Semantic Web & RDF

Karsten Tolle

Database and Information Systems (DBIS)

JOHANN WOLFGANG GOETHE
UNIVERSITÄT
FRANKFURT AM MAIN

Semantic Web: a definition

The **Semantic Web** is the representation of **data** on the World Wide Web. It is a collaborative effort led by W3C with participation from a large number of researchers and industrial partners. It is based on the Resource Description Framework (RDF), which integrates a variety of applications using XML for syntax and URIs for naming.

TBL Statement

"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*



Problems with search engines today:

- homonyms → you find to much
- synonyms → you do not find what you could

When the machines could understand the semantic of the (meta-)data, much more advanced applications could be possible.

Semantic Web Scenarios

- looking for a free space in a doctors time table and fix an appointment
- providing further search or filtering opportunities, i.e. find a specific business near your home
- comparing prices of products with specific criteria like duration of warranty

Resource Description Framework

- The Resource Description Framework (RDF) is a general-purpose language for representing information in the Web.

Gives a unique convention for metadata according:

- syntax
- structure
- semantic

Status of specifications

- **Resource Description Framework (RDF) Model and Syntax Specification** – W3C Recommendation (1999)
  it introduces the RDF data model and a syntax for representing and exchanging RDF descriptions

- **RDF/XML Syntax Specification (Revised)**
  – W3C Proposed Recommendation (Dec. 2003)
  replaces the RDF M&S above

Status of specifications

all W3C Proposed Recommendation (Dec. 2003)

- **RDF Vocabulary Description Language 1.0: RDF Schema** – introduces a schema specification language for representing RDF metadata schemas (or vocabularies)
- **RDF Primer** – to provide the basic knowledge required to effectively use RDF (an introduction)
- **Resource Description Framework (RDF): Concepts and Abstract Syntax** – defines an abstract syntax on which RDF is based (design goals, key concepts, datatyping, character normalization and handling of URI references)
- **RDF Semantics** – a precise semantics, and corresponding complete systems of inference rules
- **RDF Test Cases** – for parser and inference engines

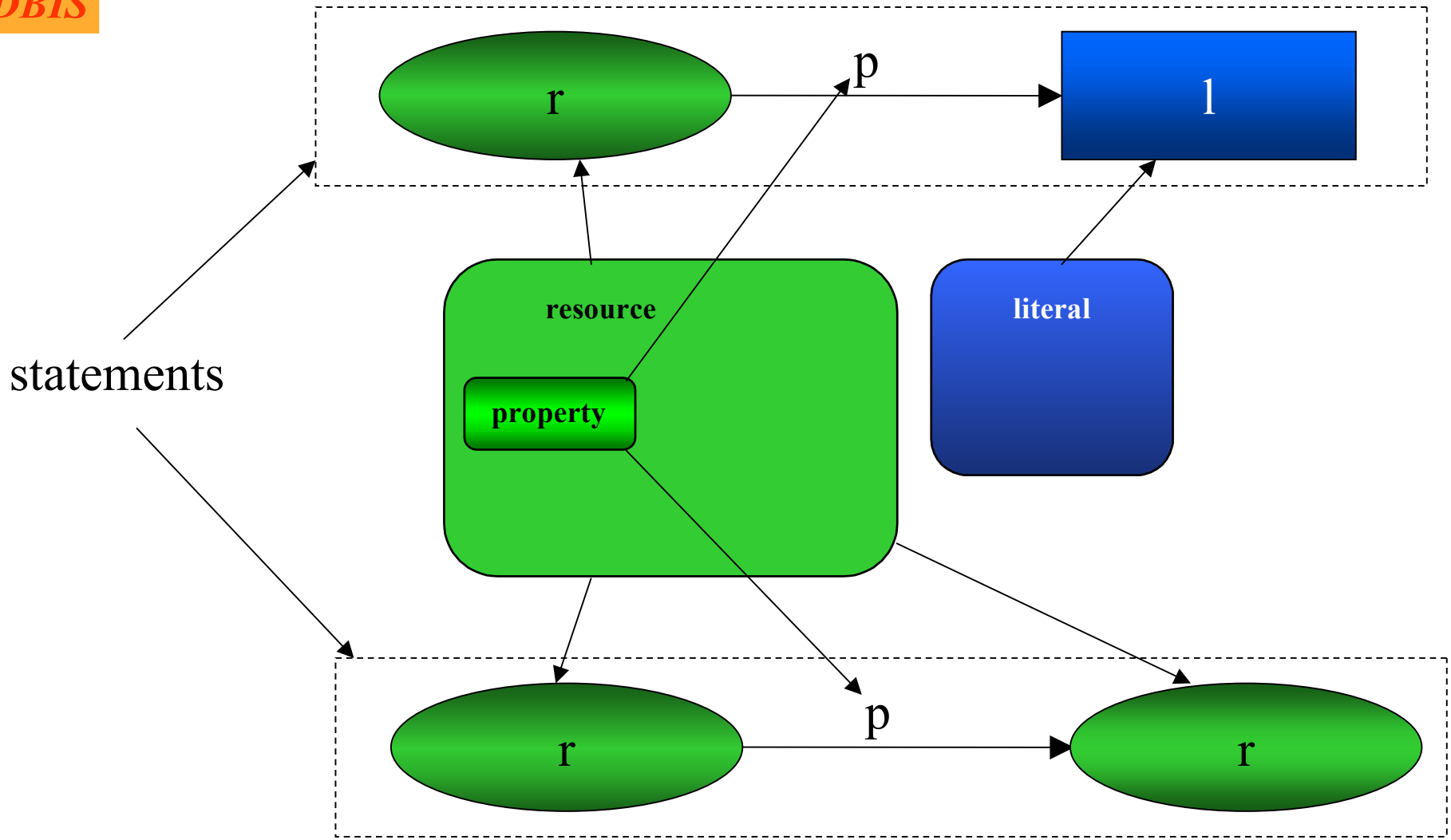
Goals of RDF

- **Independence**. It should be possible for anyone to define its own schemas and use and/or reuse them in a specific semantic way.
- **Interchange**. It should be easy to transport and storage the metadata described by RDF.
- **Scalability**. Even for a huge set of metadata it should be easy to handle and process them.

