



Forschungszentrum L3S

Research Center

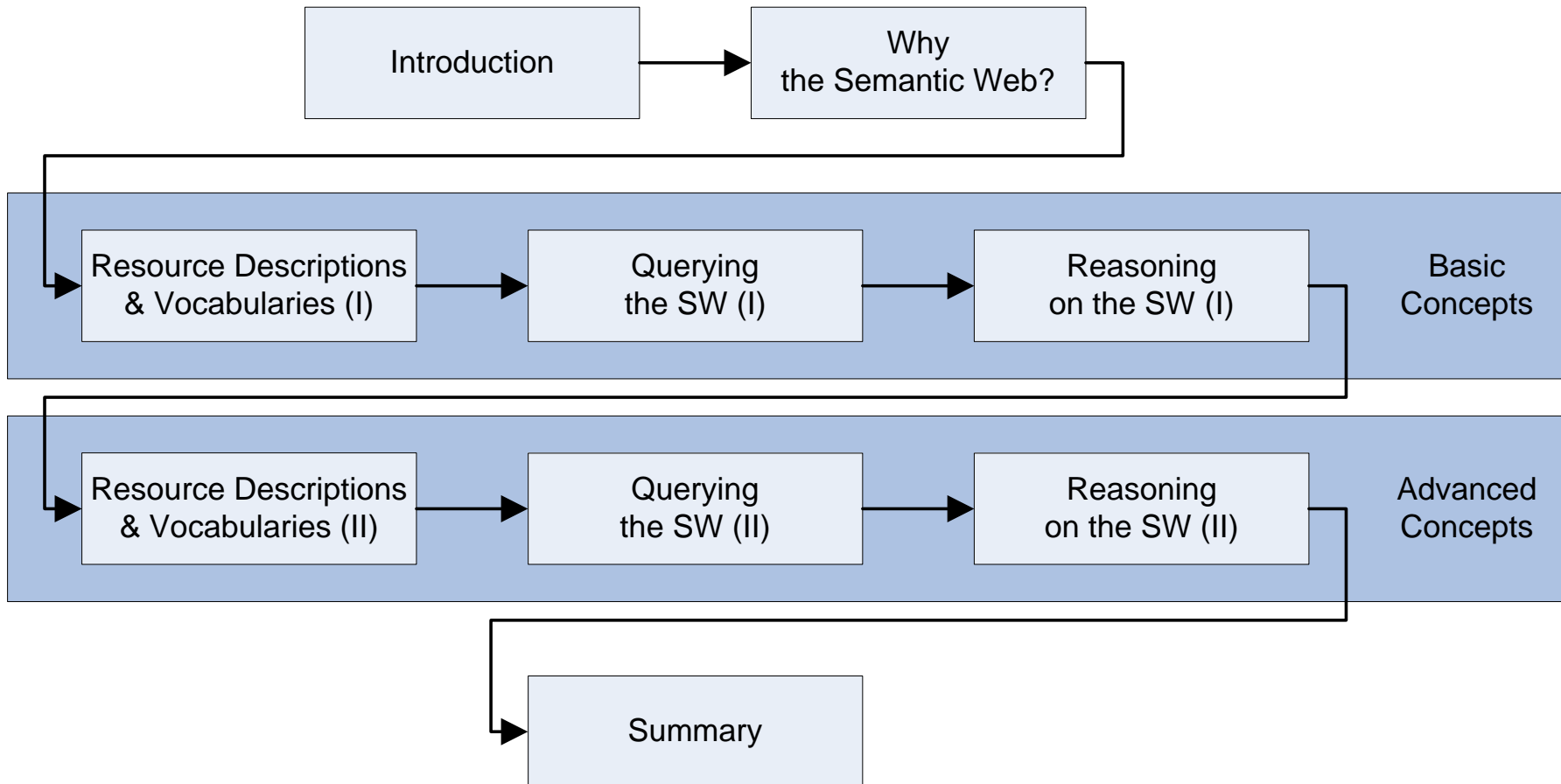
Introduction to the Semantic Web

Daniel Olmedilla and Wolf Siberski
L3s Research Center / Hannover University

TENCompetence Winter School
Innsbruck, 24th January 2007

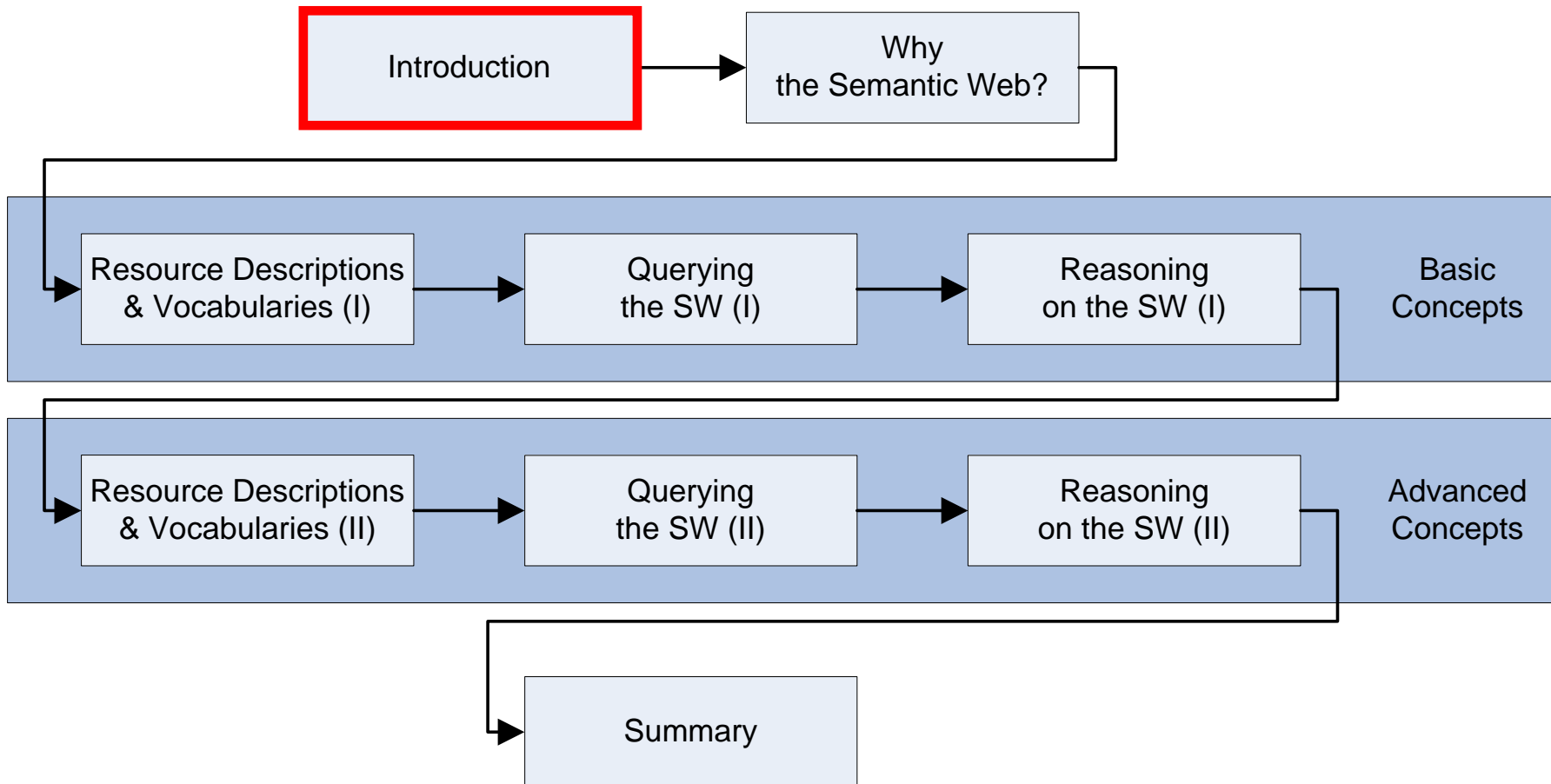
Outline

Lecture overview



Outline

Lecture overview



Introduction

Why this tutorial?



- **Lot of noise about the Semantic Web**
 - Lot of relevant papers and work on Semantic Web in last years
- **Many concepts, languages and frameworks**
- **Techniques and tools can be used in the context of lifelong learning and competence development**

Introduction

About this tutorial (I)



This tutorial is intended to provide

- reasons that motivated Semantic Web Research
- a basic understanding of its ideas and models
- an introduction to querying on the Semantic Web
- a basic introduction to reasoning techniques



And also important

- This is not
 - a conference presentation
 - a monologue
- Each module builds on concepts from previous modules
- We provide exercises to strength understanding
- You are also encouraged to interrupt us and

ASK Questions

whenever you need it

Introduction

About this tutorial (& III)



Slides are wordy so they can be easily understood offline after the tutorial

More definitions and references are available in notes and hidden slides

Tutorial is available from:

http://www.L3S.de/~olmedilla/events/2007/TENC-WS/20070124_TENC_WS.pdf



- Protégé should have been installed
- Download the OWL ontology available at
http://www.l3s.de/~olmedilla/events/2007/TENCWS_Lecture.html
- RACERPro reasoner available during the lecture at
<http://webbase.L3S.uni-hannover.de:8080>

Introduction

Configuration of Tools (& II)



OWL Preferences

General | Visibility | Datatypes | Searching | Encoding | Tabs | Tests

User Interface Features

- Drag and Drop
- Constraint checking (red borders) at edit time
- Allow the creation of external resources (untyped URIs)

Reasoning

Reasoner URL:

Class Display Format

- Manchester OWL Syntax
- Compact OWL Class Display
- DL Syntax Class Display

Icon Style:

Protege Features

- Import Protege metadata ontology
- Support user-defined XML Schema datatypes (numeric ranges)

Language Profile

- Use standard profile:
- Use custom profile:

RDF/XML Writer Settings

- Default Jena writer
- Experimental native writer
- Use XML entities
- Sort resources alphabetically

Forms

- Save forms to .forms files

Close

<http://webbase.l3s.uni-hannover.de:8080>
<http://localhost:8080>

Introduction

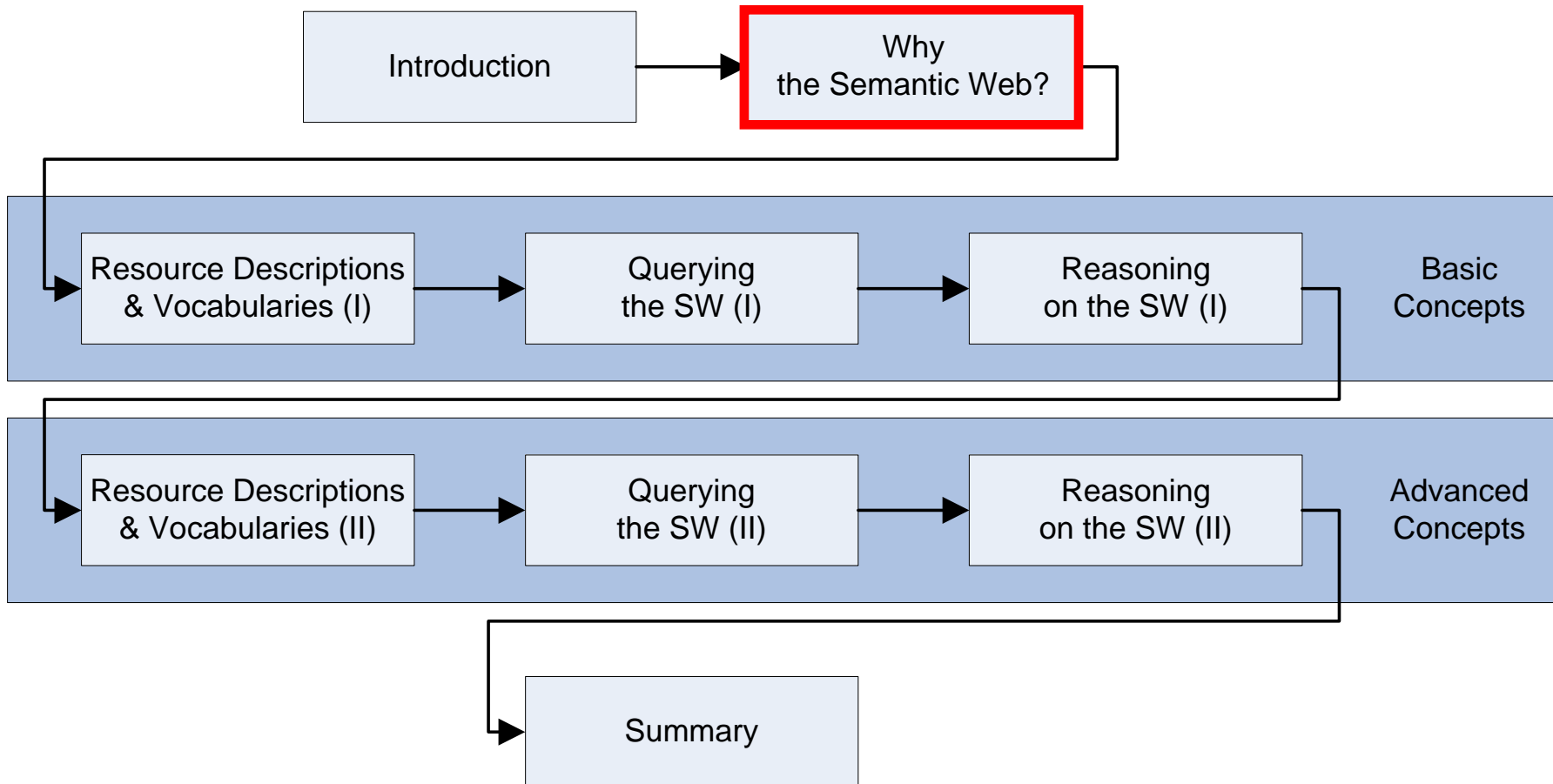
Querying the audience



- **Technical Background?**
- **Already working with Semantic Web technologies?**
- **Experience with Databases?**
- **None of the previous?**

Outline

Lecture overview



Why the Semantic Web?

HTML: in your browser



TENCompete

Innsbruck, Austria

Topics

- Educational Principles
- Knowledge Management
- Education Process Modeling
- Learning Design
- Competence Development
- Personal Learning Environments
- Simulation & Game Based Learning
- Semantic Web
- Social Software
- Open Source & Open Standards
- Software Engineering with UML
- Web Services

Lecturers

- Albert Angehrn, INSEAD, France
- Boyan Bontchev, Sofia University, Bulgaria
- Alexandar Dimov, Sofia University, Bulgaria
- Dai Griffiths, University of Bolton, United Kingdom

Forschungszentrum L3S

Why the Semantic Web?

HTML: Markup



`<h2> Topics </h2>`

`<p>`

Educational Principles `
`

Knowledge Management `
`

Education Process Modeling `
`

Learning Design `
`

Competence Development `
`

...

`</p>`

`<h2> Lecturers </h2>`

`<p>`

Albert Angehrn, INSEAD, France `
`

Boyan Bontchev, Sofia University, Bulgaria `
`

Alexandar Dimov, Sofia University, Bulgaria `
`

Dai Griffiths, University of Bolton, United Kingdom `
`

...

`</p>`

**Markup for
presentation only**

Why the Semantic Web?

HTML: Limitations



HTML deals only with formatting of data

It does not provide information about the data it contains

Query engines do a great job but queries like

- Give me the list of subjects that the winter school will deal with
- Return the affiliations of the lecturers in the winter school

are not possible on the current Web

Search on current Web is based on syntactic matching

Why the Semantic Web?

