni-goettingen.de/~may"/>
    <uni:lecture>
      <rdf:Description rdf:about="http://dbis.informatik.uni-goettingen.de/Teaching/SSD">
        <uni:name>Semistructured Data and XML</uni:name>
        <uni:lecturer rdf:resource="http://user.informatik.uni-goettingen.de/~may"/>
        <uni:assistance rdf:resource="http://user.informatik.uni-goettingen.de/~behrends"/>
        <uni:assistance rdf:resource="http://user.informatik.uni-goettingen.de/~fritzen"/>
      </rdf:Description>
    </uni:lecture>
  </rdf:Description>
</rdf:RDF>
```

- there can be different RDF files that describe the university at Göttingen, the CS institute, E.Behrends and O.Fritzen in more detail.

RDF AS AN XML APPLICATION

Root element type: <rdf:RDF >
contains <rdf:Description> elements

<rdf:Description> Element

(subject, predicate, object)

- which resource is described: @rdf:about="*uri*"
- subelements for describing properties:
  - element name: name of the property
  - value of the property:
    * element contents:
      text content or one or more nested <rdf:Description> elements
    * attribute @rdf:resource="*uri*": property points to another resource that has an RDF description of its own elsewhere

URIS AND URLS

URI according to RFC 2396 (http://www.ietf.org/rfc/rfc2396.txt)

- a URI can be a locator (URL), a "name" (URN), or a "general" URI.
  General form: *scheme*:*hierarchical_part*
- URIs: a sequence of characters; often with a hierarchical structure
- URLs are those URIs that identify resources via their physical access mechanism (i.e., http, ftp, gopher, file, ...), in general named after a protocol.
  http://www.informatik.uni-goettingen.de/people.html
  mailto:may@informatik.uni-goettingen.de
  news:de.comp.text.tex
- URI schemes can be "registered" at www.IANA.org (Internet Assigned Numbers Authority)
  example (non-registered): isbn:3-89722-153-5
- URIs serve as agreed *identifiers* in a certain community.
- some of them are valid URLs, some of them not.

RDF uses general URIs, not only URLs.

Example (with virtual resources)

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:mon="http://www.mondial.de/names#">
  <rdf:Description rdf:about="geo://country/de">
    <mon:name>Germany</mon:name>
    <mon:code>D</mon:code>
    <mon:has_province>
      <rdf:Description rdf:about="geo://country/de/prov/niedersachsen">
        <mon:name>Niedersachsen</mon:name>
        <mon:has_city>
          <rdf:Description rdf:about="geo://country/de/prov/niedersachsen/city/hannover">
            <mon:name>Hannover</mon:name>
            <mon:in_country rdf:resource="geo://country/de"/>
            <mon:population>
              <rdf:Description> <mon:year>1995</mon:year> <mon:value>525763</mon:value>
              </rdf:Description>
            </mon:population>
          </rdf:Description>
        </mon:has_city>
        <mon:has_capital rdf:resource="geo://country/de/prov/niedersachsen/city/hannover"/>
      </rdf:Description>
    </mon:has_province>
  </rdf:Description>
</rdf:RDF>
```

- geo: virtual “protocol” that represents the structure,
- mon: namespace where the names of the relationships come from.

- the graph is the same, only the URIs differ

Example B (with virtual resources)

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:uni="http://www.education.de/names#">
  <rdf:Description rdf:about="bla://unigoe.de/inf/dbis">
    <uni:name>DBIS</uni:name>
    <uni:uni rdf:resource="bla://unigoe.de"/>
    <uni:institute rdf:resource="bla://unigoe.de/inf"/>
    <uni:person>
      <rdf:Description rdf:about="bla://unigoe.de/inf/people/may">
        <uni:name>Wolfgang May</uni:name>
      </rdf:Description>
    </uni:person>
    <uni:leader rdf:resource="bla://unigoe.de/inf/people/may"/>
    <uni:lecture>
      <rdf:Description rdf:about="bla://unigoe.de/inf/Teaching/SSD">
        <uni:name>Semistructured Data and XML</uni:name>
        <uni:lecturer rdf:resource="bla://unigoe.de/inf/people/may"/>
        <uni:assistance rdf:resource="bla://unigoe.de/inf/people/behrends"/>
        <uni:assistance rdf:resource="bla://unigoe.de/inf/people/fritzen"/>
      </rdf:Description>
    </uni:lecture>
  </rdf:Description>
</rdf:RDF>
```

- the graph is the same, only the URIs differ

URIs

- basic idea of RDF: resources are globally identified by their URI
  - can be a real URL
  - can be a virtual URI
  - RDF instances can mix them (e.g. when describing Web pages, their authors etc.)
- virtual URIs in a global application scenario
  - virtual structuring of the resources
  - globally known and agreed URI schema
  - all members of a "community" can describe the resources by RDF

URI Representation/Construction Mechanisms

- describe a remote resource via its full URI (as above)
  - attribute @rdf:about="*uri*" identifies a remote resource
- use a base URI by xml:base that sets the base URI for resolving relative URI references (i.e., rdf:about, rdf:resource, rdf:ID and rdf:datatype), otherwise the base URI is that of the document.
  - set xml:base="*uri*" (e.g. in the root element)
  - @rdf:about="*relativepath*": the resource's global URI is then composed as *xmlbase relativepath*
  - @rdf:ID="*localpart*": the resource's global URI is then composed as *xmlbase#localpart*
  - then, use @rdf:resource="#*localpart*" in the object position for referencing it.
- only locally known IDs:
  - attribute @rdf:nodeID= "*name*": defines and describes a *local* resource that can be referenced only inside the same RDF instance by its ID
  - then, use @rdf:nodeID="*id*" in the object position of a property instead of @rdf:resource="*uri*"

Example (with base URI and relative paths)

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:mon="http://www.mondial.de/names#"
     xml:base="http://www.mondial.de/things/">
  <rdf:Description rdf:about="country/de">
    <mon:name>Germany</mon:name>  <mon:code>D</mon:code>
    <mon:has_province>
      <rdf:Description rdf:about="country/de/prov/ndrs">
        <mon:name>Niedersachsen</mon:name>
        <mon:has_city>
          <rdf:Description rdf:about="country/de/city/hann">
            <mon:name>Hannover</mon:name>
            <mon:in_country rdf:resource="country/de"/>
            <mon:population>
              <rdf:Description> <mon:year>1995</mon:year> <mon:value>525763</mon:value>
              </rdf:Description>
            </mon:population>
          </rdf:Description>
        </mon:has_city>
        <mon:has_capital rdf:resource="country/de/city/hann"/>
      </rdf:Description>
    </mon:has_province>
  </rdf:Description>
</rdf:RDF>
```