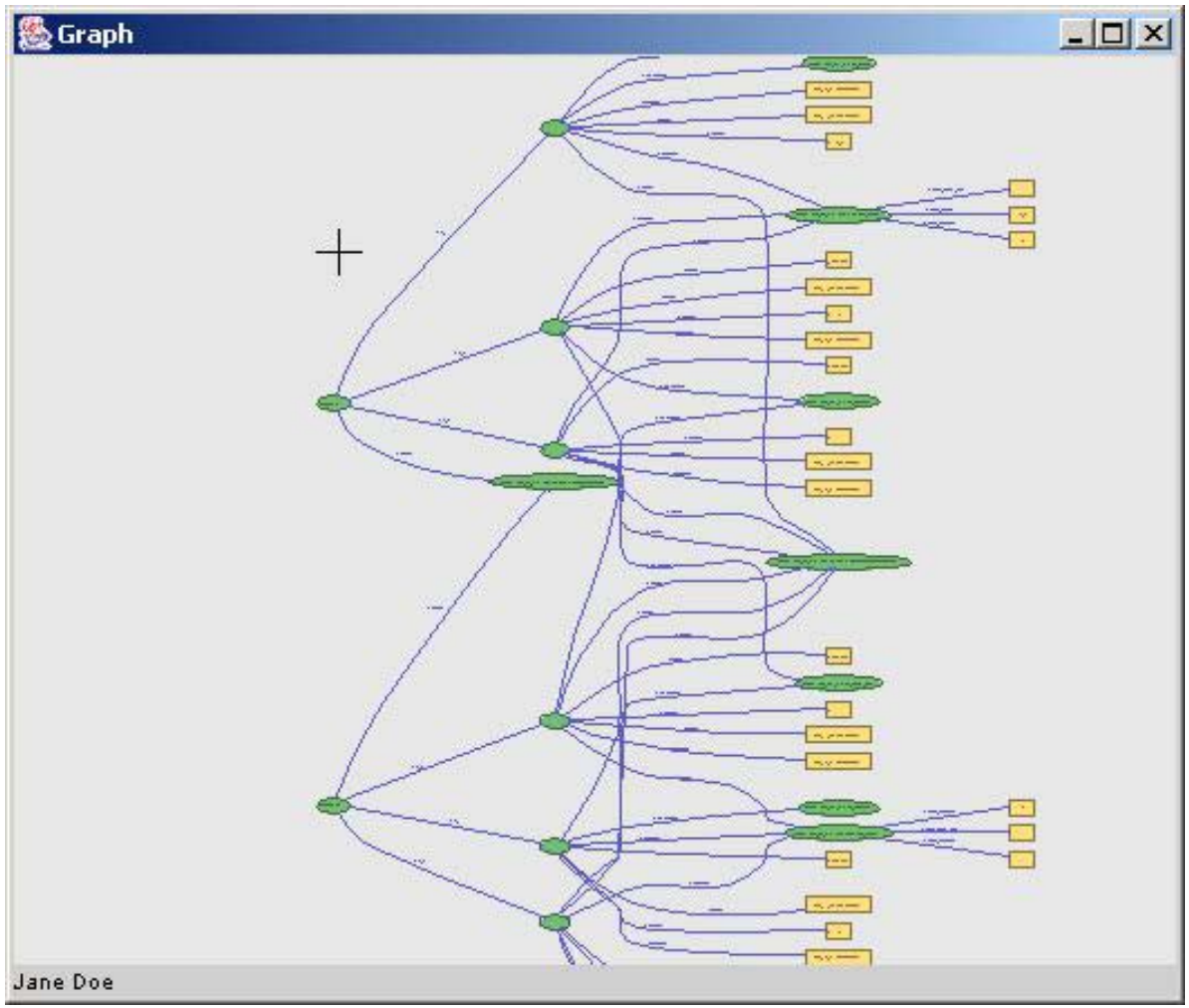
RDF Data Model

- RDF Data Model is based on the notions of **resource, property, literal** and **statement**
  - **resource** ≡ everything identified by a URI
  - **property** ≡ specific characteristics of a resource; properties are resources!
  - **literal** ≡ constant value represented by character strings
  - **statement** ≡ a specific *resource* together with a *property* and a *value* (either a resource or a literal), respectively called *subject*, *predicate* and *object*