Current Web



■ Downloadable Resources:

- identified by URL's
- untyped

■ Links:

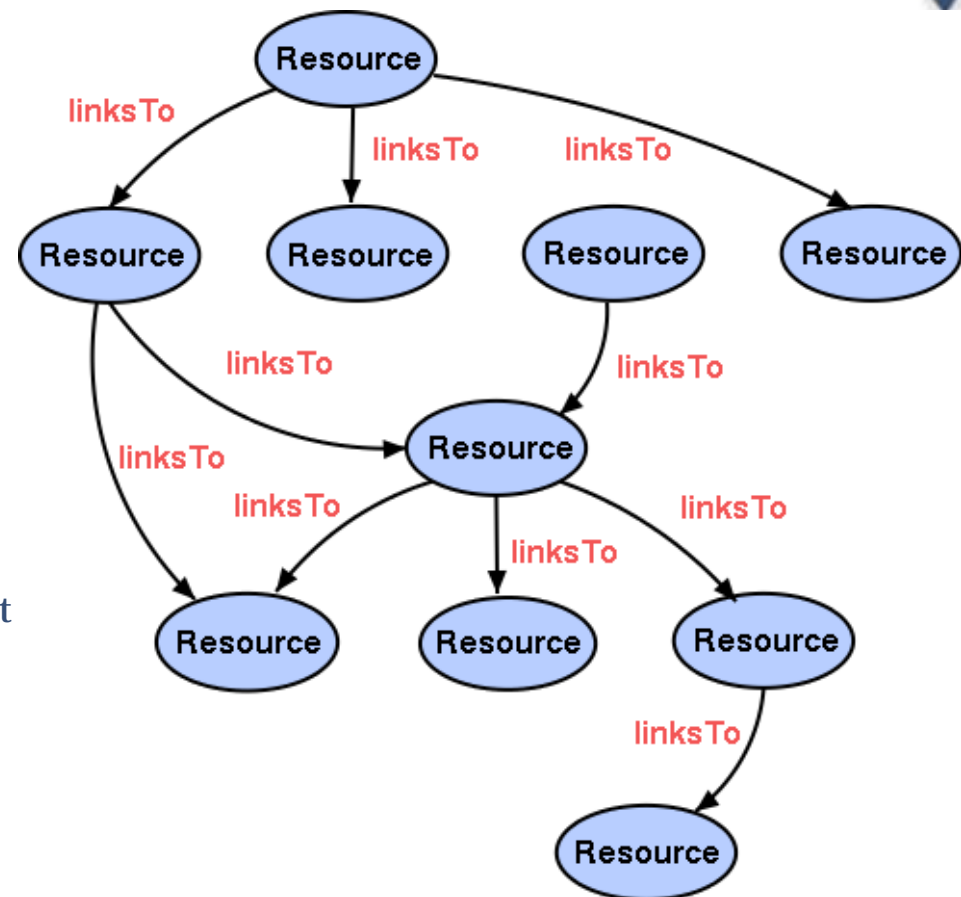
- href, src, ...
- limited, non-descriptive

■ User:

- Exciting world
 - semantics of the resource, however, gleaned from content

■ Machine processable:

- Very little information available
 - significance of the links only evident from the context around the anchor.



[Eric Miller. Weaving Meaning : An Overview of The Semantic Web. 2003]

Why the Semantic Web?

Semantic Web Definition



“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**.”

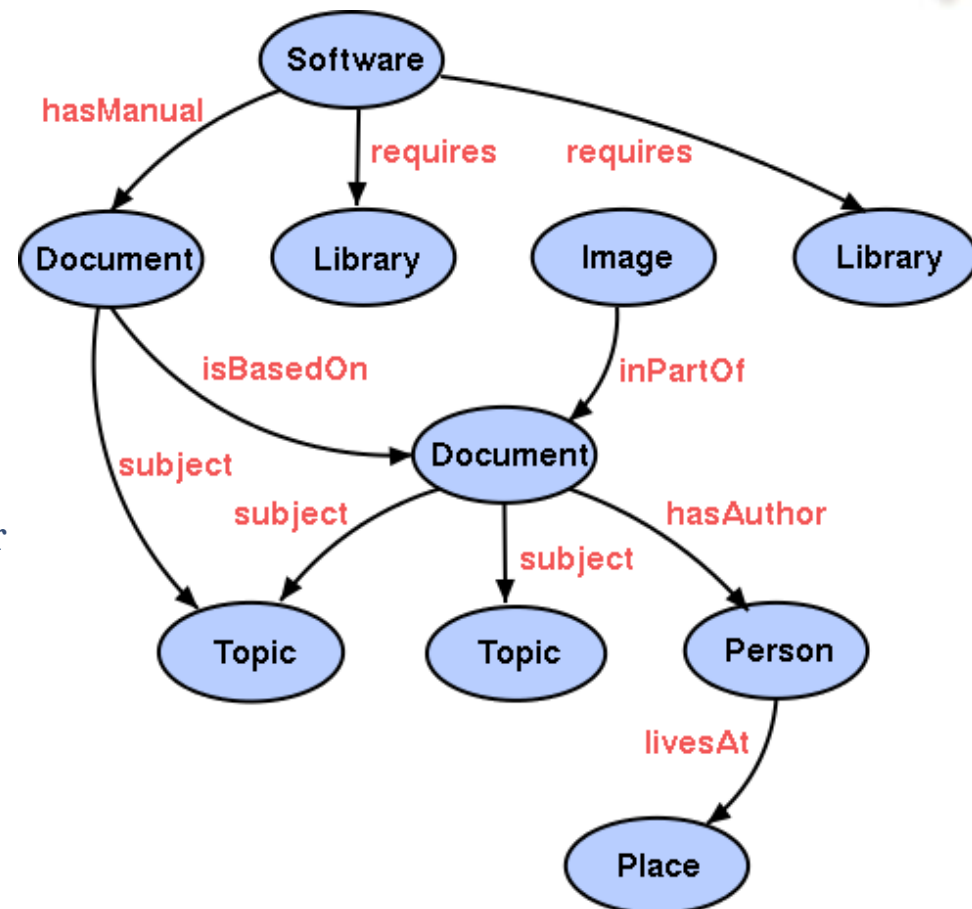
Tim Berners-Lee, James Hendler, Ora Lassila
The Semantic Web, Scientific American, May 17, 2001

Why the Semantic Web?

The Semantic Web



- Resources (any resource):
 - Globally Identified by URI's
 - Extensible
 - Relational
 - Links:
 - Identified by URI's
 - Extensible
 - Relational
-
- User:
 - Even more exciting world, richer user experience
 - Machine:
 - More processable information is available (Data Web)
 - Computers and people:
 - Work, learn and exchange knowledge effectively



[Eric Miller. Weaving Meaning : An Overview of The Semantic Web. 2003]

Why the Semantic Web?

Basic requirements



Information contained in the current Web is not sufficient

We need

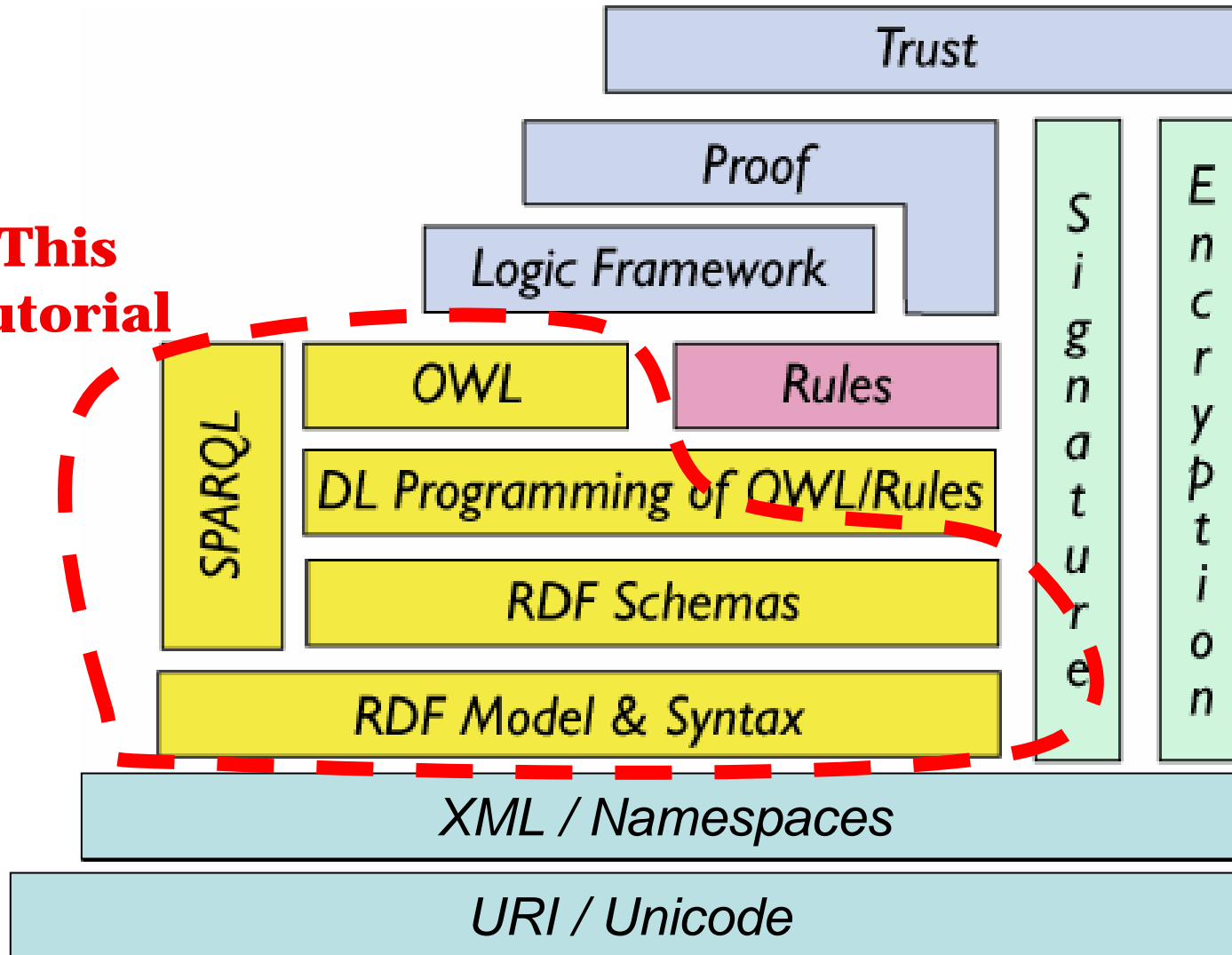
- A way to refer uniquely to resources
- Extra information about them
 - annotations/metadata
- With a flexible model
 - many different sources/databases on the Web
- With well defined and commonly agreed concepts
 - clear semantics
 - not ambiguous

Why the Semantic Web?

The Semantic Web Stack



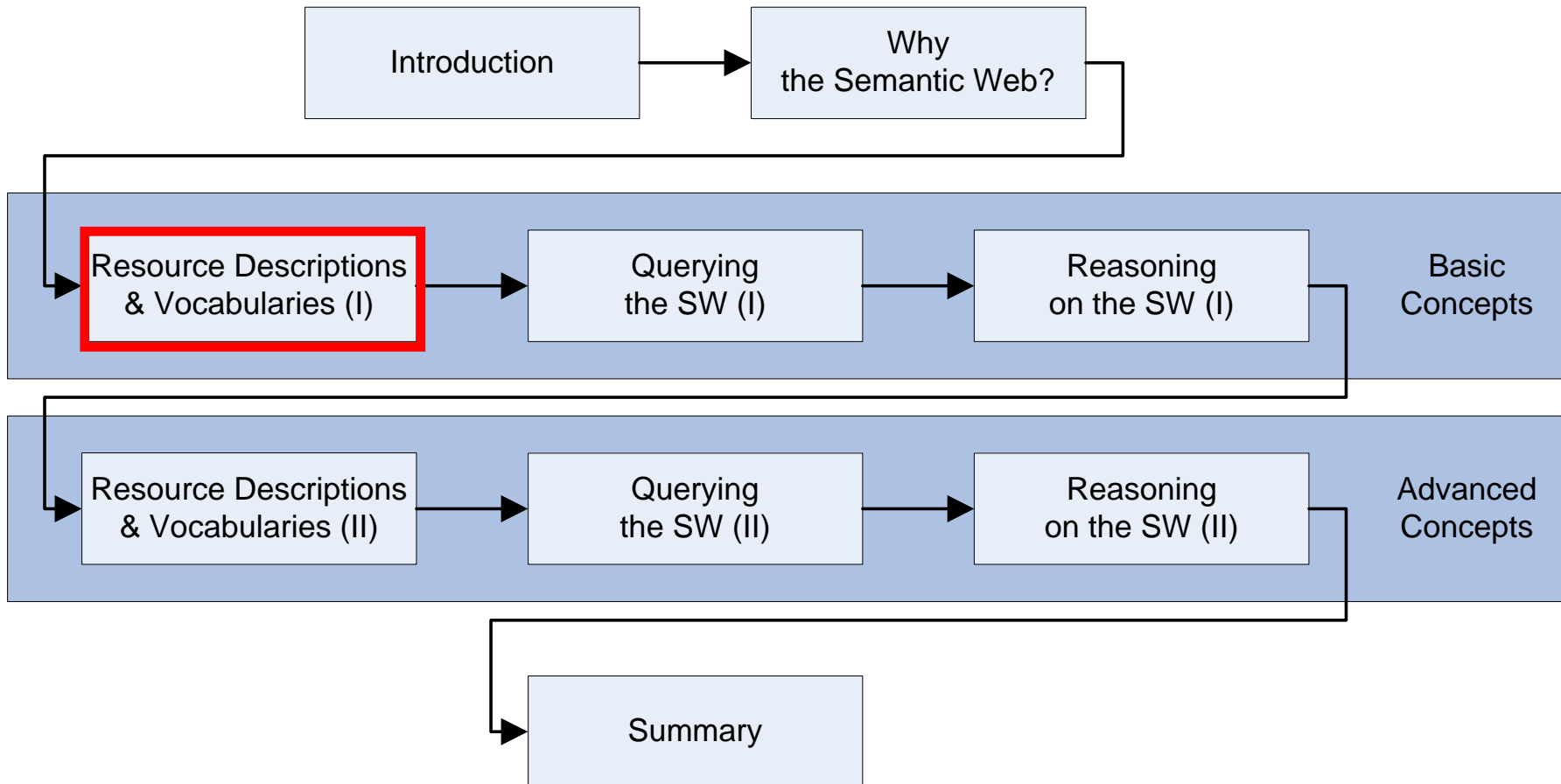
**This
Tutorial**



Forschungszentrum L3S

Outline

Lecture overview



Resource Descriptions and Vocabularies I

Adding annotations to resources



- We need to provide annotations about resources
- We will create annotations about resources as triple statements

Examples

- **TENCompetence Winter School** has an **organizator** whose value is **Milos Kravcik**
- **TENCompetence Winter School** has a **language** whose value is **English**



< **Subject** - **Predicate** - **Object** >

- **Subject:** what is described
- **Predicate:** relation between two resources
- **Object:** value of the property (what the statement says)

Example

Subject: TENCompetence Winter School
Predicate: has an organizer
Object: Milos Kravcik

Resource Descriptions and Vocabularies I

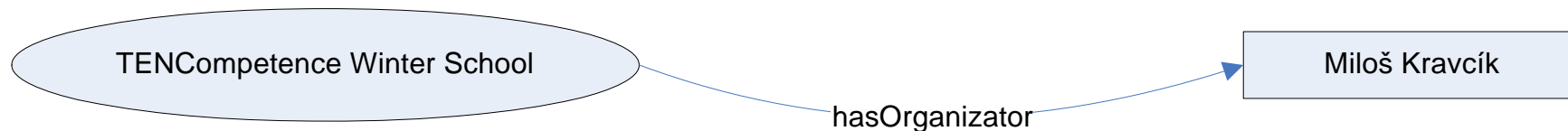
Triple as a graph



- Statements of resources are represented as a graph
 - Not bound to specific DB schema
 - More flexible

Example

Subject: **TENCompetence Winter School**
Predicate: has an **organizer**
Object: **Milos Kravcik**



Resource Descriptions and Vocabularies I

Writing triples about a person



“John Smith” has a name with value “John Smith”

“J. Smith” has a full name with value “John Smith”

“John Smith” works in “company XYZ”

“J. Smith” works in “project TENCompetence”

“John Smith” has an address with value “Market St.”

“Variable X” has an address with value “0xF32AC43”

“John Smith” has a country of birth with value “U. K.”

“J. Smith” has a country with value “United Kingdom”

Same person?
Only one John Smith?

Ambiguity?