- global URIs are e.g. http://www.mondial.de/things/country/de/city/hann and http://www.mondial.de/names#name;
  note: xml:base must end with a “/”.

Example (with base URI and local name parts)

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:mon="http://www.mondial.de/names#"
     xml:base="http://www.mondial.de/things#">
  <rdf:Description rdf:ID="de">
    <mon:name>Germany</mon:name>  <mon:code>D</mon:code>
    <mon:has_province>
      <rdf:Description rdf:ID="de-prov-ndrs">
        <mon:name>Niedersachsen</mon:name>
        <mon:has_city>
          <rdf:Description rdf:ID="de-city-hann">
            <mon:name>Hannover</mon:name>
            <mon:in_country rdf:resource="#de"/>
            <mon:population>
              <rdf:Description> <mon:year>1995</mon:year> <mon:value>525763</mon:value>
              </rdf:Description>
            </mon:population>
          </rdf:Description>
        </mon:has_city>
        <mon:has_capital rdf:resource="#de-city-hann"/>
      </rdf:Description>
    </mon:has_province>
  </rdf:Description>
</rdf:RDF>
```

- global URIs are e.g. http://www.mondial.de/things#de-city-hann and http://www.mondial.de/names#name;
  note: the “#” at the end of xml:base is optional.

Example (with local IDs)

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:mon="http://www.mondial.de/names#">
  <rdf:Description rdf:nodeID="de">
    <mon:name>Germany</mon:name>
    <mon:code>D</mon:code>
    <mon:has_province>
      <rdf:Description rdf:nodeID="de-prov-ndrs">
        <mon:name>Niedersachsen</mon:name>
        <mon:has_city>
          <rdf:Description rdf:nodeID="de-city-hann">
            <mon:name>Hannover</mon:name>
            <mon:in_country rdf:nodeID="de"/>
            <mon:population>
              <rdf:Description> <mon:year>1995</mon:year> <mon:value>525763</mon:value>
              </rdf:Description>
            </mon:population>
          </rdf:Description>
        </mon:has_city>
        <mon:has_capital rdf:nodeID="de-city-hann"/>
      </rdf:Description>
    </mon:has_province>
  </rdf:Description>
</rdf:RDF>
```

- global URIs are e.g. http://www.mondial.de/names#name;
- the local resources cannot be referenced by other RDF instances.

Prefix Expansion in N3

Triple Notation in N3:

&prefix dc: <http://purl.org/dc/elements/1.1/>.
&prefix goe: <http://dbis.uni-goe.de/>.
(<http://dbis.uni-goe.de> dc:creator goe:may).

- URLs in <...> vs. qualified names (*prefix*:*localname*) that expand to a URL.
- URL: part in front of the ":" is the protocol name
- qname: part in front of the ":" is the namespace prefix that is resolved.

The above is expanded to

(<http://dbis.uni-goe.de>
<http://purl.org/dc/elements/1.1/#creator>
<http://dbis.uni-goe.de/may>)

- all positions (not only the predicate) are subject to expansion.

NAMES VS. URIS – XMLNS VS. XML:BASE

XML-serialized RDF:

- element and attribute names are subject to namespace expansion,
- URLs in rdf:about, rdf:resource, rdf:ID and rdf:datatype are subject to expansion with xml:base.
  If one wants to use different base URLs there, *entities* are the right choice:

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
   <!ENTITY mon "http://www.mondial.de/"> ]
>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:mon="http://www.mondial.de/"
         xml:base="http://www.mondial.de/">
  <rdf:Description rdf:about="country/de"
     mon:name="Germany" mon:code="D">
    <!-- this is not expanded correctly: -->
    <mon:has_city rdf:resource="mon:country/de/city/hannover"/>
    <!-- the right way is to use an entity: -->
    <mon:has_city rdf:resource="&mon;country/de/city/hannover"/>
  </rdf:Description>
</rdf:RDF>
```


ABBREVIATED FORMS AND FURTHER FUNCTIONALITY

- short form: property attributes instead of property subelements
- omit “blank” description nodes (those without a relevant ID) by
  <*property-name* rdf:parseType=“Resource”> ... </*property-name*>
  (following example: city population data)
- if all properties of the result of a property P are string-valued, they can be added as attributes to the element P.
  (following example: city population data)
- collections: parseType=“Collection”

... but normally, RDF is not created manually, but via scripts (or XQuery statements) from databases.

Example: Abbreviated Forms

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:mon="http://www.mondial.de/names#"
         xml:base="http://www.mondial.de/things/"
  <rdf:Description rdf:about="country/de"
    mon:name="Germany" mon:code="D">
    <mon:has_province>
      <rdf:Description rdf:about="country/de/prov/niedersachsen">
        <mon:name>Niedersachsen</mon:name>
        <mon:has_city>
          <rdf:Description rdf:about="country/de/prov/niedersachsen/city/hannover"
               mon:name="Hannover">
            <mon:in_country rdf:resource="country/de"/>
            <mon:population rdf:parseType="Resource">
              <mon:year>1995</mon:year>
              <mon:value>525763</mon:value>
            </mon:population>
            <mon:population mon:year="2002" mon:value="515001"/>
          </rdf:Description>
        </mon:has_city>
        <mon:has_capital rdf:resource="country/de/prov/niedersachsen/city/hannover"/>
      </rdf:Description>
    </mon:has_province>
  </rdf:Description>
</rdf:RDF>
```