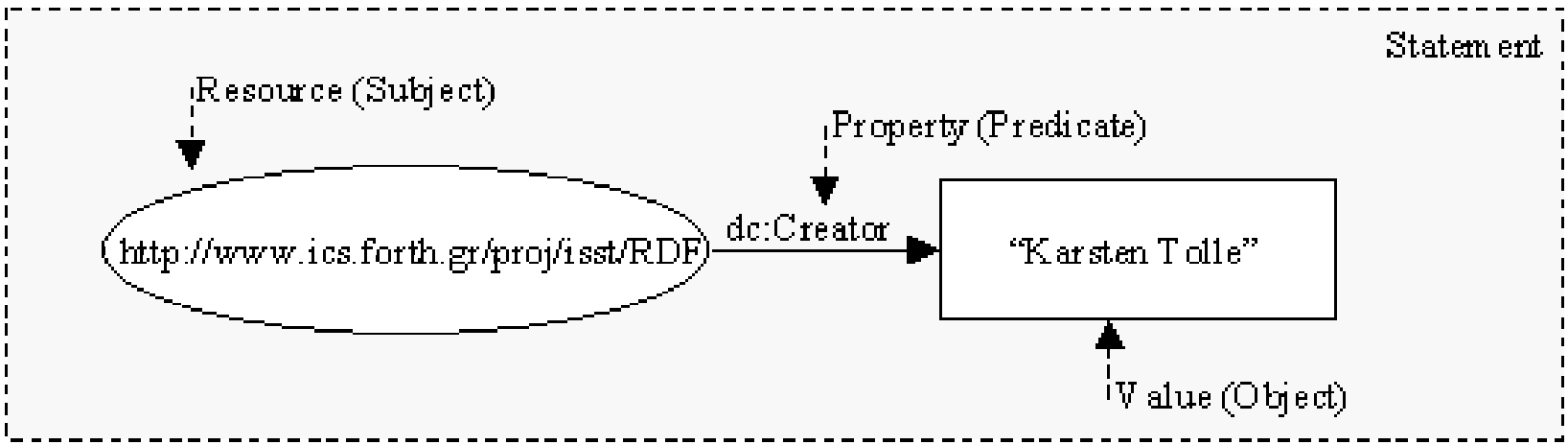
RDF Data Model



Graph Representation

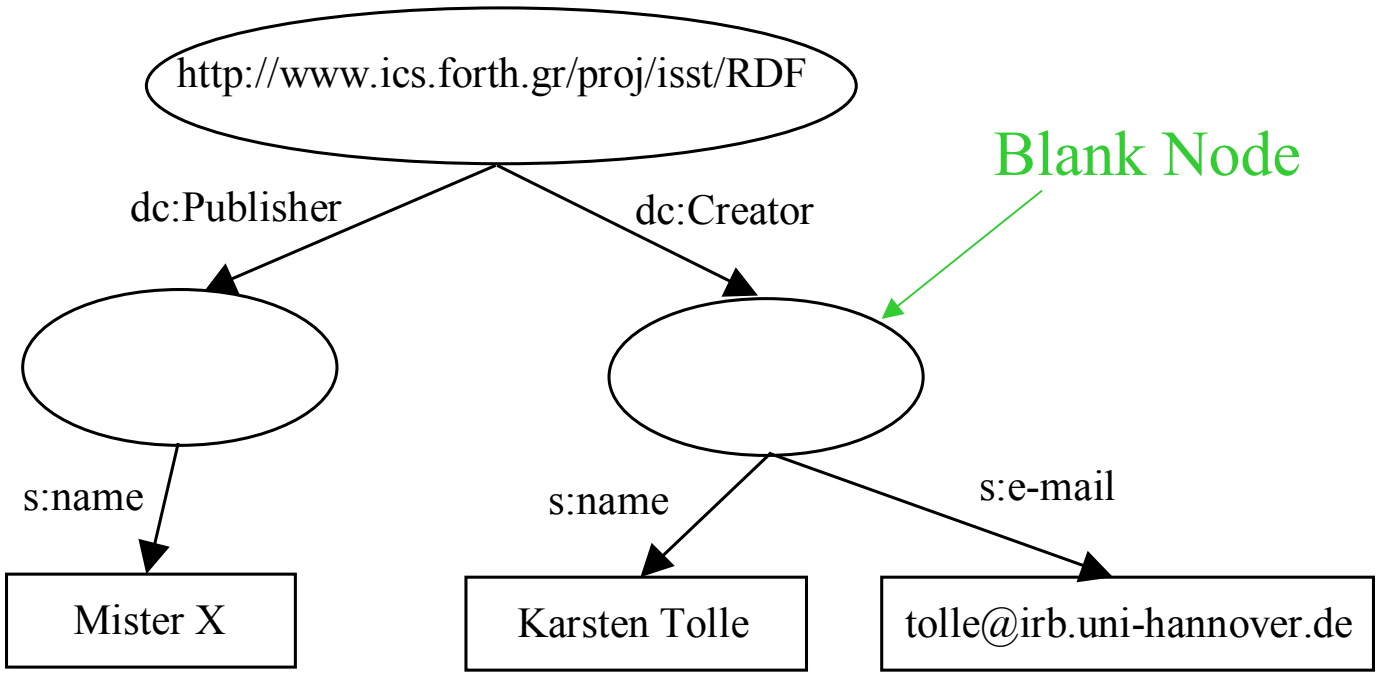


RDF Graph



Karsten Tolle is the creator of the resource
http://www.ics.forth.gr/proj/isst/RDF

RDF Graph



http://www.ics.forth.gr/proj/isst/RDF was created by an individual whose name is Karsten Tolle, email tolle@irb.uni-hannover.de and the individual whose name is Mister X published it

Representations

RDF M&S provides three representations of the RDF data model namely:

- **RDF Graph** – a syntax-neutral graphical description of the data
- **RDF 3-tuples (called triples)** – the set of statements described in triples using (s,p,o) or (p,s,o) order (equivalent to Notation 3 or N3).
- **RDF/XML** – for transportation and exchange

Note: There is a 1 – to – 1 mapping between the representations!

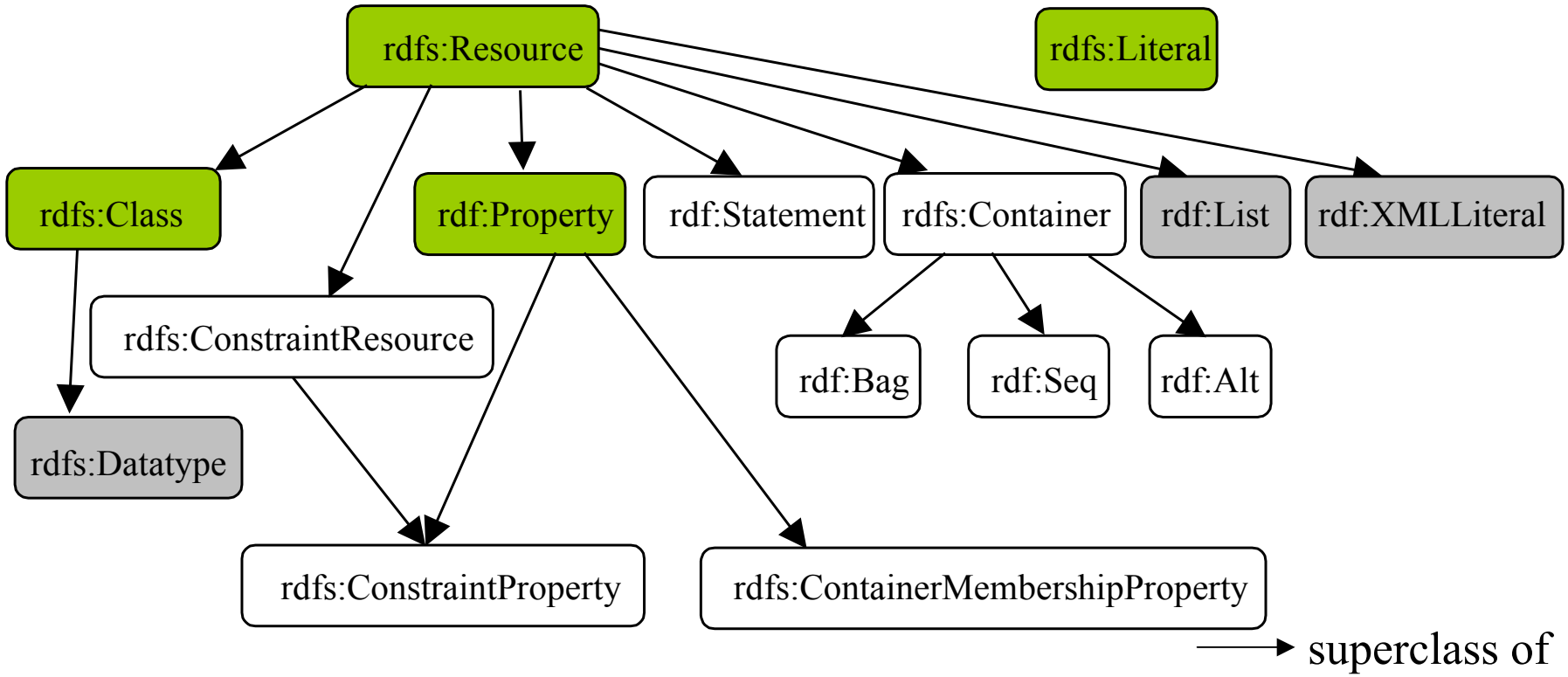
RDF/XML

```
<?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/"
         xmlns:ex="http://example.org/stuff/1.0/">
<rdf:Description rdf:about="http://www.dbis.informati.uni-
    frankfurt.de/~tolle/RDF/index.html"
    dc:title="Semantic Web and the Resource Description Framework (RDF) at DBIS">
    <ex:editor>
        <rdf:Description ex:fullName="Karsten Tolle">
            <ex:homePage rdf:resource="http://www.dbis.informati.uni-
    frankfurt.de/~tolle/"/>
        </rdf:Description>
    </ex:editor>
</rdf:Description>
</rdf:RDF>
```

RDF Schema mechanism

- Based on the basic notation of using statements, RDF defines a schema mechanism that introduces the basic constructs that can be used in the definition of RDF schemas (or vocabularies)
- An RDF schema can define for a given application or domain
  - resources
  - classes
  - properties along with some constraints
  - hierarchies for classes and properties
  - collections or containers
  - annotations to the defined vocabulary

RDF Schema mechanism (classes)



- There are two namespaces defined for RDF that can be viewed as the core vocabulary. We abbreviate them by the prefix:

  rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"

  rdfs = "http://www.w3.org/2000/01/rdf-schema#"

RDF Schema mechanism (core properties)

- **rdf:type** – instantiates a resource into a class
- **rdfs:subClassOf** – used to define a class hierarchy
- **rdfs:subPropertyOf** – used to define a property hierarchy
- **rdfs:domain** – constraints a property to be used with subject-resources of a specified class
- **rdfs:range** – constraints a property to be used with object of a specified class
- ...