Resource Descriptions and Vocabularies I

Requirements extracted from previous example



Identifiers to uniquely identify resources

- Subject
- Property
- Object

in order to allow that

- same resources being annotated are globally identified
- properties are given a non-ambiguous meaning

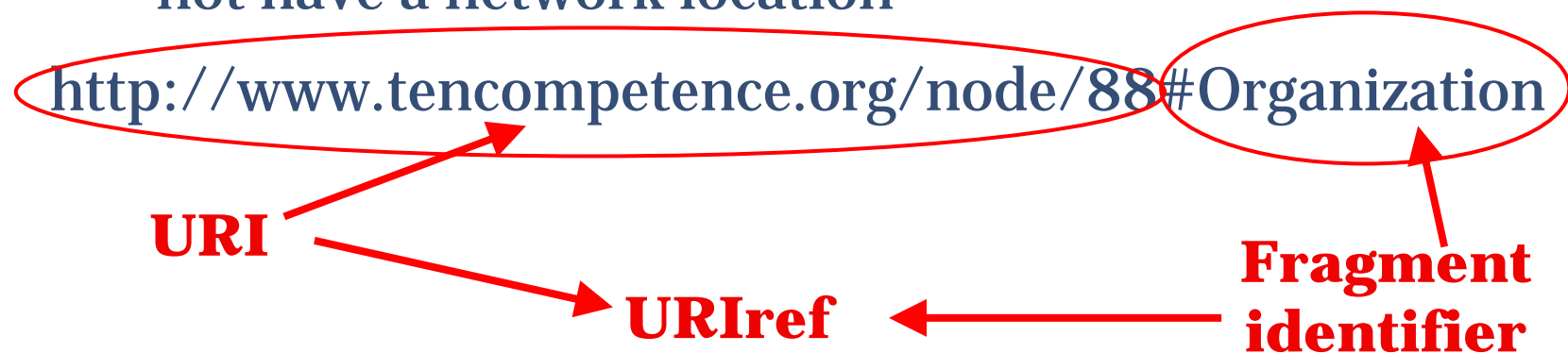
Resource Descriptions and Vocabularies I

Uniform Resource Identifier (URI)



A way to refer uniquely to resources

- URL identify things that have a network location (downloadable)
- URI identifies anything identifiable, even if it does not have a network location



Namespace: abbreviation for an URI

- tenc:Organization
- “tenc:” = <http://www.tencompetence.org/node/88#>

Resource Descriptions and Vocabularies I

Resource Description Framework (RDF)



RDF provides a common framework for metadata of web resources

- Thought to be processed by computers, not only by humans

Metadata: data about data

- Creator, date of modification, comment, etc.

Resource: anything that can be identified

- E.g. persons, prices, cars, events, ...

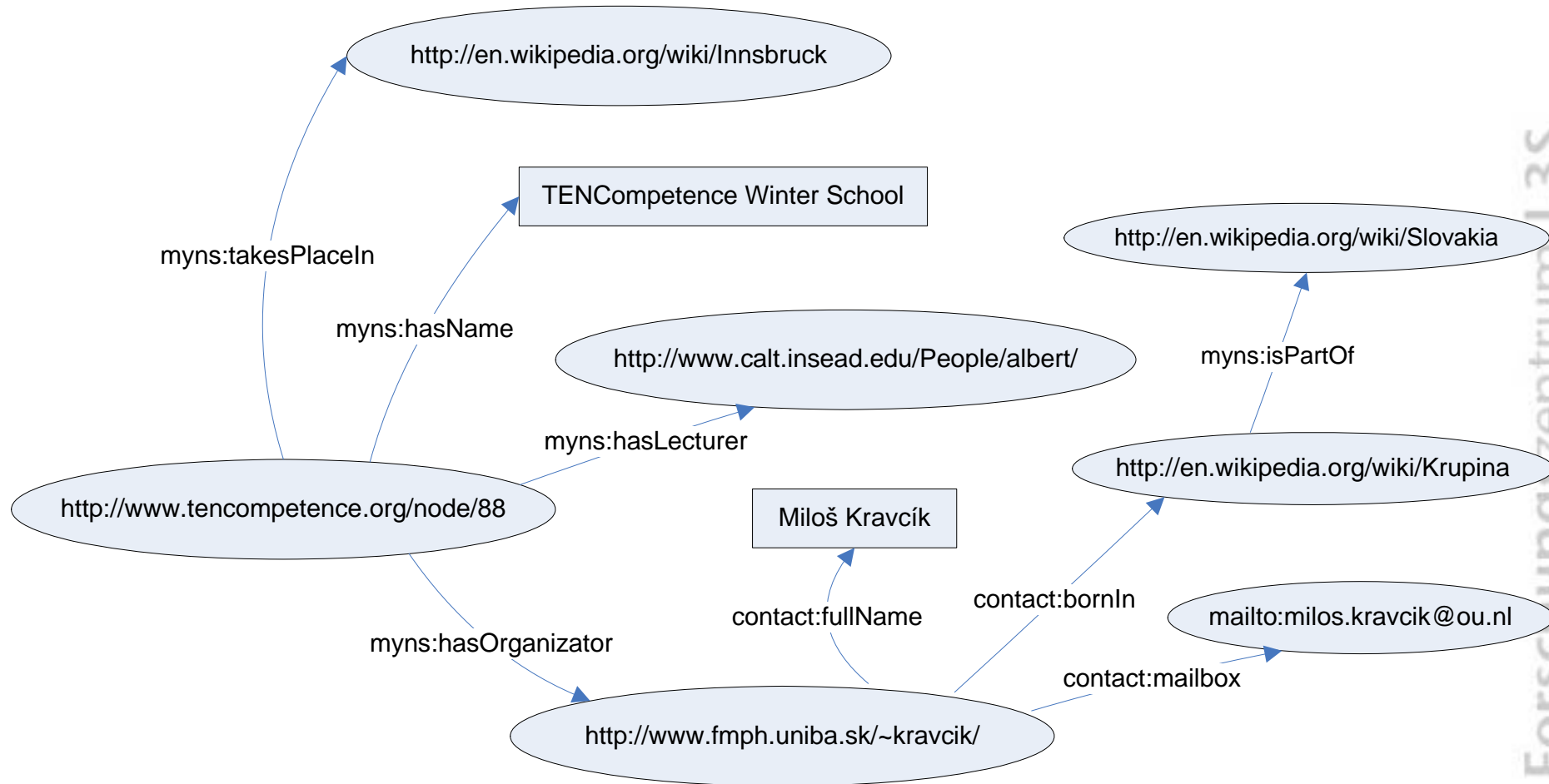


RDF statements model nodes and arcs in a graph

- **Subject**
 - Uniquely identified with an URIref
- **Predicate**
 - Uniquely identified with an URIref
 - Predicates are also resources and therefore we can annotate them
- **Object**
 - Object can be an URIref or a constant value

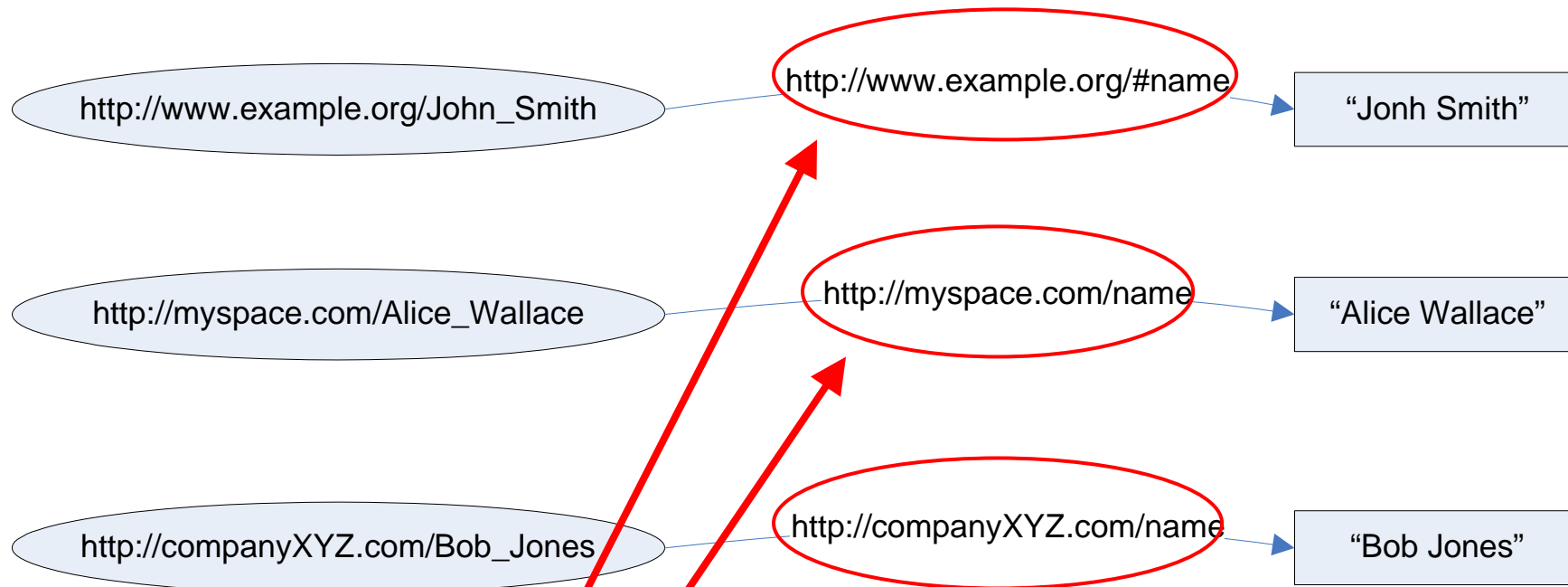
Resource Descriptions and Vocabularies I

Example Graph



Resource Descriptions and Vocabularies (I)

Annotations with URIs



**Same meaning,
Different URIrefs**



RDF does not describe the meaning, it simply states that each URIref has a meaning

There is a need for common vocabularies

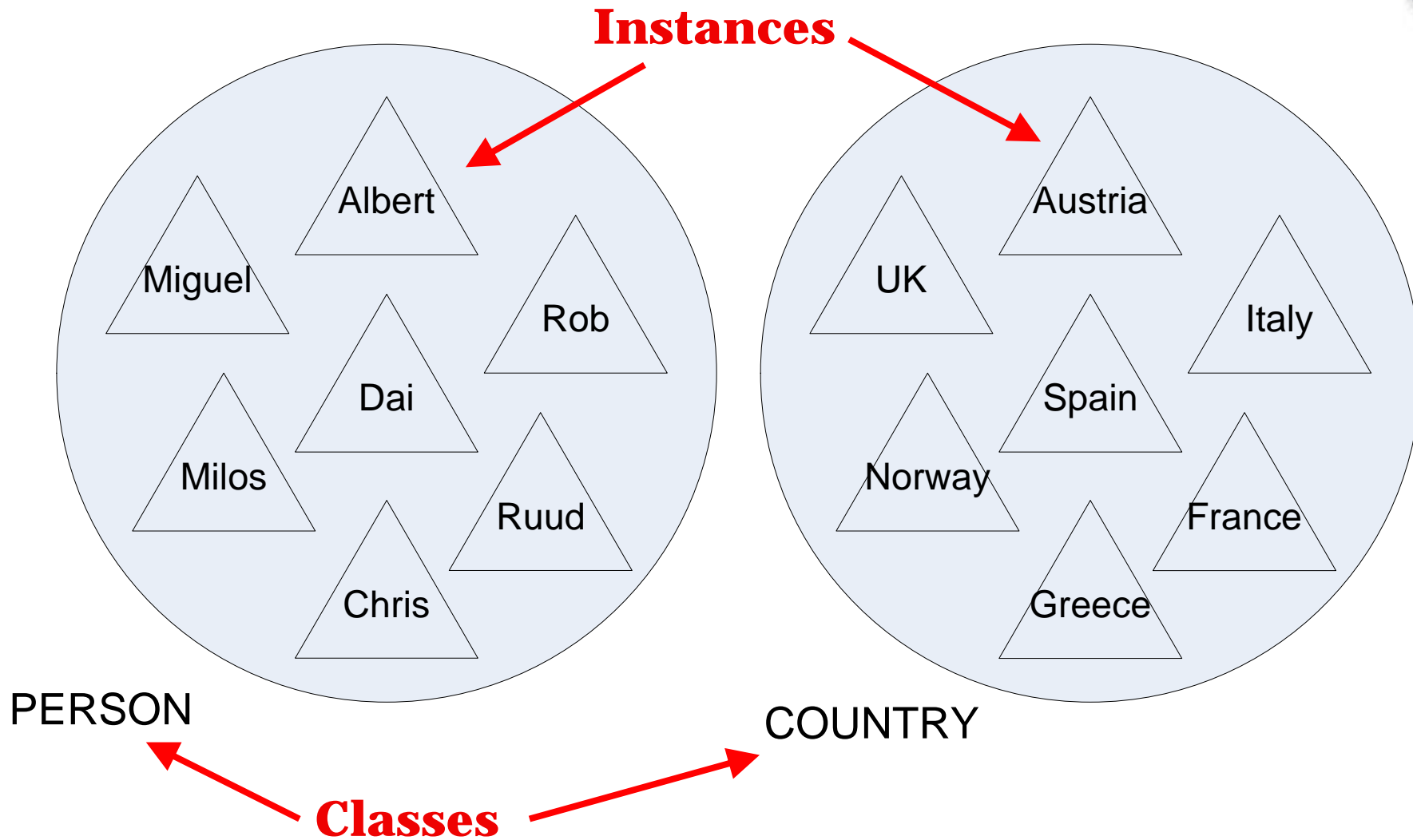
- Which can be shared among people
- Reuse of vocabularies and its definitions
 - Avoiding creating two resources with same meaning and different URIref
- Using e.g. RDF Vocabulary Description Language (aka RDF Schema or RDFS)

Resource Descriptions and Vocabularies (I)

Basic Vocabulary Concepts: Classes and Instances



Forschungszentrum L3S



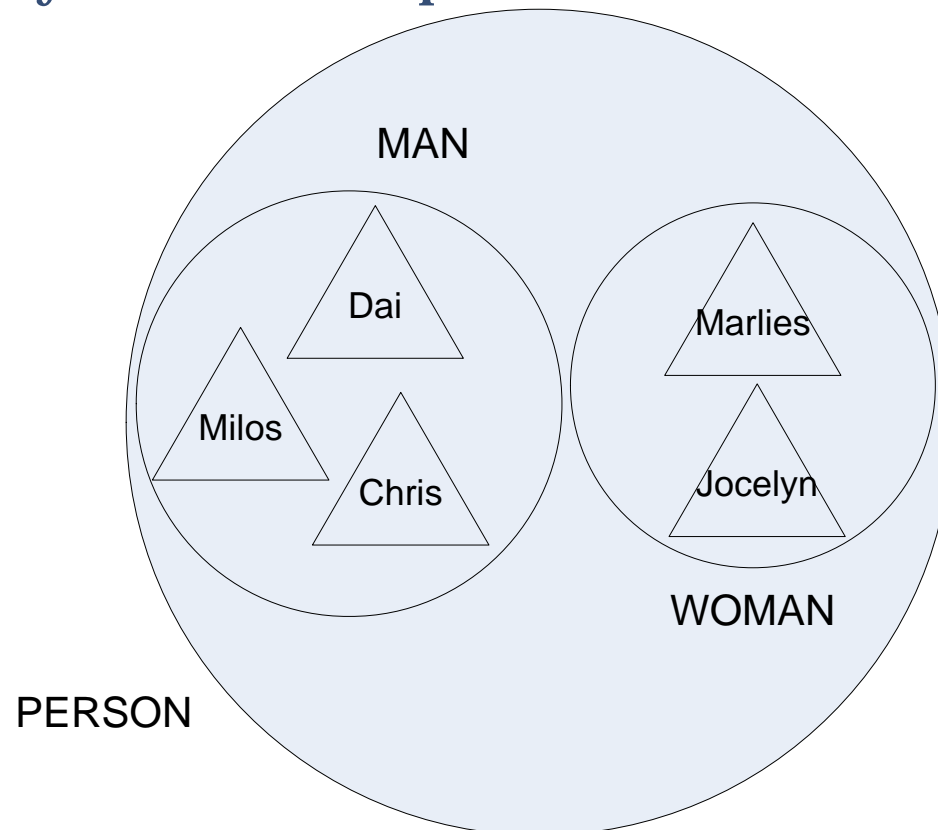
Resource Descriptions and Vocabularies (I)