Example: a Collection-Valued Property

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:uni="http://www.education.de/names#">
  <rdf:Description rdf:about="http://dbis.informatik.uni-goettingen.de">
    <uni:name>DBIS</uni:name>
    <uni:uni rdf:resource="http://www.uni-goettingen.de"/>
    <uni:institute rdf:resource="http://www.informatik.uni-goettingen.de"/>
    <uni:person>
      <rdf:Description rdf:about="http://user.informatik.uni-goettingen.de/~may"
                       uni:name="Wolfgang May"/>
    </uni:person>
    <uni:leader rdf:resource="http://user.informatik.uni-goettingen.de/~may"/>
    <uni:lecture>
      <rdf:Description rdf:about="http://dbis.informatik.uni-goettingen.de/Teaching/SSD">
        <uni:name>Semistructured Data and XML</uni:name>
        <uni:lecturer rdf:resource="http://user.informatik.uni-goettingen.de/~may"/>
        <uni:semesters>
         <rdf:Seq>
          <rdf:li rdf:resource="http://dbis.informatik.uni-goettingen.de/Teaching/SSD/SS03"/>
          <rdf:li rdf:resource="http://dbis.informatik.uni-goettingen.de/Teaching/SSD/SS04"/>
         </rdf:Seq>
        </uni:semesters>
      </rdf:Description>
    </uni:lecture>
  </rdf:Description>
</rdf:RDF>
```

- note the names _1 and _2 for the sequence elements in the graph.

13.2 RDF Metadata Information

... so far, RDF defines just an edge-labeled graph.

RDF provides also a simple type system (which is exteded later by RDF Schema):

- rdf:type: is a special property that assigns a type to something
(note that there is no way in RDF to say what are the allowed types – this will be only the case in RDFS!)

```
<rdf:Description rdf:about="http://www.deutschland.de">
  <rdf:type rdf:resource="http://www.mondial.de/names#country"/>
    :
</rdf:Description>
```

- Type not only given by the type name, but a URI that “represents” the type.

⇒ types are also resources that are associated with a URI

Types as Resources?

... this is, what a namespace URI does in XML [e.g. for element names]:

Consider RDF

```
<rdf:RDF xmlns:rdf="..."
      xmlns:mon="http://www.mondial.de/names#"
      xml:base="http://www.mondial.de/things/">
 <rdf:Description rdf:about="country/de">
  <rdf:type rdf:resource="http://www.mondial.de/names#country"/>
  <mon:name>Germany<mon:name>
 </rdf:Description>
</rdf:RDF>
```

and XML

```
<mon:mondial xmlns:mon="http://www.mondial.de/names#">
 <mon:country car_code= "D">
  <mon:name>Germany<mon:name>
 </mon:country>
</mon:mondial>
```

There’s more behind these namespaces, which is not used in plain XML.

Metadata Information: Types, Properties, and Ontologies

Types and properties (i.e., everything that is used in a namespace) are not only “names”, but are resources “somewhere in the Web”, identified by a URI (used in RDF or in XML via namespaces).

⇒ a *domain ontology* describes the notions used in a namespace.

Ontologies

- what types/classes are there,
- subclass information,
- what properties objects of a given type must have,
- cardinalities of properties,
- default values,
- to what types some property is applicable,
- that some properties are transitive, symmetric, subproperties of another or excluding each other etc.

Ontologies

Schema languages, metadata languages, modeling languages, ontologies ...

- every schema description defines a (more or less complete) ontology:
- ER Model (1976, entity types, attributes, relationships with cardinalities),
- UML (1997, classes with subclasses, associations with cardinalities etc.),
- F-Logic signatures (1989),
- ... RDFS and OWL (Web Ontology Language)

Metadata/Ontology Information in the Web

Description of types via its URI/URL (e.g., by an OWL ontology) can provide additional information about resources of a type, or about a property:

- this can be RDFS files actually accessible by HTTP (see later),
- or virtual URIs that are *described* in some (RDF/RDFS/OWL) file.

Types as Resources

rdf:Description allows also to describe things that are intended to be types:

```
<rdf:RDF xmlns:rdf="..."
      xml:base="http://www.mondial.de/names#">
  <rdf:Description rdf:ID="country">
   here we can describe the resource  http://www.mondial.de/names#country
  </rdf:Description>
</rdf:RDF>
```

... but the RDF vocabulary does not allow to describe much about types (even not that this thing *is* a type!).

Properties as Resources

- rdf:Property: this *is* a type, properties are ... properties:
  - An RDF axiom: (rdf:type rdf:type rdf:property)
    "the type (2nd)
      of "rdf:type" (1st, which is http://www.w3.org/1999/02/22-rdf-syntax-ns#type) is
      "rdf:Property"",
    means resources can have a property "rdf:type".
  - The statement (http://.../...#has_capital rdf:type rdf:property)
    means that "http://.../...#has_capital" is a property.

⇒ properties are also resources that are associated with a URI
(again, namespace URIs do this in XML [element and attribute names]):