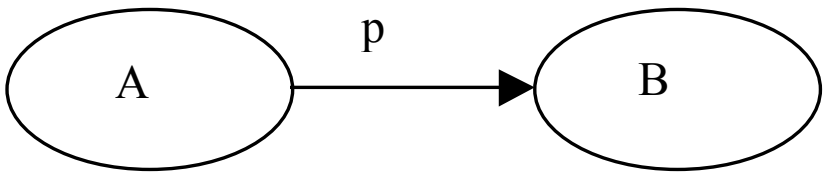
RDF/XML Schema

A part of rdfs (http://www.w3.org/2000/01/rdf-schema):

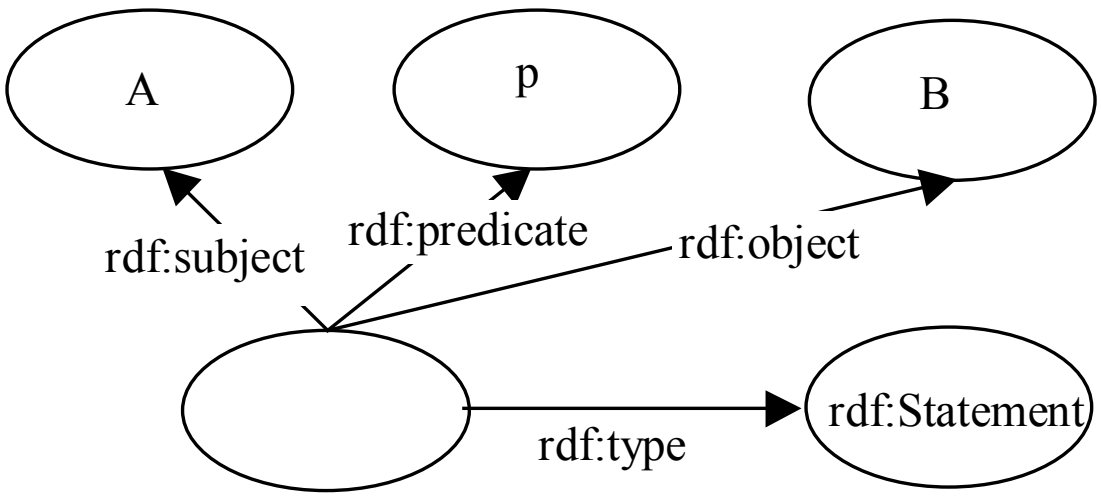
```
...
<rdf:Property rdf:about="http://www.w3.org/1999/02/22-rdf-syntax-ns#type">
        <rdfs:isDefinedBy rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#" />
        <rdfs:label>type</rdfs:label>
        <rdfs:comment>The subject is an instance of a class.</rdfs:comment>
        <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class" />
        <rdfs:domain rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource" />
</rdf:Property>

<rdfs:Class rdf:about="http://www.w3.org/2000/01/rdf-schema#Class">
        <rdfs:isDefinedBy rdf:resource="http://www.w3.org/2000/01/rdf-schema#" />
        <rdfs:label>Class</rdfs:label>
        <rdfs:comment>The class of classes.</rdfs:comment>
        <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource" />
</rdfs:Class>
...
```

Reified Statements

The original statement:

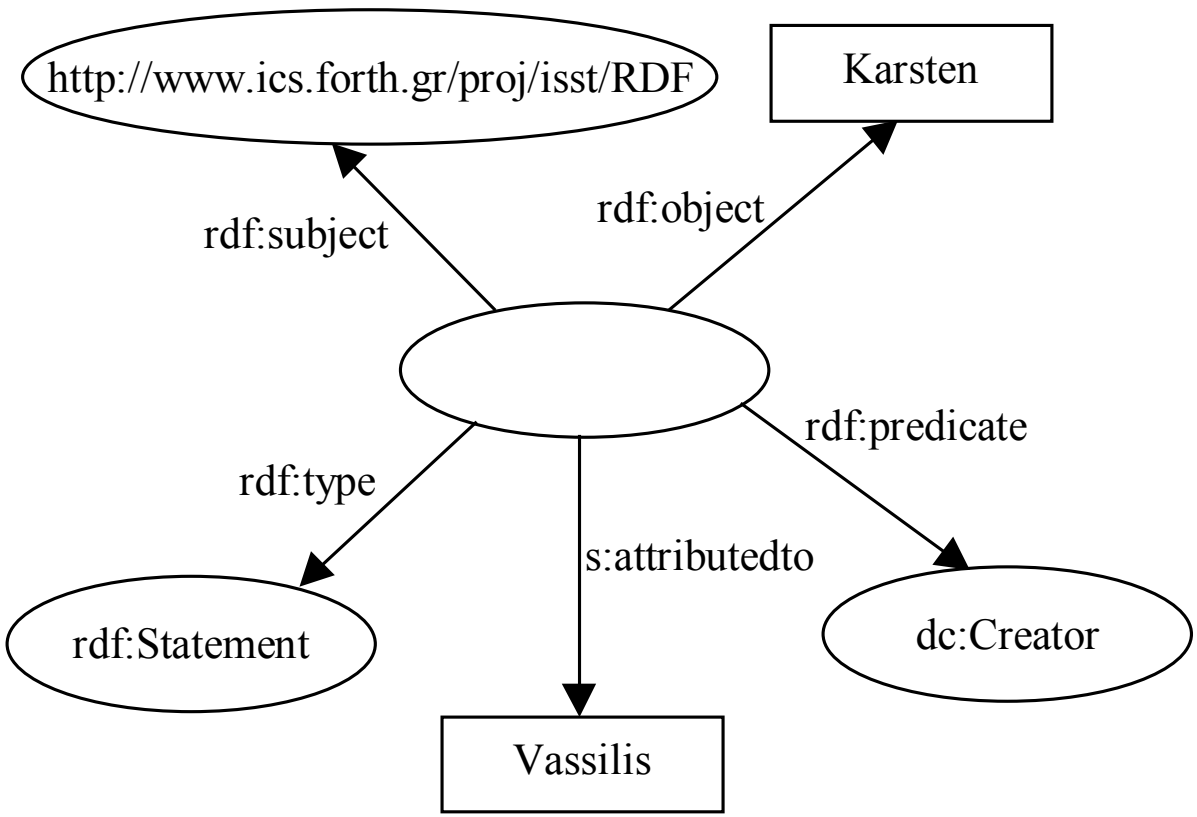


The model of the statement:



Reified Statements

Vassilis says: "Karsten is the creator of the resource http://www.ics.forth.gr/proj/isst/RDF."