Basic Vocabulary Concepts: Subclassing



Any instance in a subclass is also an instance of the superclass

- E.g. Any woman is a person



Resource Descriptions and Vocabularies (I)

Basic Vocabulary Concepts: Properties

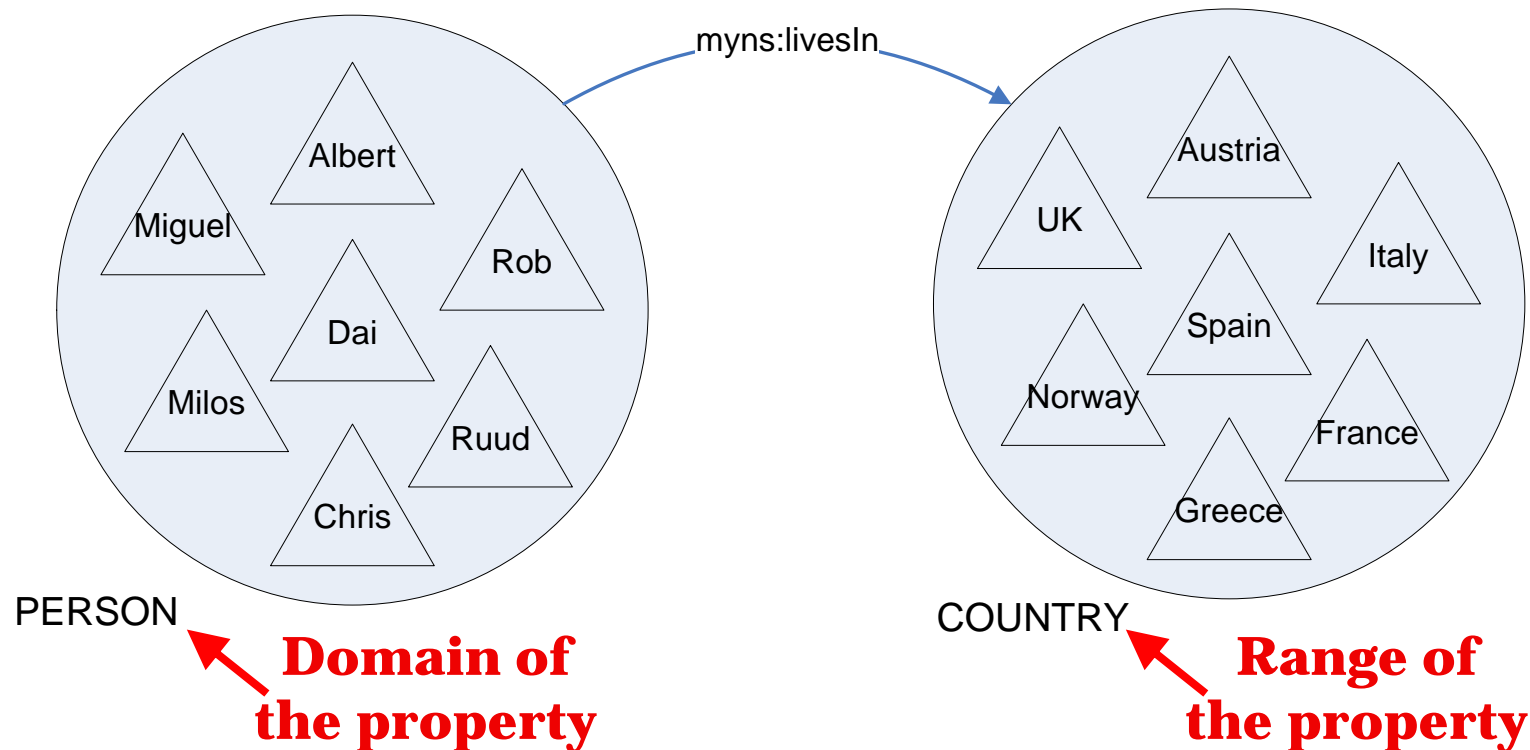


Link two individuals

- E.g., Miguel lives in Spain

or an individual and a literal

- E.g. Miguel has full name “Miguel Arjona”



Resource Descriptions and Vocabularies (I)

Protégé Vocabulary/Ontology Editor



A free, open source ontology editor and knowledge-base framework

- Suite of tools to construct domain models and knowledge-based applications with ontologies
- Based on Java, extensible, and provides a plug-and-play environment

Available from <http://protege.stanford.edu/>

Resource Descriptions and Vocabularies (I)

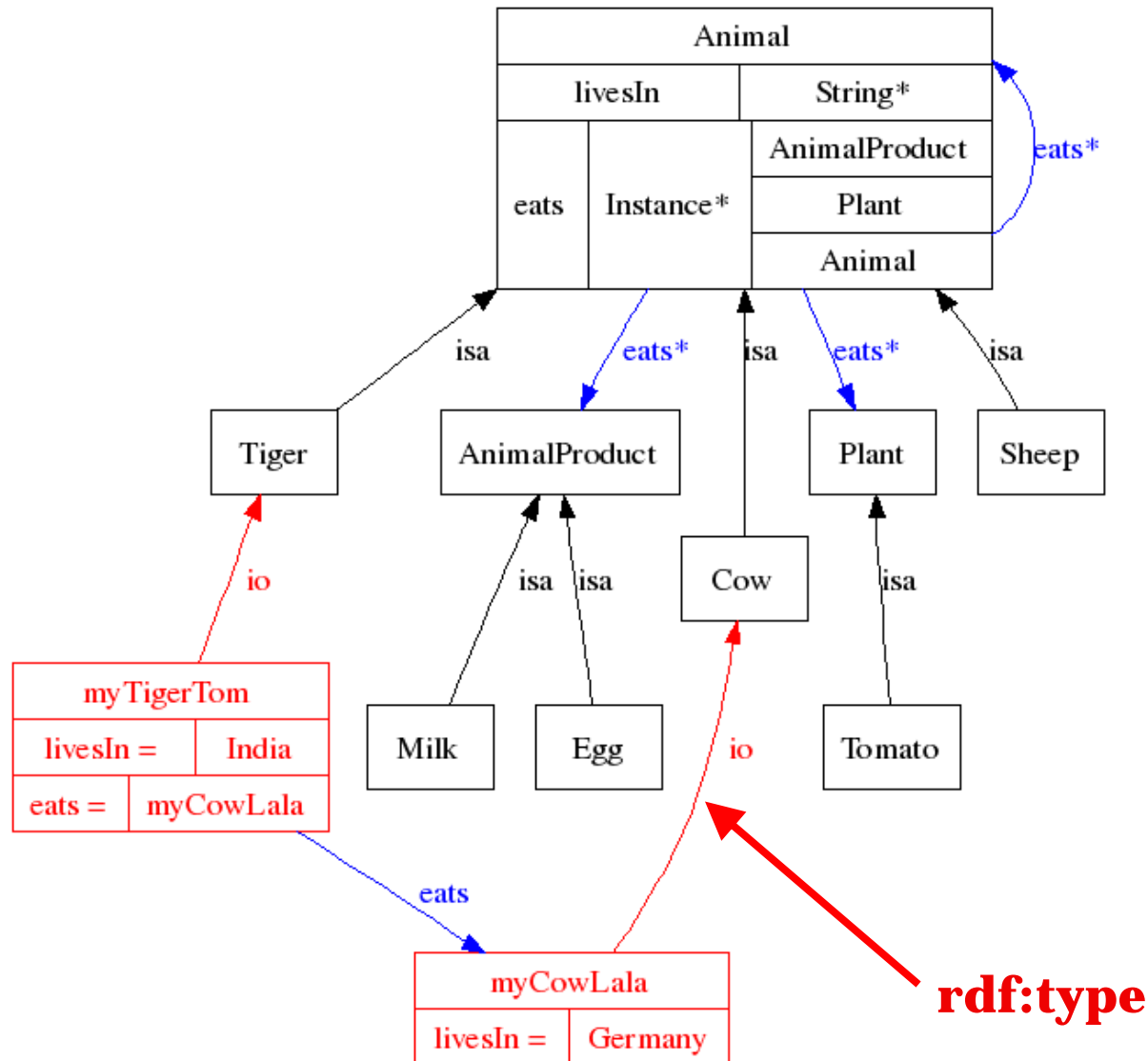
Exercise: Creating a Basic Vocabulary



- Open Protégé
- Create classes and subclasses
 - Animal (Cow, Tiger, Sheep)
 - Plant (Tomato)
 - AnimalProduct (Milk, Egg)
- Properties
 - eats
- Instances
 - myCow and myTiger
- Visualization
- Save the project

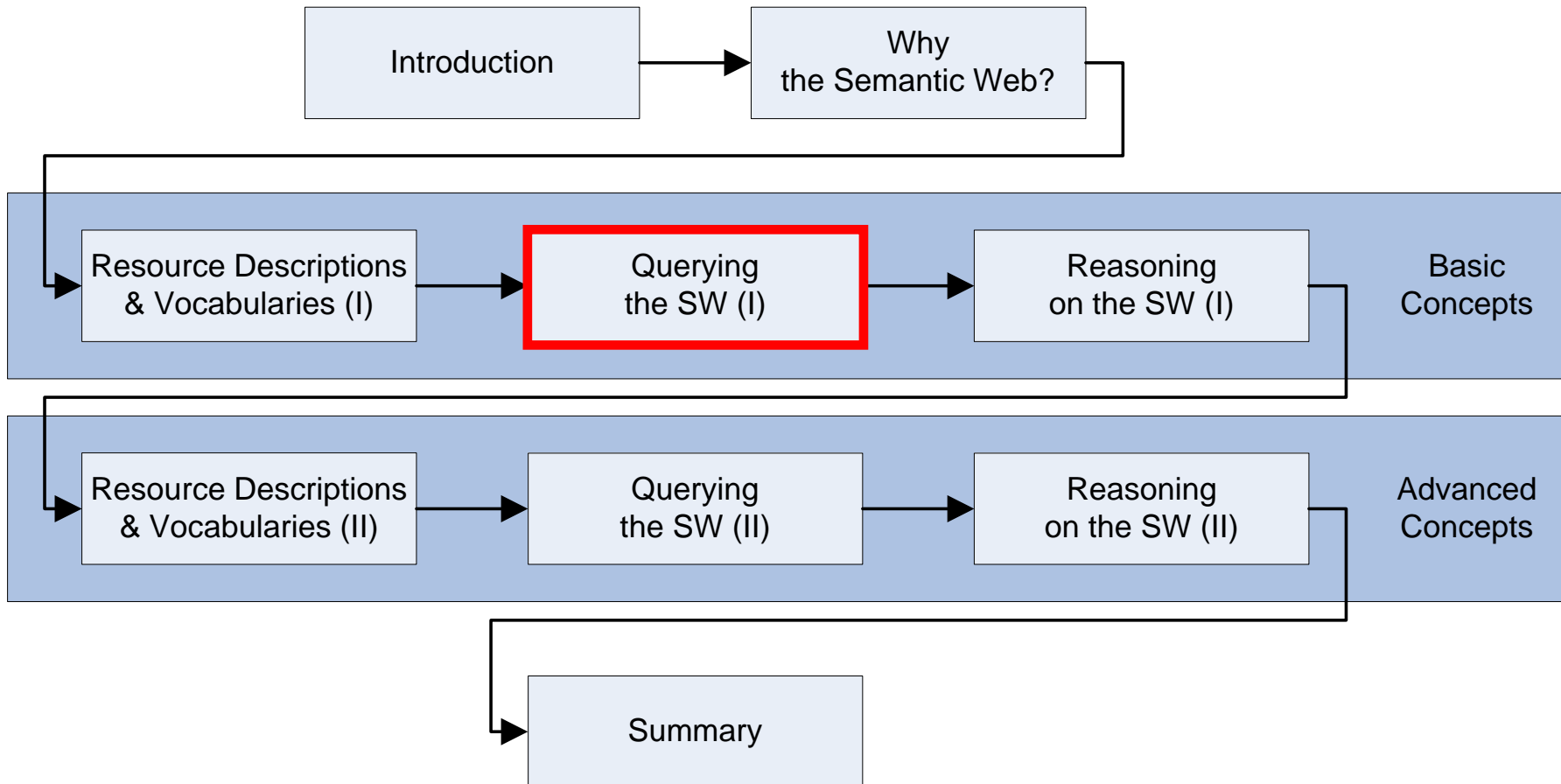
Resource Descriptions and Vocabularies (I)

Result of the exercise



Outline

Lecture overview



Querying the Semantic Web I

Prerequisites: Publication ontology and graph



Describes publications, authors, venues, etc.

- Introduced to annotate ISWC conference proceedings
- Data generated from bibtex files

Main classes

- Publication
- Person

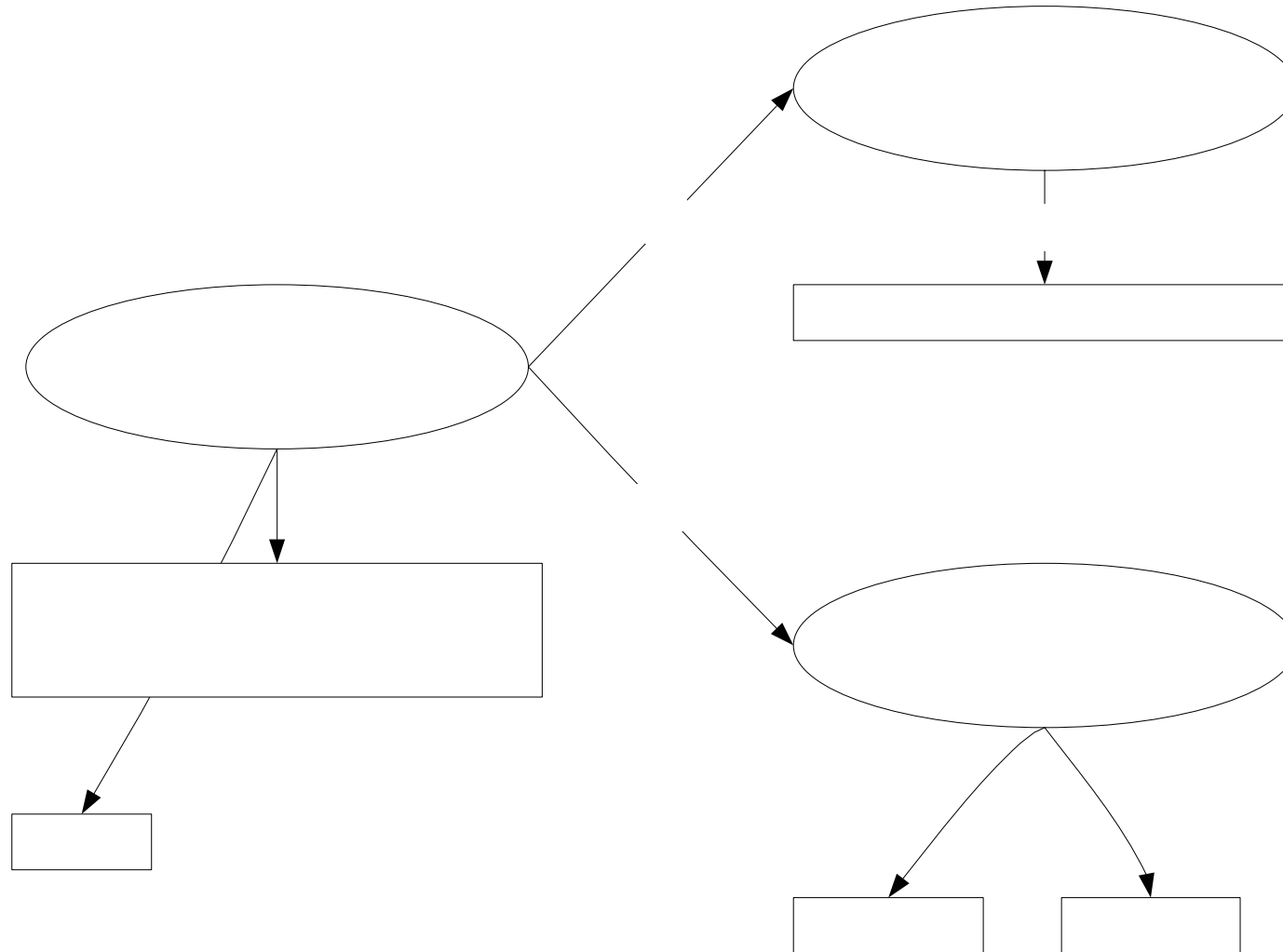
Main properties

- author
- in

- ▼ ● Publication
 - Article
 - Book
 - Booklet
- ▼ ● Collection
 - Journal
 - Periodical
 - Proceedings
- InBook
- InCollection
- InProceedings

Querying the Semantic Web I

Example Publication instance





Browsing

- Follow links between resources
- Web: Browsing facility
- SemWeb: Protegé, ...