```
<rdf:RDF xmlns:rdf="..."
      xml:base="http://www.mondial.de/names#">
  <rdf:Property rdf:ID="has_capital">
   here we can describe the resource  http://www.mondial.de/names#has_capital
  </rdf:Property>
</rdf:RDF>
```

... but what does RDF actually provide?

RDF Metadata Information

With the predefined RDF metadata vocabulary, one can

- say that a resource R is of some kind T: (R rdf:type T)
  - without having a possibility what types T exist,
- say that a resource P is not a standard "thing", but a property: (P rdf:type rdf:Property)
  - without saying what properties things of a given type can have (domain) or what kind of type the result should have (range)

The RDF Semantics and Model Theory (see `http://www.w3.org/TR/rdf-mt`) is based on rdf:type and rdf:Property.

RDF Metadata Information: Examples

as long as "naive" models are used, there are some hints:

- Consider
  reasonable: (http://.../geo#germany rdf:type http://.../geo#country)
  reasonable: (http://.../geo#berlin rdf:type http://.../geo#city)
  non-reasonable: (http://.../geo#germany rdf:type http://.../geo#berlin)
  ⇒ things that are of some type should not be types themselves?
- but this crashes with meta-metadata:
  reasonable: (http://.../bio#penguin rdf:type http://.../bio#species)
  reasonable: (http://.../bio#tweety rdf:type http://.../bio#penguin)
- Consider
  reasonable: (http://...#has_capital rdf:type rdf:property)
  not reasonable: (http://...#has_capital http://...#has_population 42)
  ⇒ a property does not have other properties?
- but this crashes with meta-metadata:
  reasonable: (http://.../geo#has_capital rdf:type rdf:property)
  reasonable: (http://.../geo#has_capital http://.../meta#has_range http://.../geo#city)

⇒ a very weak model-theory.

Data and Metadata

At least, it is possible to

- describe data and metadata in the same file,
- have them in the same (virtual) URI structure:

http://www.mondial.de/rdf#name,
http://www.mondial.de/rdf#country,
http://www.mondial.de/rdf#country/de

- and to add descriptions of these things in other RDF files.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:mon="http://www.mondial.de/rdf#"
     xmlns:comment="http://www.mondial.de/comment#"
     xml:base="http://www.mondial.de/rdf">
  <rdf:Description rdf:ID="country">
    <bla:comment> cannot assign an RDF type to this
    </bla:comment>
  </rdf:Description>
  <rdf:Description rdf:ID="province"/>
  <rdf:Property rdf:ID="has_province"/>
  <rdf:Property rdf:ID="name"/>
  <rdf:Property rdf:ID="code"/>
  <rdf:Description rdf:about="country/de">
    <rdf:type rdf:resource="#country"/>
    <mon:name>Germany</mon:name>
    <mon:code>D</mon:code>
    <mon:has_province>
      <rdf:Description rdf:about="country/de/prov/nrds">
        <rdf:type rdf:resource="#province"/>
        <mon:name>Niedersachsen</mon:name>
      </rdf:Description>
    </mon:has_province>
  </rdf:Description>
</rdf:RDF>
```


Reification

Take statements (=triples) as resources and make statements about them:

- `rdf:Statement` which has properties `rdf:subject`, `rdf:predicate`, `rdf:object`, that yield a resource
  (note that RDF cannot formalize the above sentence – this will be possible with RDFS)!
- XML: give an ID to the statement.
  "The statement "Germany had 83536115" was valid in year 1997":

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:mon="http://www.mondial.de/rdf#"
     xml:base="http://www.mondial.de/rdf">
  <rdf:Description rdf:about="country/de">
    <rdf:type rdf:resource="#country"/>
    <mon:name>Germany</mon:name>
    <mon:population rdf:ID="de-pop">83536115</mon:population>
  </rdf:Description>
  <rdf:Description rdf:about="#de-pop">
    <mon:year>1997</mon:year>
  </rdf:Description>
</rdf:RDF>
```

Reification: Example

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:mon="http://www.mondial.de/rdf#"
     xml:base="http://www.mondial.de/rdf">
  <rdf:Description rdf:about="country/de">
    <rdf:type rdf:resource="#country"/>
    <mon:name>Germany</mon:name>
    <mon:population rdf:ID="de-pop">83536115</mon:population>
  </rdf:Description>
  <rdf:Description rdf:about="#de-pop">
    <mon:year>1997</mon:year>
  </rdf:Description>
</rdf:RDF>
```

Triples (added automatically by the RDF semantics (see RDF validator)):

(<http://www.mondial.de/rdf#country/de> <http://www.mondial.de/rdf#population> "83536115")

(<http://www.mondial.de/rdf#de-pop> rdf:type rdf:Statement)

(<http://www.mondial.de/rdf#de-pop> rdf:subject <http://www.mondial.de/rdf#country/de>)

(<http://www.mondial.de/rdf#de-pop> rdf:predicate <http://www.mondial.de/rdf#population>)

(<http://www.mondial.de/rdf#de-pop> rdf:object "83536115")

(<http://www.mondial.de/rdf#de-pop> <http://www.mondial.de/rdf#year> "1997")

Reification in the Semantic Web

Annotating Statements:

- with probabilities, trust, "who says ...",
- annotations can be in different files than the annotated statements (cf. out-of-line XLink arcs),
- can be used for reasoning.