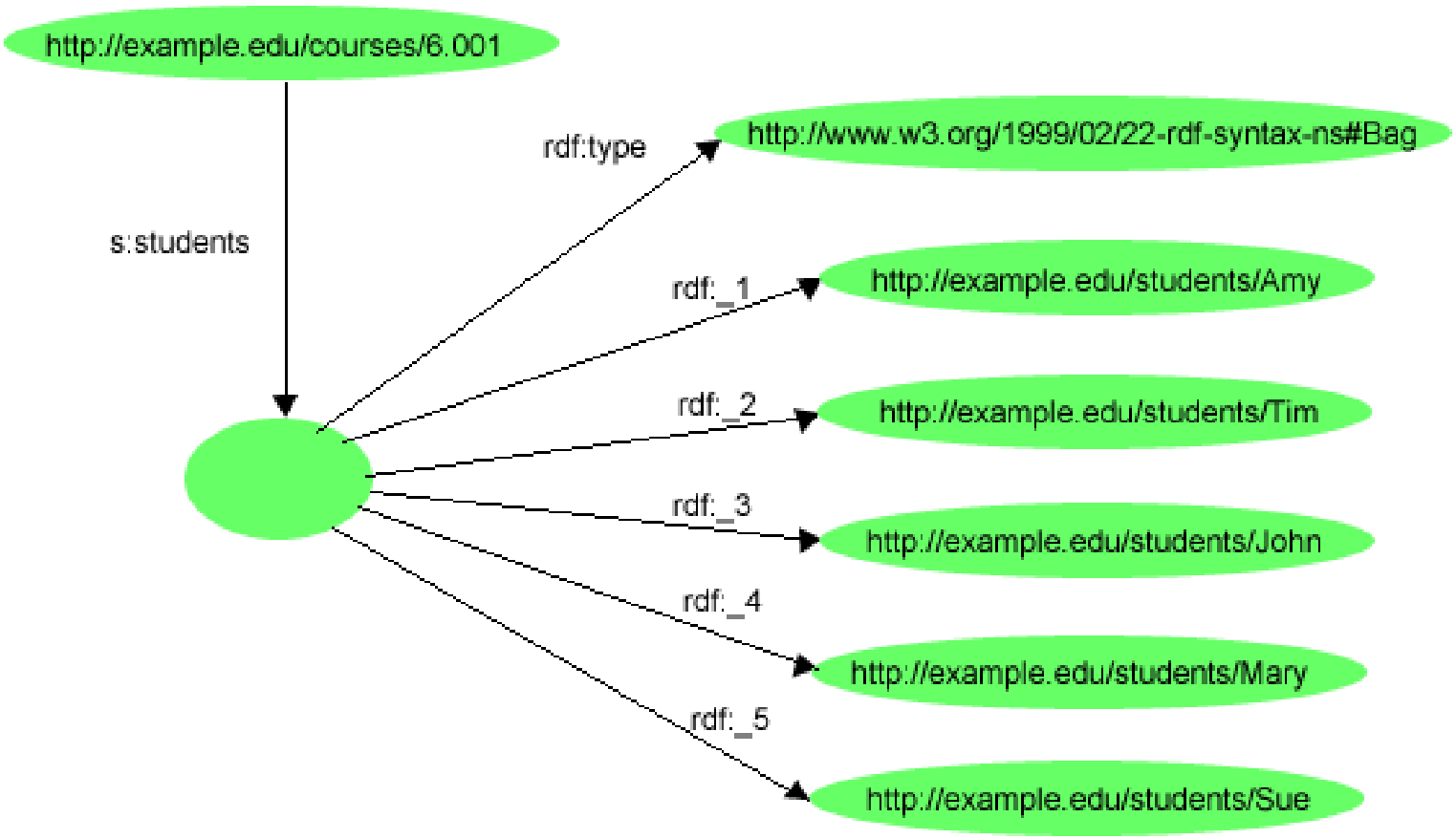


Container

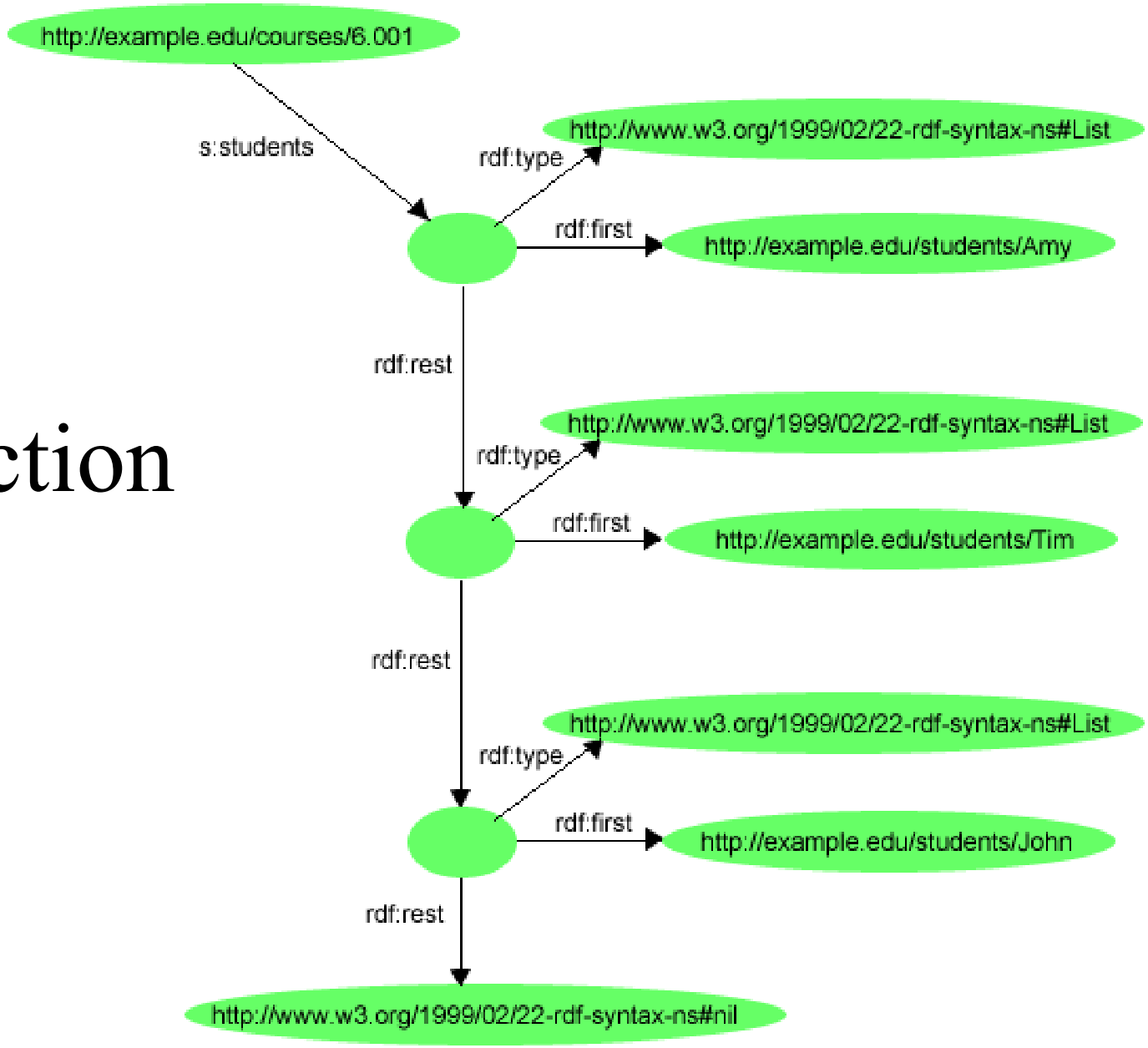
Three types:

- **rdf:Bag** – a unordered set {a, b, c, d}
- **rdf:Seq** – an ordered set [a, b, a, c]
- **rdf:Alt** – an alternative (rdf:_1 is default)

Container


http://example.edu/courses/6.001
s:students
rdf:type
http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag
rdf:_1
http://example.edu/students/Amy
rdf:_2
http://example.edu/students/Tim
rdf:_3
http://example.edu/students/John
rdf:_4
http://example.edu/students/Mary
rdf:_5
http://example.edu/students/Sue

Collection

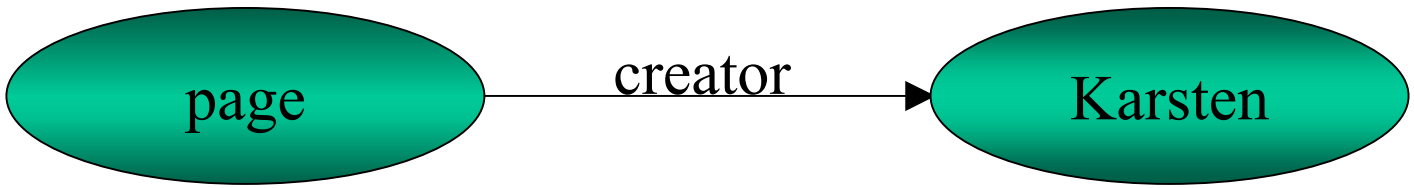

http://example.edu/courses/6.001
s:students
http://www.w3.org/1999/02/22-rdf-syntax-ns#List
rdf:type
rdf:first
http://example.edu/students/Amy
rdf:rest
http://www.w3.org/1999/02/22-rdf-syntax-ns#List
rdf:type
rdf:first
http://example.edu/students/Tim
rdf:rest
http://www.w3.org/1999/02/22-rdf-syntax-ns#List
rdf:type
rdf:first
http://example.edu/students/John
rdf:rest
http://www.w3.org/1999/02/22-rdf-syntax-ns#nil

Container vs Collection

- With a collection you can ensure (when using correct structure) that all elements are contained.
- Containers have a semantic, collections not.

Problem: The creator of an RDF file needs to choose between these two constructs.

RDF vs XML



How to represent this information in XML?

```
<document>
        <uri>page</uri>
        <creator>Karsten</creator>
</document>
```

```
<creator ID="Karsten">
        <document uri="page"/>
</creator>
```

```
<uri ID="page" type="document">
        <creator ID="Karsten"/>
</uri>
```

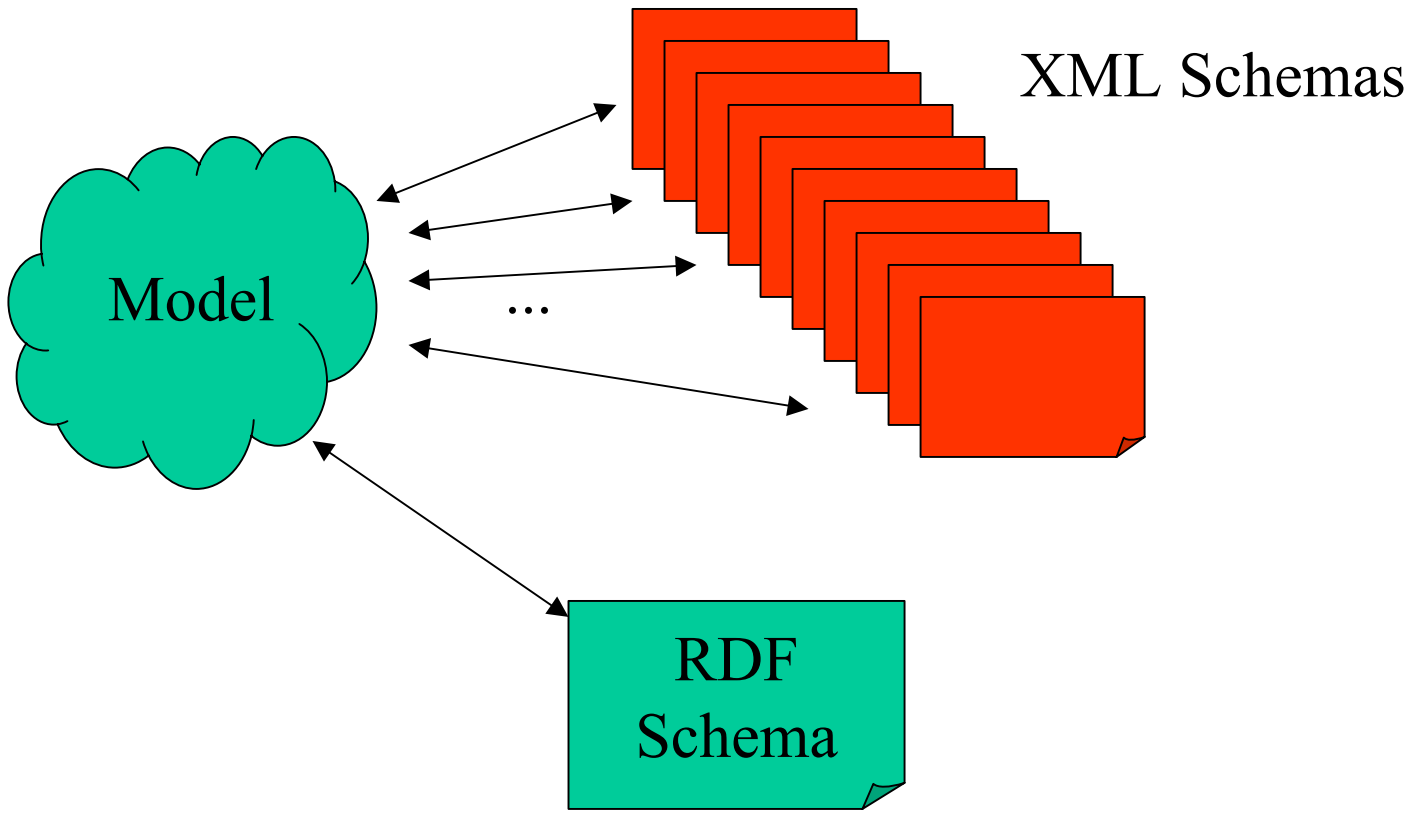
```
<document uri="page" creator="Karsten" />
```

RDF vs XML

```
<a>
        <b attr1="344">
                <c>ttttttt</c>
                <d>uuuuu</d>
        </b>
        <e attr2="vvvvvvv"/>
</a>
```

Without knowing the schema you only have the structure!