Querying

- Finds resources according to given search criteria
- Web: keyword query / search engine
- SemWeb: SPARQL query / query engine

Querying the Semantic Web I

Why keywords are not enough



Find titles of all proceedings in which York Sure has published

- Impossible to translate to keyword query
 - relation between search terms
 - needs reference to classes/properties

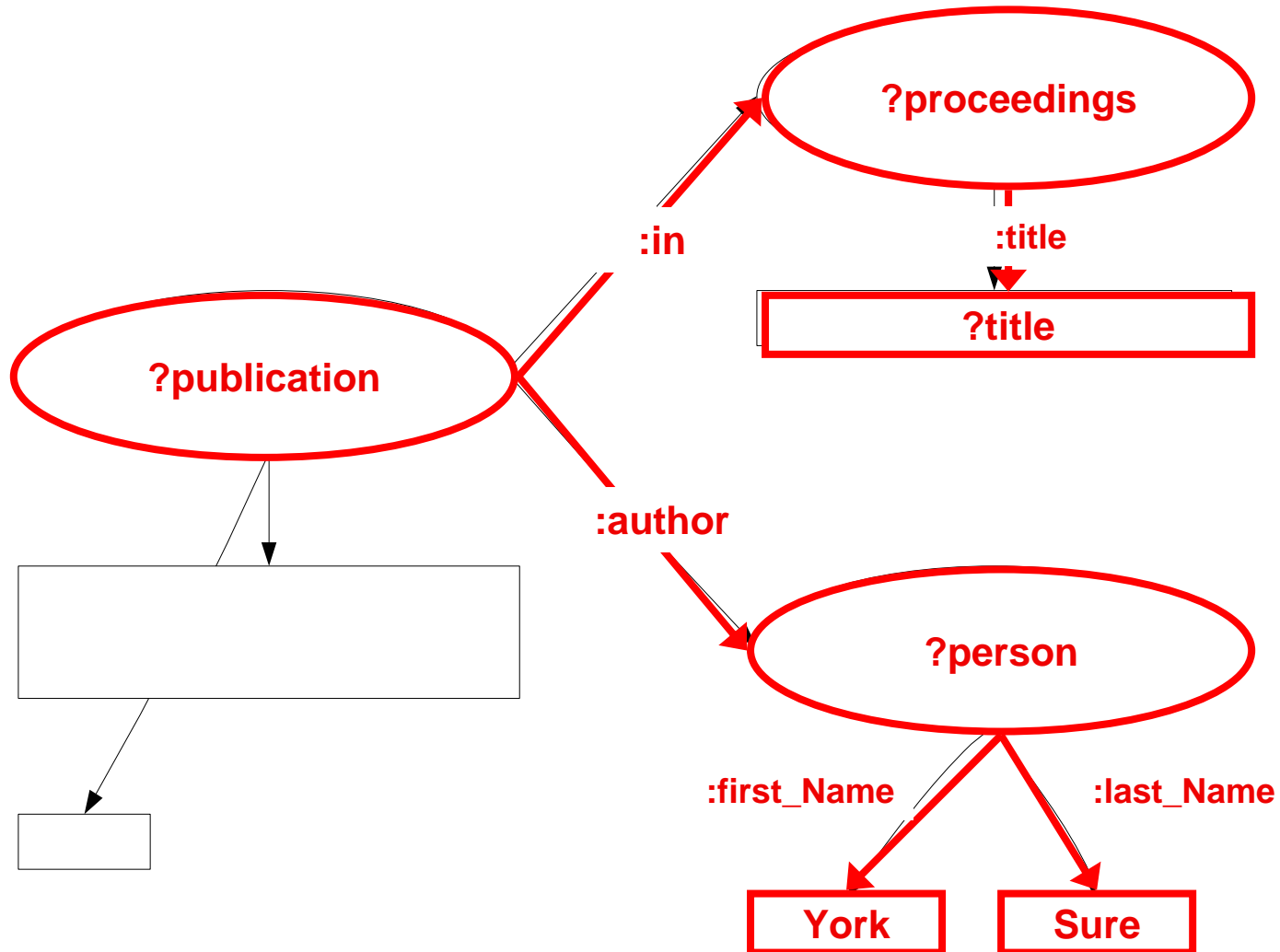
→ Querying as RDF graph pattern matching

Querying the Semantic Web I

Querying as Pattern Matching



Find titles of all venues in which York Sure has published





W3C Working Draft

Resembles SQL

- Basic query structure:

SELECT <what>

WHERE <search criteria>

<what>: query variables; start with '?'

<search criteria>

- Statement patterns
- additional conditions (e.g., '?year > 2000')
- ...

Querying the Semantic Web I

RDF Pattern matching



Find all titles

```
SELECT ?t WHERE
```

```
{ ?x :title ?t. }
```

Result specification
Statement pattern

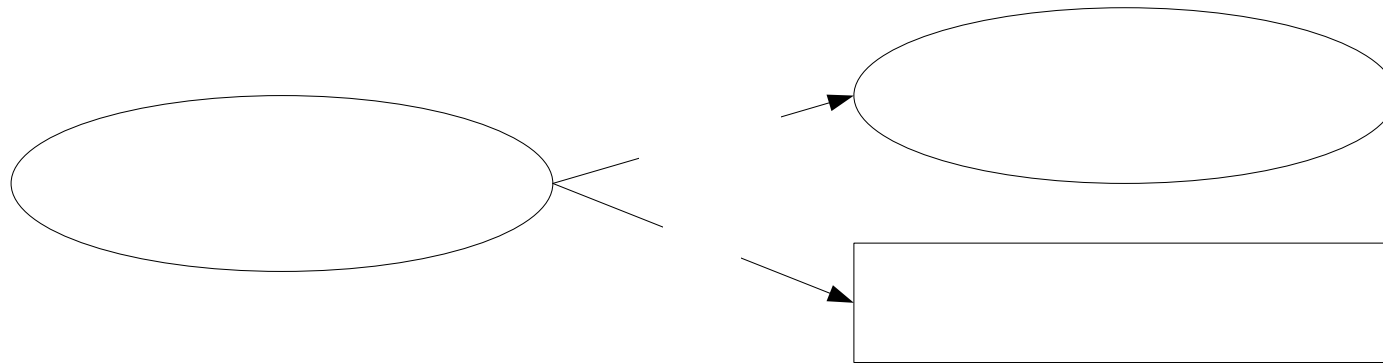
- Query variables start with '?'
- Variables are *bound* to actual statement values:

?t
Introducing Ontology-based Skills Management ...
Webster Online Dictionary
IEEE 1484.20.1/Draft - Draft Standard for Reusable Competency Definitions...
A Formal Approach to Ontology-Based Semantic Match of Skills Descriptions.
...



Find titles of all Proceedings

```
SELECT ?t WHERE
{ ?x :title ?t.
  ?x rdf:type :Proceedings. }
```



- Use `rdf:type` statement to refer to classes



Namespace declaration to abbreviate URIs

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
```

```
PREFIX      : <http://annotation.semanticweb.org/2004/iswc#>
```

```
SELECT ?t WHERE  
{  
  ?x :title ?t.  
  ?x rdf:type :Proceedings. }  
  
■ automatically added to query by Protegé
```



Find first and last names of Persons

```
SELECT ?first ?last WHERE
{
  ?x :first_Name ?first.
  ?x :last_Name ?last.
  ?x rdf:type :Person.}

```

Find first and last names of all article authors

```
SELECT ?last WHERE
{
  ?x :last_Name ?last.
  ?a :author ?x.
  ?a rdf:type :Article.}

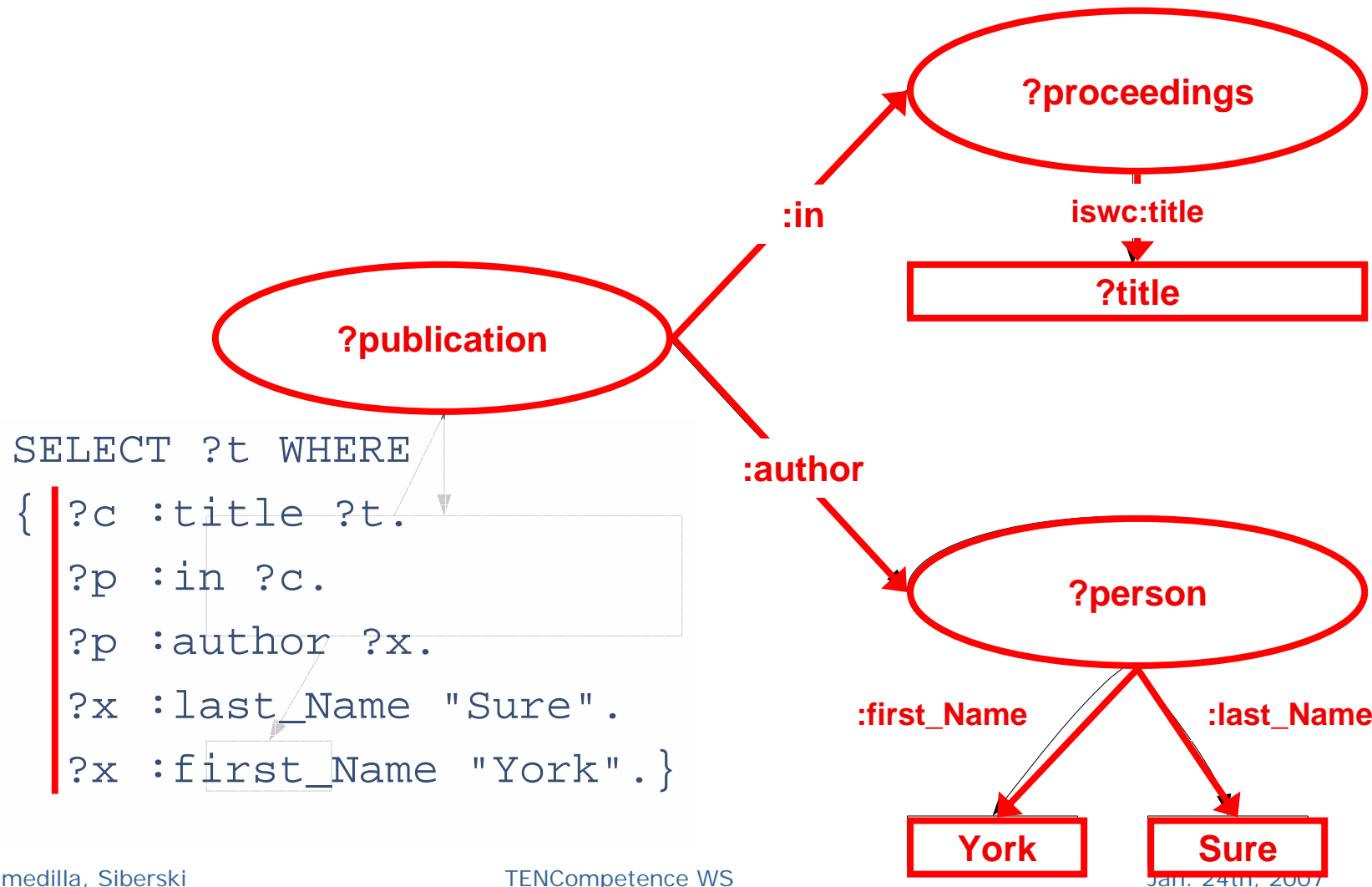
```

Querying the Semantic Web I

Pattern Matching Examples (II)



Find titles of all venues in which York Sure has published





Open Protegé SPARQL query panel

Write and execute the following queries:

- Find the titles of all articles
- Find the titles and years of all conference papers (InProceedings)
- Find the last names of all authors of books

Querying the Semantic Web I

Solutions



find the titles of all articles

```
SELECT ?t WHERE
{ ?x :title ?t.
  ?x rdf:type :Article.}
```

find the titles and years of all conference papers

```
SELECT ?t ?y WHERE
{ ?x :title ?t.
  ?x :year ?y.
  ?x rdf:type :InProceedings.}
```

find the last names of all authors of books

```
SELECT ?last WHERE
{ ?x :last_Name ?last.
  ?a :author ?x.
  ?a rdf:type :Book.}
```

Querying the Semantic Web I

FILTER: add constraints



find the titles of all articles not older than 2002

```
SELECT ?t WHERE
{ ?x :title ?t.
  ?x rdf:type :Article.
  ?x :year ?y.
  FILTER (?y >= "2002"). }
```

find titles of articles or books:

```
SELECT ?t WHERE
{ ?x :title ?t.
  ?x rdf:type ?c.
  FILTER (?c = :Article || ?c = :Book). }
```



Boolean expressions

- And (&&), or (||), not (!)

Comparison expressions

- =, !=, <, >, <=, >=

Numeric expressions

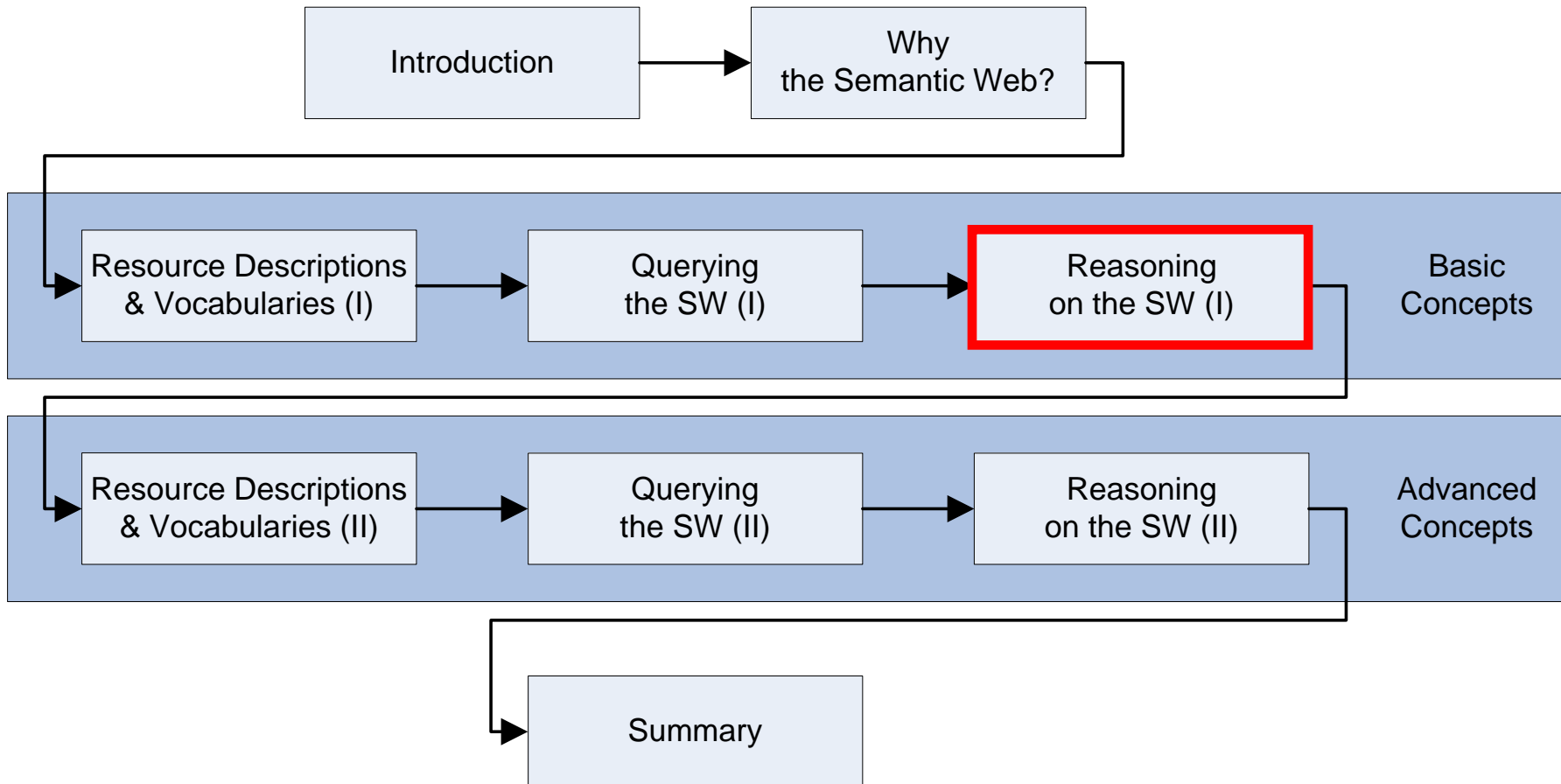
- +, -, *, /

Builtin functions

- Regular expressions
- ‚Node type expressions‘ (isLiteral, isURI)
- ...

Outline

Lecture overview





“An ontology is a specification of a conceptualization”

“A data model that represents a domain and is used to reason about the objects in that domain and the relations between them”

“Form of knowledge representation about the world or some part of it”

Reasoning on the Semantic Web I

Use of ontologies



Used to capture knowledge about a domain of interest

- Explicit Knowledge
- Implicit Knowledge

Complex concepts can be built from simpler ones. Using a reasoner

- updates hierarchy (useful when classes may have more than one parent)
- checks consistency
- find out where a class or instance belongs to

Reasoning on the Semantic Web I

Simple examples of inference (reasoning)



- When it rains the ground is wet
- It rains
- The ground is wet

- An animal that eats animals is carnivore
- A tiger eats animals
- A tiger is carnivore

Notation:

- Explicit knowledge
- Inferred knowledge

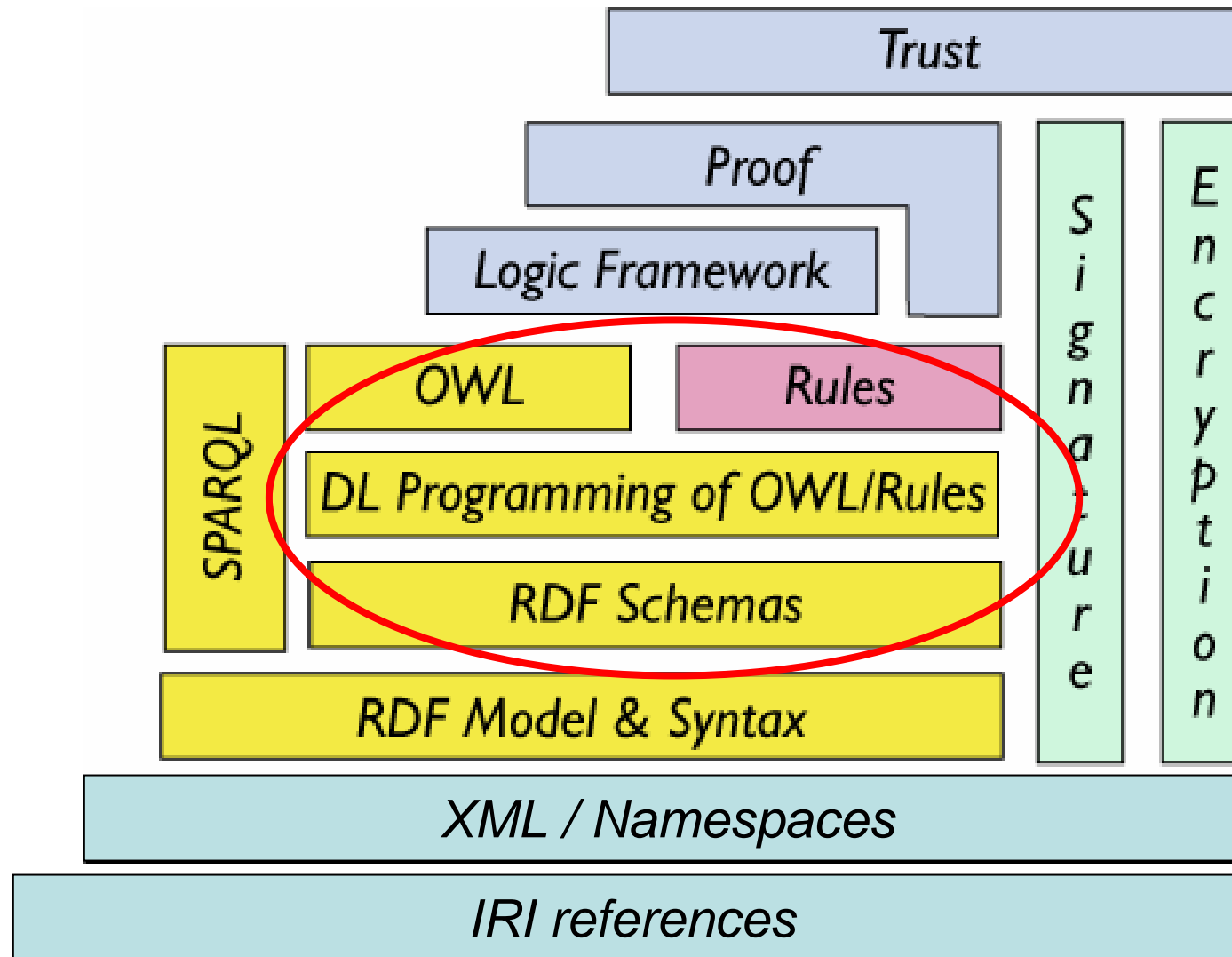


OWL: Web Ontology Language

~~Ontology = OWL~~

Reasoning on the Semantic Web I

The ontology level



Reasoning on the Semantic Web I

Description Logics (DL)



Family of knowledge representation languages

The name description logic refers to

- Concept descriptions used to describe a domain
- Logic-based semantics which can be given by a translation into first-order predicate logic

Logic in which OWL-DL is based

In this part of the lecture we will use only DL notation



Concepts (aka Classes) represent sets of individuals

- Tiger: class representing all existing tigers
- Other examples: Woman, Father, ...
- owl:Thing: set containing all individuals
- **Described based on requirements for membership of the class**

Roles (aka Properties, Slots): link two individuals

- eats: A tiger eats a cow
- hasChild: a person is the child of another

Reasoning on the Semantic Web I

Description Logics: Class Definitions



Primitive classes (aka concept inclusion, necessary)

- If an individual is a member of the class, then it must satisfy conditions
- E.g. A Tiger is an Animal ($\text{Tiger} \subseteq \text{Animal}$)
- E.g. A Researcher is a Person and she has a Publication
 - $\text{Researcher} \subseteq \text{Person} \sqcap \exists \text{hasPublication.T}$

Defined classes (aka concept equivalence, necessary and sufficient)

- If an individual is a member of the class, then it must satisfy conditions and
- Any individual satisfying the conditions must be a member of the class
- E.g. A Woman is a Female Person ($\text{Woman} \equiv \text{Person} \sqcap \text{Female}$)
- E.g. A Mother is a Woman and she has a child
 - $\text{Mother} \equiv \text{Woman} \sqcap \exists \text{hasChild.T}$



■ **Universal** ($\forall R.C$) (aka **only** restriction)

- Set of all individuals with all roles R with value C
- DogLover is someone whose pets are all dogs
- DogLover $\equiv \forall \text{hasPet.Dog}$
- Note!! If no relation exists, the restriction is still satisfied

■ **Existential** ($\exists R.C$) (aka **some** restriction)

- Set of all individuals with at least one role R with value C
- DogLiker is someone who owns a dog
- DogLiker $\equiv \exists \text{hasPet.Dog}$

R is the role and C is the filler of the role

Reasoning on the Semantic Web I

Exercise: Add Definitions to our Ontology



Add the following definitions:

- A tiger eats cows
 - $\text{Tiger} \subseteq \exists \text{ eats.Cow}$
- Carnivore = anything that eats other animals
 - $\text{Carnivore} \equiv \exists \text{ eats.Animal}$
- Vegetarian = anything that does not eat Animals
 - $\text{Vegetarian} \equiv \forall \text{ eats.}\neg\text{Animal}$



- **Subsumption (\sqsubseteq)**
 - One class is a subclass (is subsumed by) of another class
- **Equivalence (\equiv)**
 - Two classes are synonyms
 - $C \equiv D$ iff both $C \sqsubseteq D$ and $D \sqsubseteq C$
- **Membership**
 - An individual belongs to a class
- **Consistency**
 - Whether it is possible for a class to have any instances
 - There are no inconsistencies (mistakes) in the ontology

Reasoning on the Semantic Web I