Note:

- information about a *predicate* (which describes the predicate "name" (e.g., the source where all this data is taken from, transitivity or symmetry) or the set of all instances (cardinalities), or each its (range, domain))

is different

- from describing/annotating a *statement* (i.e. one instance of a predicate).

Additional RDF Vocabulary

The rdf namespace provides some more vocabulary for

- Collections: `rdf:Alt`, `rdf:Bag`, `rdf:Seq`, `rdf:List` are collections.
  Lists have properties `rdf:first` (a resource) and `rdf:rest` (a list). Others have properties _1, _2, ... that refer to their members.
- and some more.

... but not yet really useful. It's a model for representing facts as triples.

Example: World Wide RDF Web

- many such information sources that describe resources
- higher level information management (e.g., portals): use some of these sources for accessing *integrated* information

Example (RDF source see next slide) – the example is not based on real data

- mondial: countries, cities
- `http://sw.mondial.de/mondial.rdfs`: meta information about geography ontology
- another resource: cities and their airports
- `http://sw.iata.org/onto.rdfs` (International Air Transport Assoc.): ontology about flight information
- `bla:sw.iata.org/flights/`*flight*: resource associated with a given flight (e.g. LH42).
- `bla:sw.iata.org/airports/`*abbrev*: resource associated with a given airport (e.g., FRA, CDG).
- there will probably be a Lufthansa RDF database that describes the flights in their terminology

Example (Cont'd) [Filename: RDF/flightbase.rdf]

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

  <rdf:Description rdf:ID="de"
     xmlns:mon="http://www.mondial.de/onto.rdfs#">
    <rdf:type rdf:resource="http://www.mondial.de#country"/>
    <mon:name>Germany</mon:name>
    <mon:has_population rdf:parseType="Resource">
      <rdf:type rdf:resource="http://www.mondial.de#measure"/>
      <mon:year>1995</mon:year>  <mon:value>83536115</mon:value>
    </mon:has_population>
    <mon:has_city>
      <rdf:Description rdf:about="http://www.berlin.de">
        <rdf:type rdf:resource="http://www.mondial.de#city"/>
        <mon:name>Berlin</mon:name>
      </rdf:Description>
    </mon:has_city>
    <mon:has_capital rdf:resource="http://www.berlin.de"/>
    <mon:has_neighbor rdf:resource="#fr"/>
  </rdf:Description>

  <rdf:Description rdf:ID="fr"
      xmlns:mon="http://www.mondial.de/onto.rdfs#">
    <rdf:type rdf:resource="http://www.mondial.de#country"/>
    <mon:name>France</mon:name>
    <mon:has_population rdf:parseType="Resource">
      <rdf:type rdf:resource="http://www.mondial.de#measure"/>
      <mon:year>1995</mon:year>  <mon:value>58317450</mon:value>
    </mon:has_population>
    <mon:has_city>
      <rdf:Description rdf:about="http://www.paris.fr">
        <rdf:type rdf:resource="http://www.mondial.de#city"/>
        <mon:name>Paris</mon:name>
      </rdf:Description>
    </mon:has_city>
    <mon:has_capital rdf:resource="http://www.paris.fr"/>
    <mon:has_neighbor rdf:resource="#de"/>
  </rdf:Description>

  <rdf:Description rdf:about="http://www.berlin.de"
       xmlns:touri="http://www.world-tourism.org/onto.rdfs#">
    <touri:has_airport rdf:resource="bla:sw.iata.org/airports/BLN"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.paris.fr"
       xmlns:touri="http://www.world-tourism.org/onto.rdfs#">
    <touri:has_airport rdf:resource="bla:sw.iata.org/airports/CDG"/>
  </rdf:Description>