RDF vs XML



RDF vs XML

In XML you have

- A many to one mapping.
- You need to know the schema.
- Limited to a tree.
- No inference defined.
- No default constructs like in RDF
  like classes and hierarchies, containers, ...)
- ...

RDF Limitations

- No equivalence
  - needed to resolve synonymy
- No distinction between schemas and data
  - anyone can add to schema definitions
- Weak schema versioning mechanism
  - schemas will evolve over time

Extensions for RDF

- OWL – Web Ontology Language
  - OWL lite
  - OWL DL (description logics)
  - OWL full
- Namespace:
  xmlns:owl ="http://www.w3.org/2002/07/owl#"

OWL Lite (some examples)

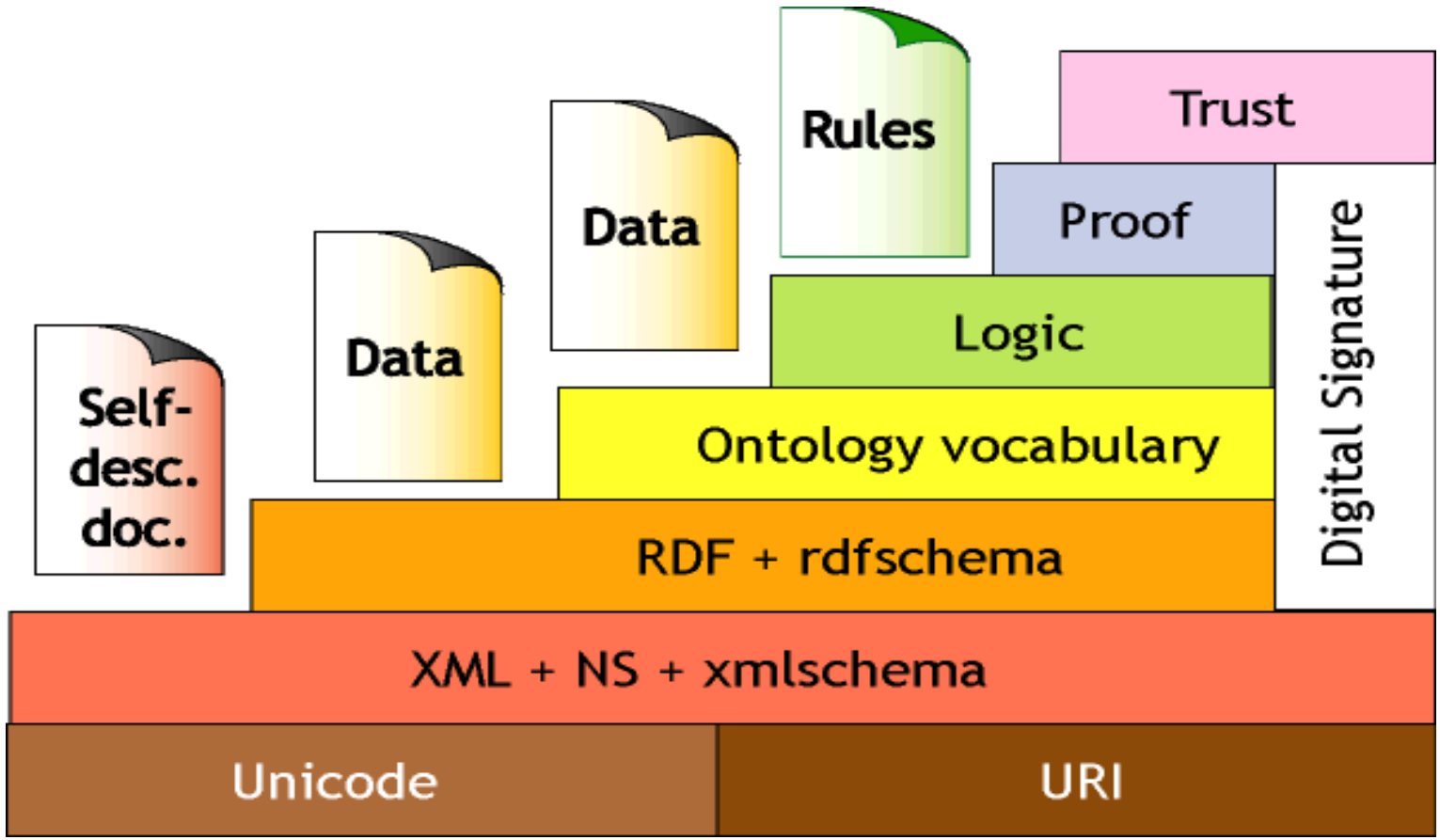
(In)Equality:

- *equivalentClass*
- *equivalentProperty*
- *sameAs*
- *differentFrom*
- *allDifferent*