Subsumption Example



- A cow and a tiger are animals
 - $Cow \subseteq Animal$
 - $Tiger \subseteq Animal$
 - A tiger eats cows
 - $Tiger \subseteq \exists \text{ eats.Cow}$
 - Carnivore = anything that eats other animals
 - $Carnivore \equiv \exists \text{ eats.Animal}$
- **Tiger is a subclass of carnivore**
- **$Tiger \subseteq Carnivore$**

Reasoning on the Semantic Web I

Membership Example



- A cow and a tiger are animals
 - $Cow \subseteq Animal$
 - $Tiger \subseteq Animal$
 - A tiger eats cows
 - $Tiger \subseteq \exists \text{ eats.Cow}$
 - myTiger is a tiger
 - $myTiger \in Tiger$
 - Carnivore = anything that eats other animals
 - $Carnivore \equiv \exists \text{ eats.Animal}$
- myTiger is an individual of carnivore**
- $myTiger \in Carnivore$



Add the following definitions:

- $Cow \subseteq Animal \sqcap \neg Vegetarian$
- A MadCow is a cow and eats sheeps
 - $MadCow \subseteq Cow \sqcap \exists \text{ eats.Sheep}$

Reasoning on the Semantic Web I

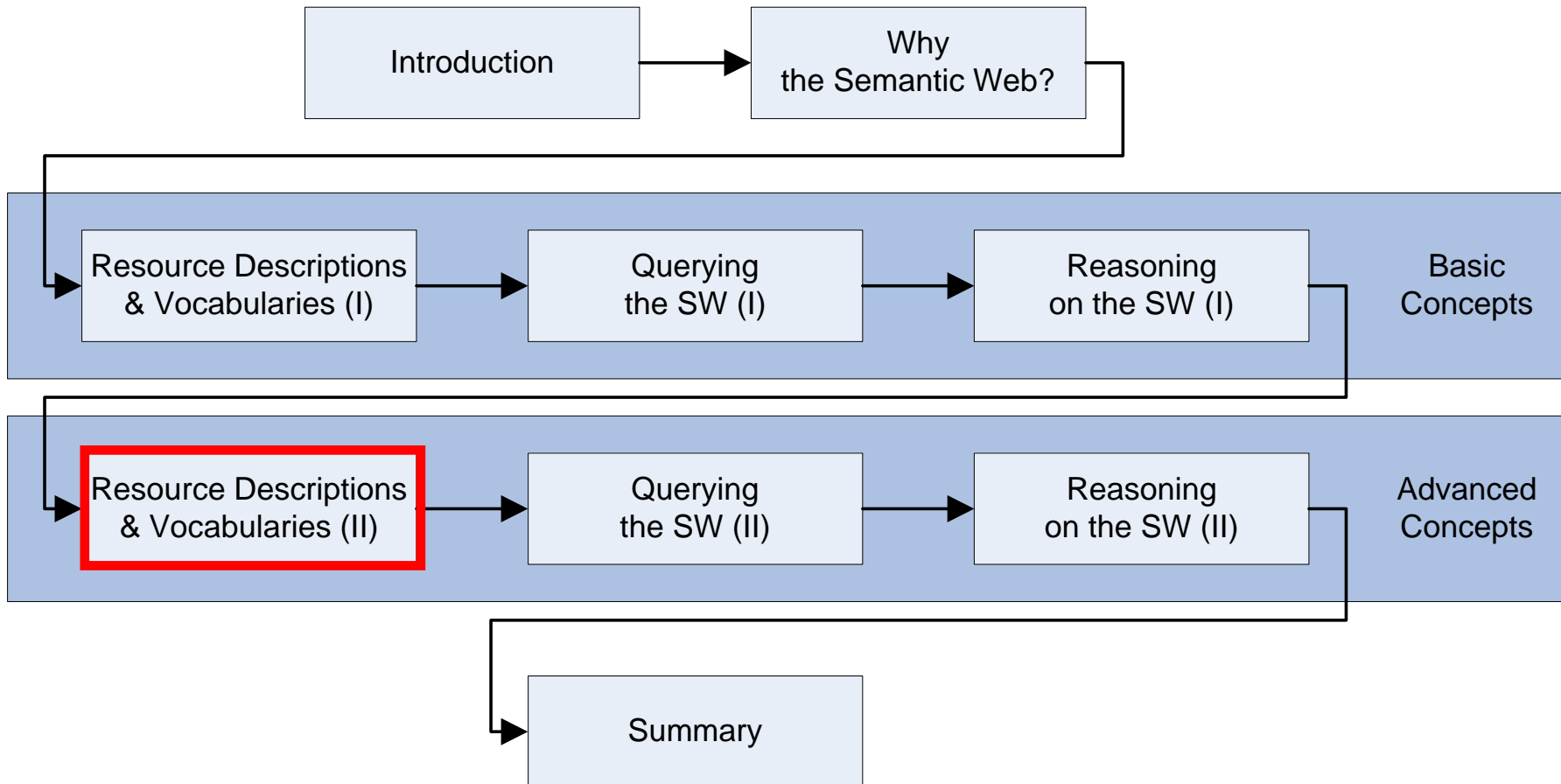
Inconsistency Example



- A cow and a Sheep are animals
 - $Cow \subseteq Animal \sqcap Vegetarian$
 - $Sheep \subseteq Animal$
- A MadCow is a cow and eats sheeps
 - $MadCow \subseteq Cow \sqcap \exists \text{ eats.Sheep}$
- Vegetarian = anything that does not eat Animals
 - $Vegetarian \equiv \forall \text{ eats.}\neg Animal$
- **MadCow is inconsistent !!!**
 - $MadCow \subseteq Cow \sqcap \exists \text{ eats.}\neg Animal \sqcap \exists \text{ eats.Sheep}$
 - $Sheep \subseteq Animal$

Outline

Lecture overview



Resource Descriptions and Vocabularies II

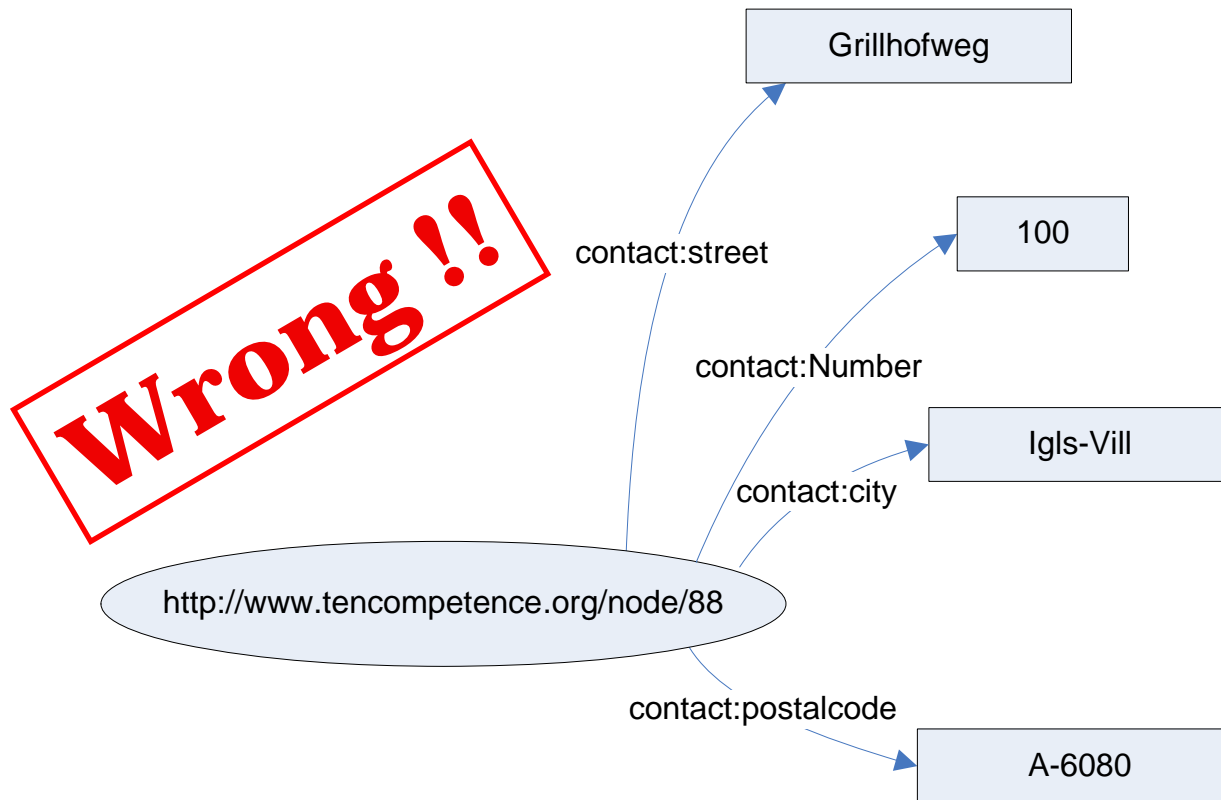
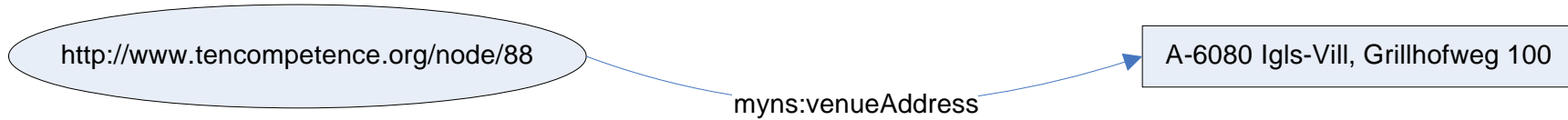
Exercise: Modeling more complex examples



- The above graph provides the address of the winter school as a whole.
- Redraw the graph so the address is split into street, number, postal code and city

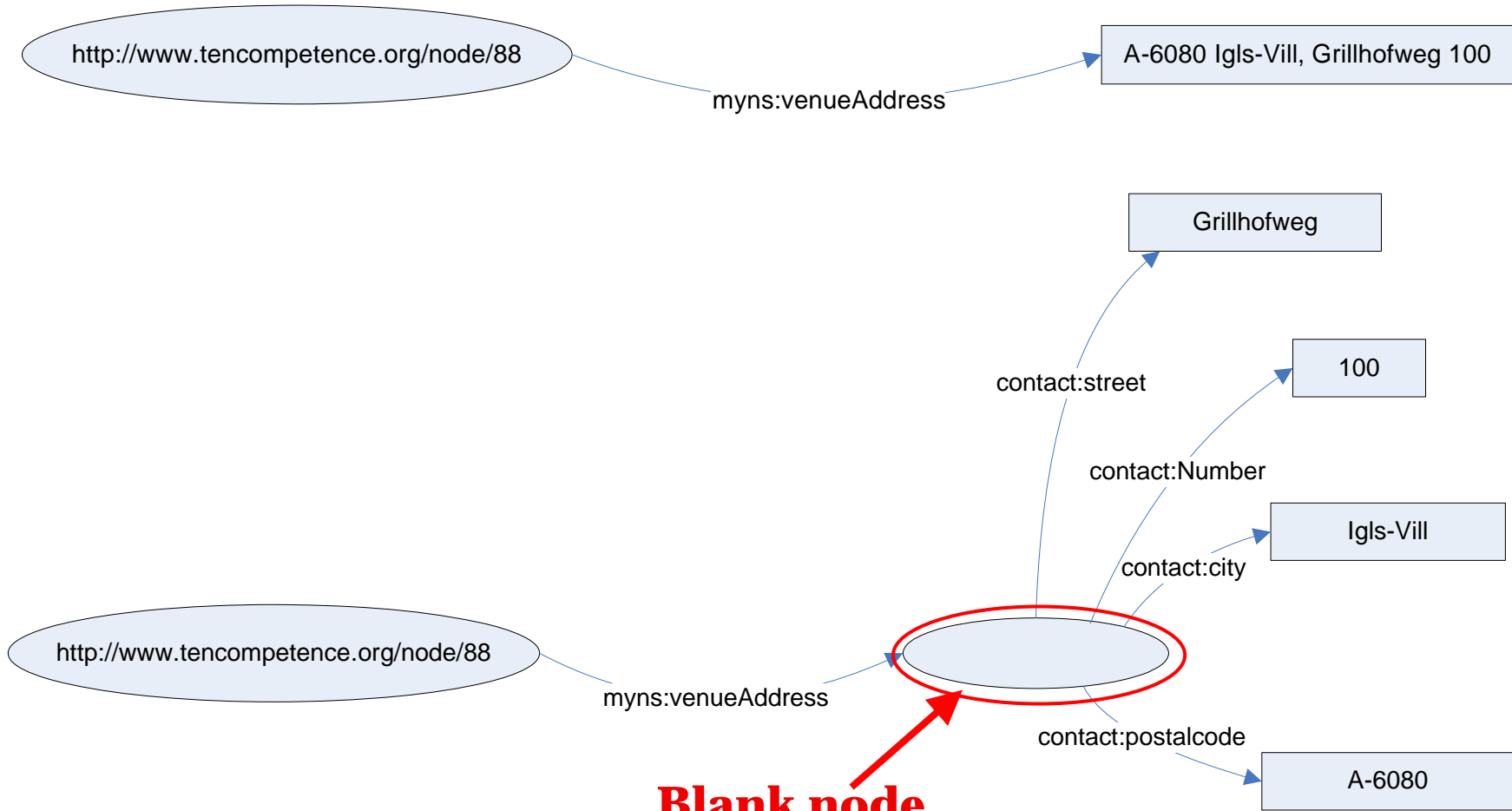
Resource Descriptions and Vocabularies II

Exercise: Solution



Resource Descriptions and Vocabularies II

Exercise: Actual Solution



**Blank node
aka anonymous resource**

Resource Descriptions and Vocabularies II

Blank Nodes



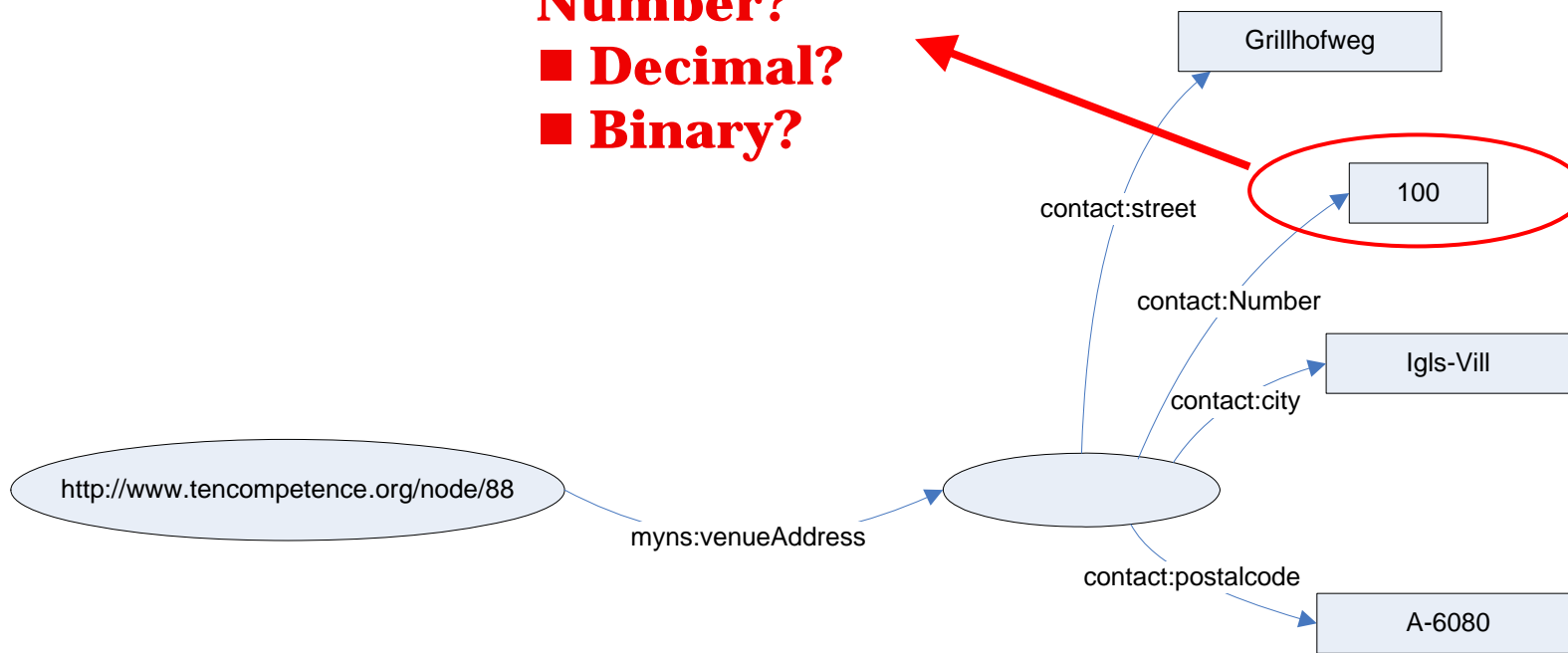
- RDF represents only binary relations
- This example is an n-ary relation
 - Must be broken into a group of binary relations
 - Blank node is needed
- Blank nodes are given so called blank node (anonymous) identifiers
 - This identifiers are not considered part of the graph
- Since they are nodes (not arcs), they can only appear as subject or object of triple statements
- Other example: weights with different units, price with different currencies

Resource Descriptions and Vocabularies II

RDF Typed Literals (I)



String?
Number?
■ Decimal?
■ Binary?



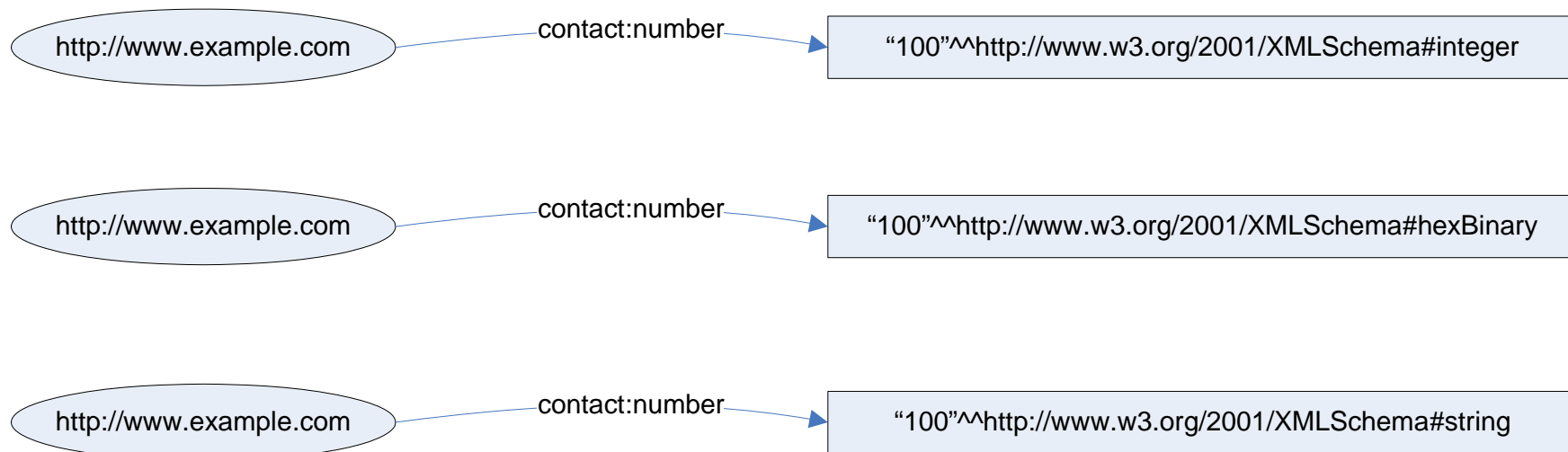
Resource Descriptions and Vocabularies II

RDF Typed Literals (& II)



The RDF graph does not give information about the datatype of a literal

- It can be explicitly added
- Use of XML datatypes



Resource Descriptions and Vocabularies II

RDF Containers



Need to describe groups of things

Container: resource that contains things (either resources or literals)

- Bag (rdf:Bag)
 - Allows duplicates
 - No order among members
- Sequence (rdf:Seq)
 - Allows duplicates
 - Order among members
- Alternative (rdf:Alt)
 - Represent alternatives (e.g., languages)
 - Specifies that any one of the members can be chosen
 - `rdf:_1` is considered the default or preferred value

This are intended meanings, no actual semantics of RDF

Containers are not necessarily closed (more members may exist)



Containers specify members of a group

- **They are not close**

Collections describe a group and specify all its members

- **No more members exist**

Rdf:List, rdf:first, rdf:Rest (like in Prolog)

Resource Descriptions and Vocabularies II

Reification



Metadata about statements

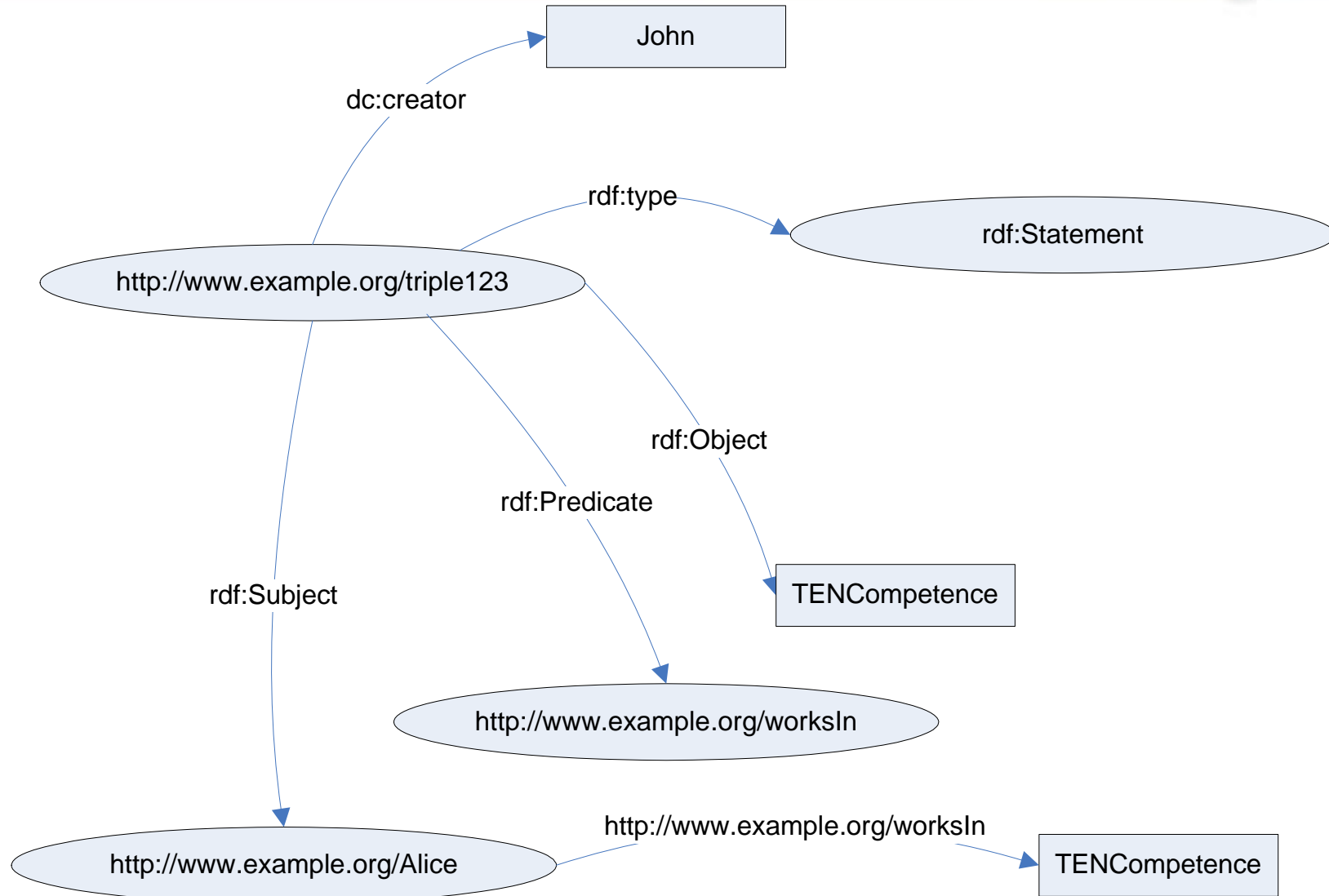
- Provenance
 - John dc:creator “Alice worksIn TENCompetence”
- Beliefs
 - John believes “Alice trusts Bob”
- ...

rdf:Statement, rdf:Subject, rdf:Predicate, rdf:Object

- aka “reification quad”

Resource Descriptions and Vocabularies II

Reification Example



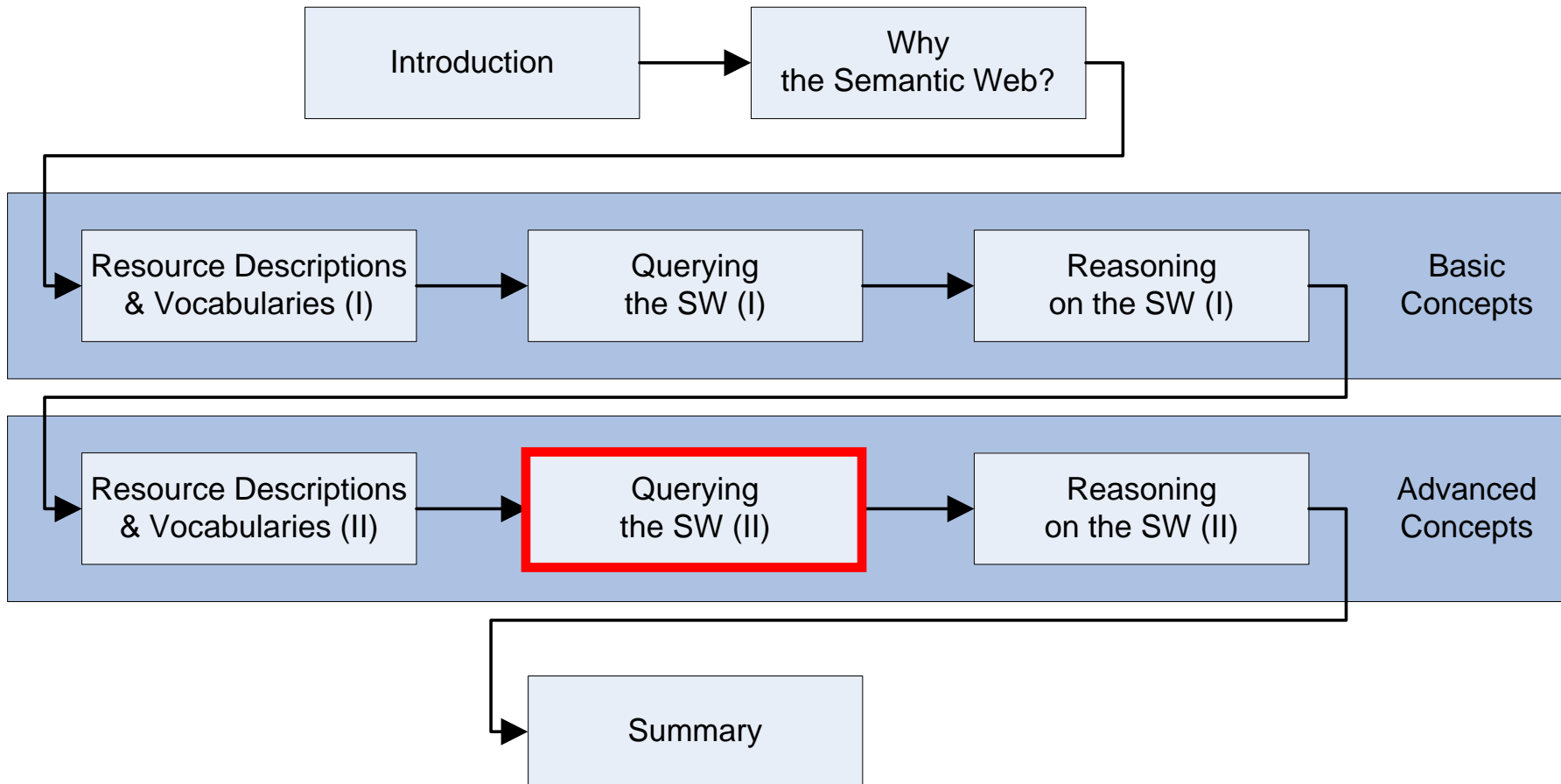


Different syntaxes have been developed

- **RDF/XML (“official” W3C format)**
- **Triple**
- **n3**
- **Turtle**
- **RXR**
- ...

Outline

Lecture overview





find title and year of all resources