  <rdf:Description rdf:about="bla:sw.iata.org/flights/LH42"
       xmlns:iata="http://sw.iata.org/onto.rdfs#">
    <rdf:type rdf:resource="http:sw.iata.org/onto.rdfs#flight"/>
    <iata:from rdf:resource="bla:sw.iata.org/airports/BLN"/>
    <iata:to rdf:resource="bla:sw.iata.org/airports/CDG"/>
  </rdf:Description>
</rdf:RDF>
```


RDF vs. XML

Everything that can be expressed by XML can also be expressed by RDF

(Exercise: create an RDF mondial instance with local resources)

+ RDF can also be used to *describe* resources
  (pictures, films, ..., programs, Web services, ...)
+ RDF can be represented as a graph, independent from the structure of the (distributed) RDF instances
+ RDF data can be distributed in different files that describe the same resources
+ RDF has a connection to global schema description mechanisms (see later)

o RDF can be queried in the same way by XPath/XQuery ...

- but: which RDF instances?
  – if local resources are used: relatively easy
  – if global resources are used: appropriate RDFs must be searched for.

? High-level metadata query languages (proposal: e.g. RQL) ...

13.3 RDF Schema

- RDF ist the instance level
- XML: DTDs and XML Schema for describing the structure/schema of the instance
- RDF Schema: stronger than DTD/XML – "semantic-level"
  - describe the structure of the RDF instance (i.e. the "schema" of the RDF graph, not of the XML-RDF file):
    - "rdf:" elements part described by RDF DTD
    - names of property subelements/attributes follow from an (application-specific) RDFS instance
      (that is coupled with the namespace declaration)
  - describes the schema in terms of an (lightweight) ontology (OWL provides then much more features):
    - *notions* are described and related *semantically*:
      - class/subclass
      - property/subproperty, domains and ranges
- classes and properties are also resources, described by RDFS instances.

RDF Schema Documents

Similar to RDF/XML documents:

- additionally include rdfs namespace declaration

  <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

- description of classes

  <rdfs:Class rdf:about="*uri*"> <!-- used by <rdf:type resource="*uri*">-->
  <rdfs:subClassOf rdf:resource="*uri*">
  </rdfs:Class>

- description of properties

  <rdf:Property rdf:about="*uri*"> <!-- used by <*namespace:name*>-->
  <rdfs:subPropertyOf rdf:resource="*uri*"/>
  <rdfs:domain rdf:resource="*uri*"/>
  <rdfs:range rdf:resource="*uri*"/>
  </rdf:Property>

- for @rdf:about="*uri*" the other notations (xml:base + local part, or local-ids) can be used.

Defining URIs of RDFS Classes and Properties

Classes and properties are “usual” resources, identified/defined by

<rdfs:Class rdf:about=“*uri*”> . . . </rdfs:Class>
reference by rdf:resource=“*uri*”

<rdfs:Class rdf:ID=“*classname*”> . . . </rdfs:Class> (+ base-uri)
reference by rdf:resource=“*#classname*” (local)
reference by rdf:resource=“*base-uri#classname*” (from remote)

<rdfs:Class rdf:nodeID=“*classname*”> . . . </rdfs:Class>
reference by rdf:nodeID=“*classname*” (only for local definitions)

- references to classes: see above
- references to properties:
  - from the RDF instance: by property names (including namespaces)
    <*namespace:property*> – expanding the xmlns:*namespace*=“*uri*” declaration
  - from other RDFS elements (e.g. in rdf:subproperty): similar to references to classes (@rdf:resource or @rdf:nodeID)

Version A: Classes and Properties as “Real” Resources in the RDFS Instance

Classes

Anything that is defined in an RDFS document - e.g., in

http://www.mondial.de/mondial.rdfs#

(or with appropriate setting of xml:base) as an

<rdfs:Class rdf:ID=“country”>
<!-- SubclassOf-defs etc.-->
</rdfs:Class>

can be used in another RDF document as an rdf:type, e.g.

<rdf:Description rdf:ID=“country-D”>
mondial:name=“Germany” mondial:carcode= “D”>
<rdf:type rdf:resource=“http://www.mondial.de/Mondial/mondial.rdfs#country”/>
</rdf:Description>

Properties

Consider RDF triples:

- subject, object: identified by a URI or a local ID
- predicate: of the form *namespace:property*, e.g., mondial:capital

Using Namespaces

Namespaces are bound to a URI by the namespace declaration

xmlns:mondial="http://www.mondial.de/mondial.rdfs#"

extending <mondial:capital> yields the URL (with local part)

http://www.mondial.de/mondial.rdfs#capital

that addresses

```
<rdf:Property rdf:ID="capital">
  <rdfs:domain rdf:resource="#political_entity"/>
    <!-- i.e., country, province, district, county -->
  <rdfs:range rdf:resource="#city"/>
</rdf:Property>
```


VERSION B: "VIRTUAL" RESOURCES

Classes

- straightforward by <rdfs:Class rdf:about="*uri*"> and <rdf:type rdf:resource="*uri*"/>

Properties

- the same way: the whole vocabulary is "identified" with a base URI, e.g. www.mondial.de (where no special RDFS instance is located)
- the RDFS document (located somewhere) uses rdf:about= "*base-uri/property-name*"
- the namespace is declared as xmlns:*namespace*="*base-uri/*"
- and *namespace:property* refers to *base-uri/property-name*

COMPARISON

- Version A: class/property resources are fragments of the RDFS instance:
  - \+ @rdf:resource can actually be dereferenced and yields the class/property definition
  - \- only practical if the RDFS is non-distributed
    (although remote RDFS instances can also describe this resource by using rdfs:about)

  ⇒ centralized ontologies
- Version B: class/property resources are identified by a virtual URI
  - \+ arbitrary RDFS instances can contribute to the resource description
  - \- users/clients have to know where the resource descriptions can be found

  ⇒ distributed ontologies (only a central/common schema for class/property URIs required)

RDFS DATATYPES

- classes: complex, application-specific notion
- literal datatypes: basic, common notion

RDFS does not define its own datatypes ... it uses those from XML Schema:

- Literal values can refer to their XML Schema datatypes as e.g.

<rdf:Description rdf:ID="D">
<rdf:type rdf:resource="http://www.mondial.de/mondial.rdfs#country"/>
<mondial:name rdfs:Datatype="http://www.w3.org/2001/XMLSchema#string">
Germany</mondial:name>
<mondial:area rdfs:Datatype="http://www.w3.org/2001/XMLSchema#float">
356910</mondial:area>
</rdf:Description>

(note: `http://www.w3.org/2001/XMLSchema#` can be defined as an entity in the local DTD to the RDF/RDFS instance and then used as rdfs:Datatype="&xsd;string")

USAGE OF RDFS DATATYPES IN METADATA

- usually, the datatype information is not provided in the RDF instance, but in the RDFS Schema:
- Range and domain of <rdf:Property> can refer to XML Schema types, e.g.