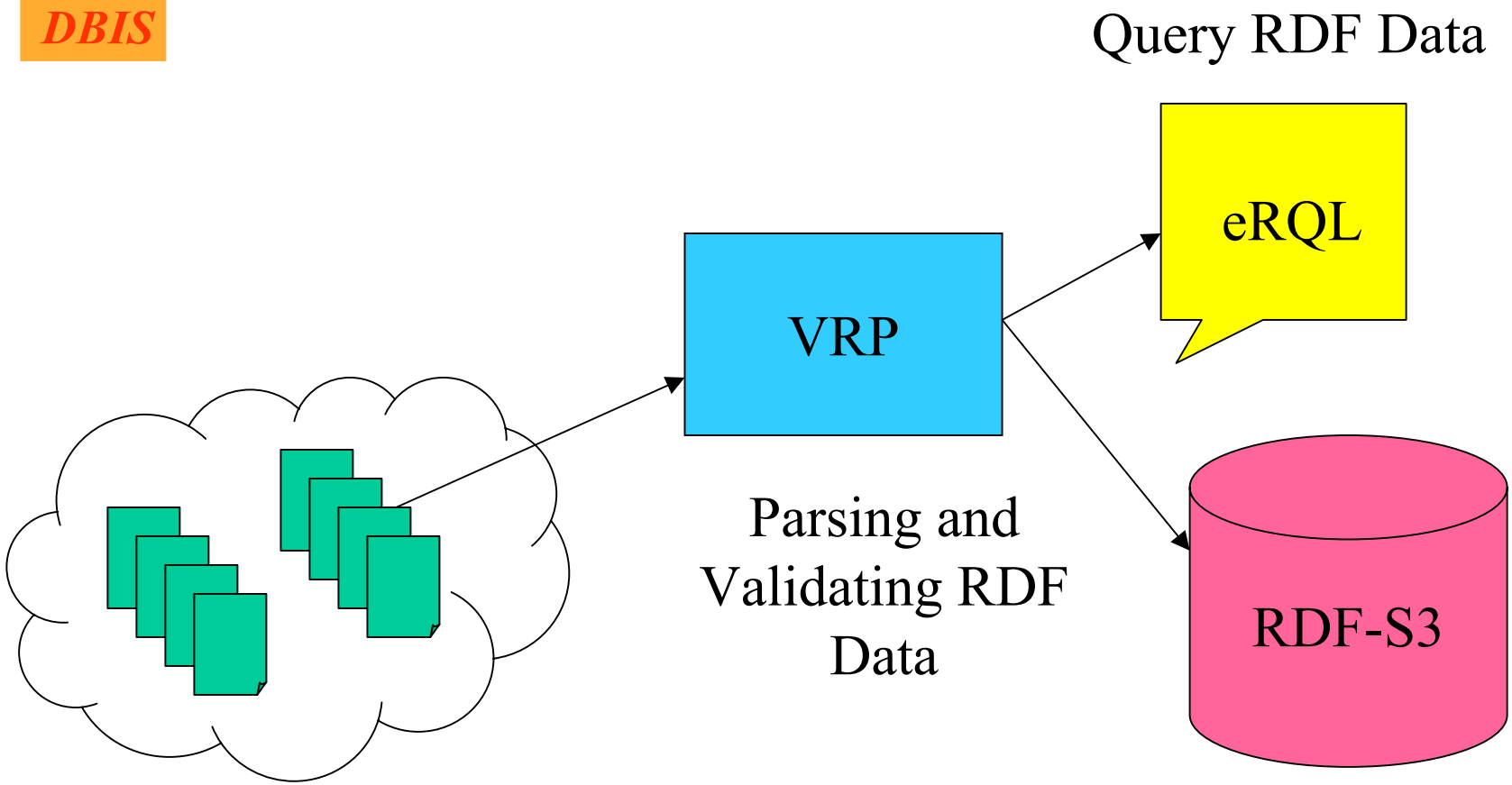
Property Characteristics:

- *inverseOf*
- *TransitiveProperty*
- *SymmetricProperty*
- *FunctionalProperty*
- *InverseFunctionalProperty*

Layers for the Semantic Web



Research at DBIS



Validating RDF Parser

FORTH
Institute of Computer Science

ICS-VRP

Main Features

- Supports
  - Embedded RDF in HTML or XML
  - XML Schema Data Types
  - Full Unicode
- Enables customisation of Semantic Validation Constraints
- Inference on Resource Types & Properties
- Provides options for debugging & statistics
- Easy to use as a standalone application (GUI)
- Easy to integrate with other applications

RDF-Source Storage System

start

RDF-S3

Main Features

- stores the relationship of the RDF statements to their source
- supports validation options by using VRP
- automatic inclusion of missing (rdfs:Class & rdf:Property) type information
- only need JDBC-compliant RDBMS (supporting standard SQL)
- easy to use GUI (from the look and feel similar to RSSDB)
- user friendly by using stored preferences

Easy RQL

eRQL

Main Design Goals

- **Simplicity** - A user should be enabled to make simple queries even without any preliminary knowledge.
- **Schema independence** - Queries should be possible without knowing the underlying RDF schema for the given data.
- **Power** - Powerful queries should be generated by concatenating more simple ones.
- **Domain independence** - The query language should be independent of a specific domain.

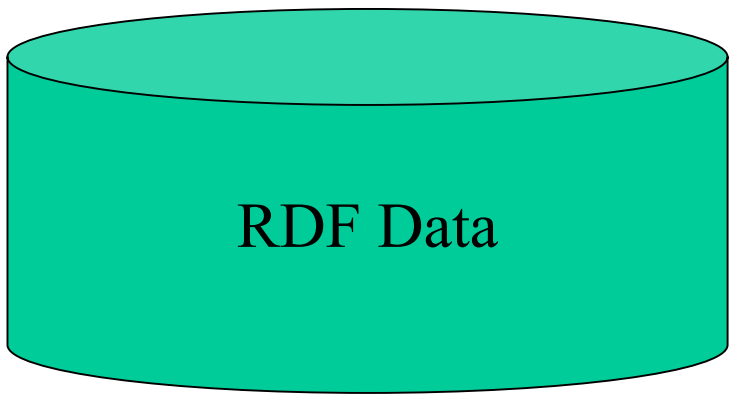
Research Areas

- Schema Evolution

What to do with the data expressed in the old schema?

Research Areas

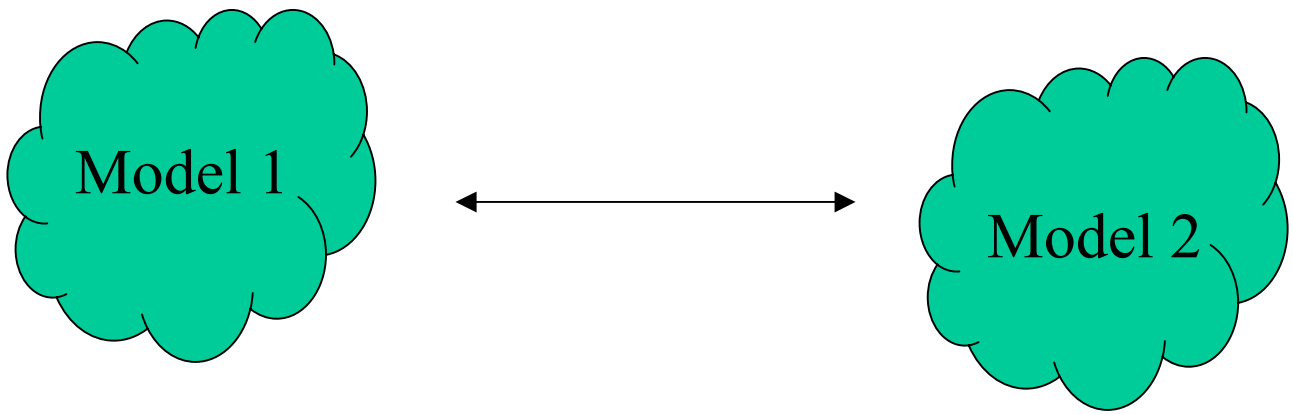
- Changes in a RDF repository (DB)



Are the data still valid after a change?
What happens if a schema is changed? (see before)

Research Areas

- Schema matching and mapping



What parts are equal and how to map them?