```
<rdf:Property rdf:ID="population">
  <rdfs:domain rdf:resource="#political_entity"/>
    <!-- i.e., country, province, district, county -->
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/>
</rdf:Property>
```


Example: Typing

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:mon="http://www.mondial.de/mondial.rdfs#">
  <rdf:Description rdf:about="mon://country/de"
    mon:name="Germany" mon:code="D">
    <rdf:type rdf:resource="http://www.mondial.de/mondial.rdfs#country"/>
    <mon:has_province>
      <rdf:Description rdf:about="mon://country/de/prov/niedersachsen">
        <rdf:type rdf:resource="http://www.mondial.de/mondial.rdfs#province"/>
        <mon:name>Niedersachsen</mon:name>
        <mon:has_city>
          <rdf:Description rdf:about="mon://country/de/prov/niedersachsen/city/hannover"
               mon:name="Hannover">
            <rdf:type rdf:resource="http://www.mondial.de/mondial.rdfs#city"/>
            <mon:in_country rdf:resource="mon://country/de"/>
            <mon:population rdf:parseType="Resource">
              <rdf:type rdf:resource="http://www.mondial.de/mondial.rdfs#measure"/>
              <mon:year rdfs:Datatype="http://www.w3.org/2001/XMLSchema#integer">1995</mon:year>
              <mon:value rdfs:Datatype="http://www.w3.org/2001/XMLSchema#integer">525763</mon:value>
            </mon:population>
          </rdf:Description>
        </mon:has_city>
        <mon:has_capital rdf:resource="mon://country/de/prov/niedersachsen/city/hannover"/>
      </rdf:Description>
    </mon:has_province>
  </rdf:Description>
</rdf:RDF>
```

Components of an RDFS instance

- application-oriented data: structured by (subject, predicate, object) tripels, values
- application-oriented metadata: information about types/classes, names of properties
- RDF/RDFS "infrastructure"

Example: Components of an RDFS instance

```
<rdf:RDF xmlns:rdf=”...” xmlns:mon=”http://www.mondial.de/mondial.rdf#”>
 <rdf:Description rdf:ID=”de” mon:name=”Germany” mon:code=”D”>
  <rdf:type rdf:resource=”http://www.mondial.de/mondial.rdf#country”/>
  <mon:has_province>
   <rdf:Description rdf:ID=”de-ndrs” mon:name=”Niedersachsen”>
    <rdf:type rdf:resource=”http://www.mondial.de/mondial.rdf#province”/>
    <mon:has_city>
     <rdf:Description rdf:ID=”de-hann” mon:name=”Hannover”>
      <rdf:type rdf:resource=”http://www.mondial.de/mondial.rdf#city”/>
      <mon:in_country rdf:resource=”#de”/>
      <mon:population rdf:parseType=”Resource”>
       <rdf:type rdf:resource=”http://www.mondial.de/mondial.rdf#measure”/>
       <mon:year rdfs:Datatype=“http://www.w3.org/2001/XMLSchema#integer”>1995</mon:year>
       <mon:value rdfs:Datatype=“http://www.w3.org/2001/XMLSchema#integer”>525763</mon:value>
      </mon:population>
     </rdf:Description>
    </mon:has_city>
    <mon:has_capital rdf:resource=”#de-hann”/>
   </rdf:Description>
  </mon:has_province> </rdf:Description>  </rdf:RDF>
```

RDFS IS ITSELF AN RDF APPLICATION

Note: rdfs:Class is a shorthand for RDF where classes are defined as instances of the RDFS class “Class”:

<rdf:Description id=“country”>
<rdf:type rdf:resource=“http://www.w3.org/2000/01/rdf-schema#Class”/>
<rdfs:Subclass ... >
</rdf:Description>

- analogously for `http://www.w3.org/2000/01/rdf-schema#Property`
- analogously for rdfs:range and rdfs:domain as properties whose domain is a property and whose range is the set of classes.
- a class can be a subclass of itself (allows for defining equivalence)

⇒ RDF model theory (allows for formal reasoning)
related to *Description Logics*

SUMMARY: RDF SCHEMA VOCABULARY

Classes on the Meta Level

- rdfs:Resource, rdfs:Class, rdfs:Literal, rdfs:Datatype
- rdf:Property

Properties on the Meta Level

- rdfs:domain, rdfs:range
- rdfs:subclassOf, rdfs:subpropertyOf

⇒ can be used as
  - integrity constraints:
    “an interpretation is only an RDFS model if $A \subseteq B$
  - axioms for drawing conclusions: “since our knowledge is an RDFS model we can conclude from knowing $A \subseteq B$ that ... holds.

Other

- (rdfs:Container, rdfs:member, rdfs:ContainerMembershipProperty)

RDFS AXIOMATIC TRIPLES

See RDF Semantics and Model Theory, `http://www.w3.org/TR/rdf-mt`.

Axioms: expected to hold in and RDF structure:

```
rdf:type rdfs:domain rdfs:Resource .
rdfs:domain rdfs:domain rdf:Property .
rdfs:range rdfs:domain rdf:Property .
rdfs:subPropertyOf rdfs:domain rdf:Property .
rdfs:subClassOf rdfs:domain rdfs:Class .

rdf:type rdfs:range rdfs:Class .
rdfs:domain rdfs:range rdfs:Class .
rdfs:range rdfs:range rdfs:Class .
rdfs:subPropertyOf rdfs:range rdf:Property .
rdfs:subClassOf rdfs:range rdfs:Class .

rdfs:Datatype rdfs:subClassOf rdfs:Class .
```

... and some more.

USING RDF IN THE WORLD WIDE WEB

- The (Semantic) Web is not seen as a collection of documents, but as a collection of correlated information (described via documents)
- using RDF, everybody can make statements about any resource (cf. link-bases in XLink)
  - incremental, world wide data and meta-data
  - distributed RDFS,
  - distributed RDF,
  - often using only virtual resources.
- not assumed that complete information about any resource is available.

REASONING BASED ON RDFS

- RDF/RDFS *model theory* as above.
- incomplete knowledge when reasoning
  (“open world assumption” – monotonic or nonmonotonic reasoning)
- potentially even inconsistent
- statements can be equipped with probabilities or labeled as opinions;
  fuzzy reasoning, belief revision ...

... lots of artificial intelligence applications ...

... but there is even more.

Chapter 14
Ontologies and the Web Ontology Language – OWL

- *vocabularies* can be defined by RDFS
  - not so much stronger than the ER Model or UML
    (even weaker: no cardinalities)
  - not only conceptual model, but a “real language”
  - *incremental* world-wide approach
  - “global” vocabulary can be defined by autonomous partners
- but: still restricted when *describing* the vocabulary.

Ontologies/ontology languages further extend the expressiveness:

- Description Logics
- Topic Maps (in SGML) since early 90s, XTM (XML Topic Maps)
- Ontolingua – non-XML approach from the Knowledge Representation area
- OIL (Ontology Inference Layer): initiative funded by the EU programme for Information Society Technologies (project: On-To-Knowledge, 1.2000-10.2002); based on RDF/RDFS
- DAML (Darpa Agent Markup Language; 2000) ... first ideas for a Semantic Web language
- DAML+OIL (Jan. 2001)
- developed into OWL (1st version March 02, finalized Feb. 04)

THREE VARIANTS OF OWL

Several expressiveness/complexity/decidability levels:

- OWL Full: extension of RDF
  - classes can also be regarded as individuals (classes of classes ... higher-order reasoning)
- OWL DL
  - fragment of OWL that fits into the Description Logics Framework
  - decidable reasoning
- OWL Lite
  - subset of OWL DL
  - easier migration from frame-based tools
    (note: F-Logic was a frame-based framework)
  - easier reasoning

OWL INSTANCE STRUCTURE

extend RDF (and collect namespaces ...)

```
<rdf:RDF xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
 <!-- now the Ontology header follows -->
 <owl:Ontology
      rdf:about="... base-uri for subsequent stuff ..."/>
 <!-- and then the ontology itself -->
 :
</rdf:RDF">
```

- the <owl:Ontology> element can contain additional information about the version, previous versions etc.

Content of the OWL instance: RDF/RDFS with additional OWL information

- RDFS class definitions
- RDFS property definitions

CLASSES

Class definition: name introduction or reference + defining statements:

- <owl:Class rdf:ID="*new_name*"> ... </owl:Class> defines a new class (that can then be referenced by the URI *currentdocument#new_name*)
- <owl:Class rdf:about="*uri*"> ... </owl:Class> adds information about a class that is defined elsewhere

The elements in the contents of the owl:Class element then describe restrictions that must (all) be satisfied.

Expressing Information about Classes

- <rdfs:subClassOf> (with @resource or contents)
- <owl:equivalentClass> or <owl:sameAs> (sameAs derived as subPropertyOf from subClassOf)
- <owl:unionOf>, <owl:intersectionOf>
- <owl:complementOf>, <owl:disjointWith>
- <owl:oneOf> (enumeration of elements)

Anonymous classes (as classes in the above elements)

Elements <owl:Restriction> ... </owl:Restriction> define *anonymous* classes that satisfy the given restrictions
(owl:Restriction is a subclass of owl:Class):

- <owl:onProperty rdf:resource="..."> ... </owl:onProperty>
  - <owl:hasValue>, <owl:allValuesFrom>, <owl:someValuesFrom>,
  - <owl:cardinality>, <owl:minCardinality> or <owl:maxCardinality>
  - including @rdfs:Datatype restrictions for the range

PROPERTIES

derived from <rdf:Property>:

- <rdfs:ObjectProperty> – subclass of rdf:Property; object-valued (i.e. rdfs:range must be a Object class)
- <rdfs:DatatypeProperty> – subclass of rdf:Property; datatype-valued (i.e. rdfs:range must be an xsd:datatype)

Properties can then be specified in more detail:

- <rdfs:subPropertyOf rdf:resource="..."/>
- <owl:constraintProperty>
- <owl:equivalentProperty>
- <owl:inverseOf rdf:resource="...">
- <owl:TransitiveProperty> , <owl:SymmetricProperty>, <owl:FunctionalProperty> , <owl:InverseFunctionalProperty>
  (often as <rdf:Type rdf:resource="&owl;..."/>)

OBJECTS

- used in enumerations (<owl:oneOf>)
- used to describe actual instances
- <owl:Thing>
  - <owl:sameIndividualAs> – states that 2 resources are the same thing
  - <owl:differentFrom> – forbids deriving equivalence

```
<owl:Thing rdf:ID="Germany">
  <rdf:Type rdf:resource="#Country">
  <owl:sameIndividualAs ...>
  <owl:differentFrom resource="#France">
</owl:Thing>
```

equivalent short form:

```
<Country rdf:ID="Germany"> ... </Country>
```


SEMANTIC WEB: XML; RDF AND OWL

In contrast to XPath/XQuery, XSLT, XML Schema, XLink etc., RDF and OWL are *not* languages *"inside"* the XML world, but are concepts of their own that have - incidentally- also an XML syntax.

The combination of XML data and RDF/RDFS/OWL concepts is the base for the *Semantic Web*.

A Semantic Web application e.g. exists of

- a "central" portal that uses the following things:
- a set of ontological (OWL, RDFS) sources,
- a set of RDF sources,
- reasoning (using OWL and RDFS information),
- a semantical description of itself for allowing others to use it.

Literature

`http://www.w3.org/2004/OWL/`: OWL Use Cases and Requirements, OWL Language Overview, OWL Language Guide ...