```
SELECT ?t ?y WHERE
{ ?x :title ?t.
  ?x :year ?y. }
```

- This only returns resources with title and year

find titles of all resources, and year if it exists

```
SELECT ?t ?y WHERE
{ ?x :title ?t.
  OPTIONAL{?x :year ?y.} }
```

- Includes year if it exists, otherwise ?y is not bound



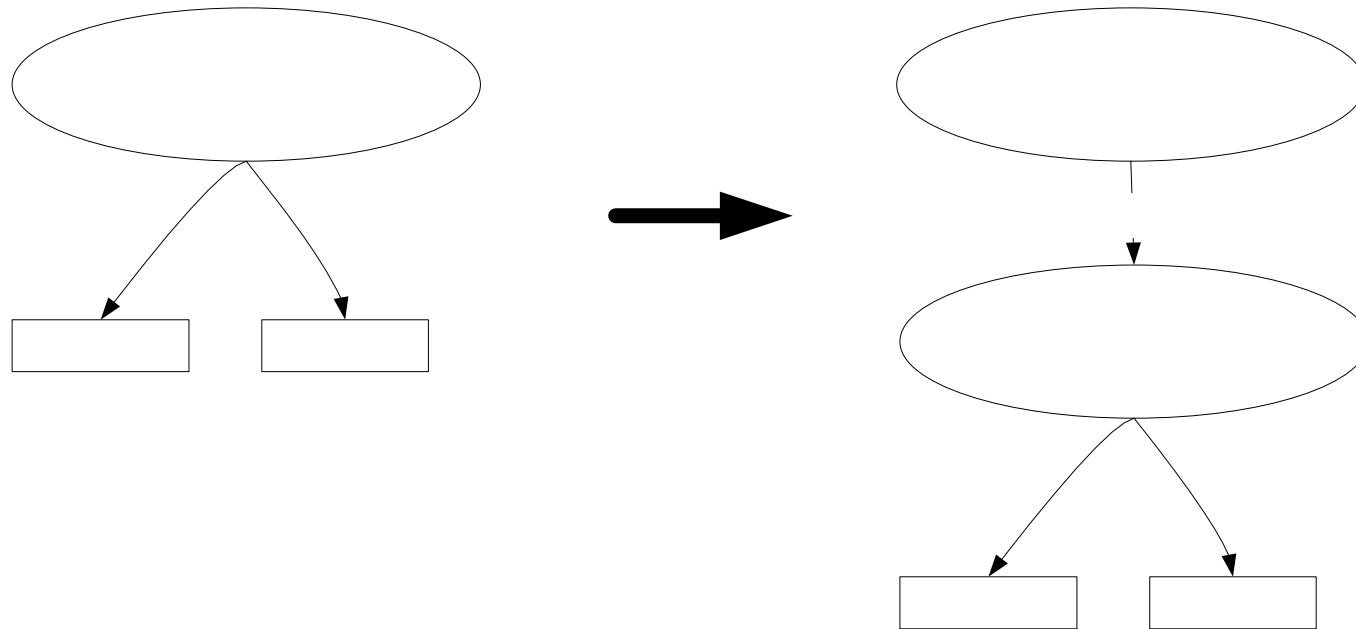
List resource names, where either title or last name is used as name

```
SELECT ?name WHERE
{ {?x :title ?name.}
  UNION
  {?x :last_Name ?name.}}}
```

- Use if the same information may be stored in different properties



Example: *convert ISWC to vCard format*





Two different output formats

- Tables (`SELECT...`)
- Graph (`CONSTRUCT...`)

Example: *convert ISWC to vCard format*

```
CONSTRUCT { ?x vcard:N _:v.  
            _:v vcard:givenName ?first.  
            _:v vcard:familyName ?last.  
        }
```

WHERE

```
{ ?x :first_Name ?first.  
  ?x :last_Name ?last. }
```



Sort names alphabetically

```
SELECT ?first ?last WHERE
{
  ?x :first_Name ?first.
  ?x :last_Name ?last.}
ORDER BY ?last ?first
```

- Orders by last name, then (if last name is equal) by first name

Remove duplicates

```
SELECT DISTINCT ?first ?last WHERE
{
  ?x :first_Name ?first.
  ?x :last_Name ?last.}
```

Querying the Semantic Web II

Solution Modifiers (& II)



Return only first 10 results

```
SELECT ?first ?last WHERE
{ ?x :first_Name ?first.
  ?x :last_Name ?last.}
ORDER BY ?last ?first
LIMIT 10
```

Return next 10 results

```
SELECT ?first ?last WHERE
{ ?x :first_Name ?first.
  ?x :last_Name ?last.}
ORDER BY ?last ?first
OFFSET 10
LIMIT 10
```



Find the title of all publications

```
SELECT ?t WHERE
{
  ?x :title ?t.
  ?x rdf:type :Publication.
}
```

- Doesn't work with plain SPARQL

,Solution': Entailment Regimes

- Extend RDF model by all inferred statements
- Foreseen regimes
 - RDF entailment (no entailment)
 - RDFS entailment
 - OWL entailment



Mix DL and SPARQL for complex queries

Find titles of all publications except collections

- In OWL:

$\text{NoCollection} \equiv \text{Publication} \sqcap \neg\text{Collection}$

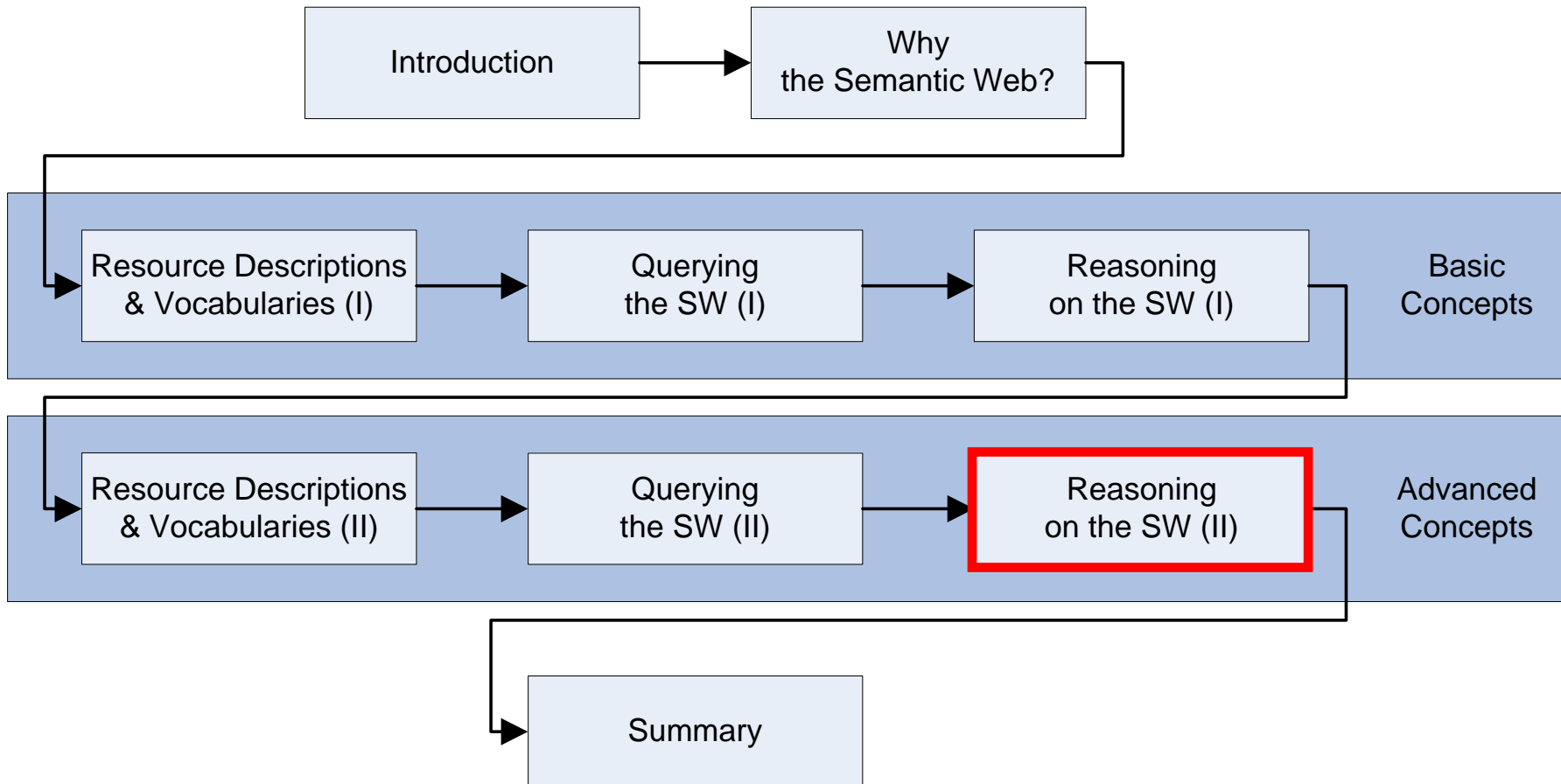
- In SPARQL:

```
SELECT ?title ?type WHERE
{
  ?x :title ?t.
  ?x rdf:type ?type.
  ?x rdf:type :NoCollection.
}
```

- Assumes OWL entailment regime

Outline

Lecture overview



Reasoning on the Semantic Web II

Property Types



- **Object:** links an individual to an individual
 - Animal eats Animal
- **Datatype:** links and individual to an XML schema datatype value or an RDF literal
 - Animal livesIn string
- **Annotation:** adds metadata
 - Class Animal is created by John Smith
 - Animal dc:creator "John Smith"

Subproperty (if two individuals are linked by a subproperty, then they are also by the superproperty)

- hasSibling isSubProperty of hasRelative

Reasoning on the Semantic Web II

Property Characteristics



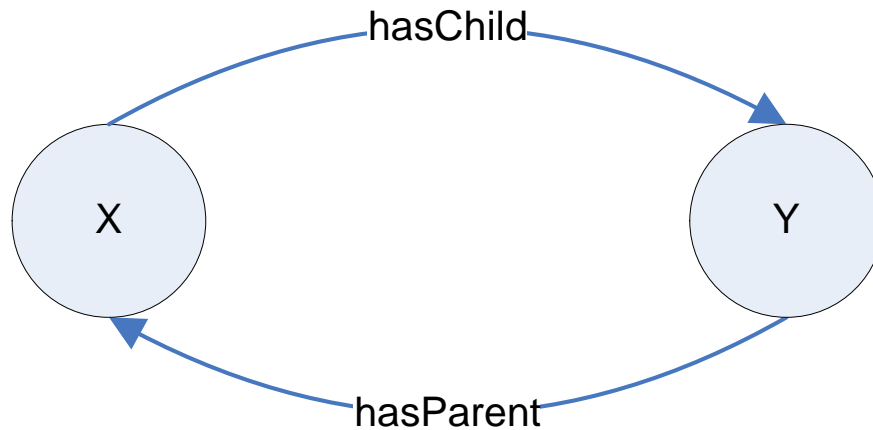
- **Inverse**
- **Transitive**
- **Symmetric**
- **Functional** (single valued property)
- **Inverse functional**

Reasoning on the Semantic Web II

Inverse Property



- $X \text{ p1 } Y \text{ iff } Y \text{ p2 } X$



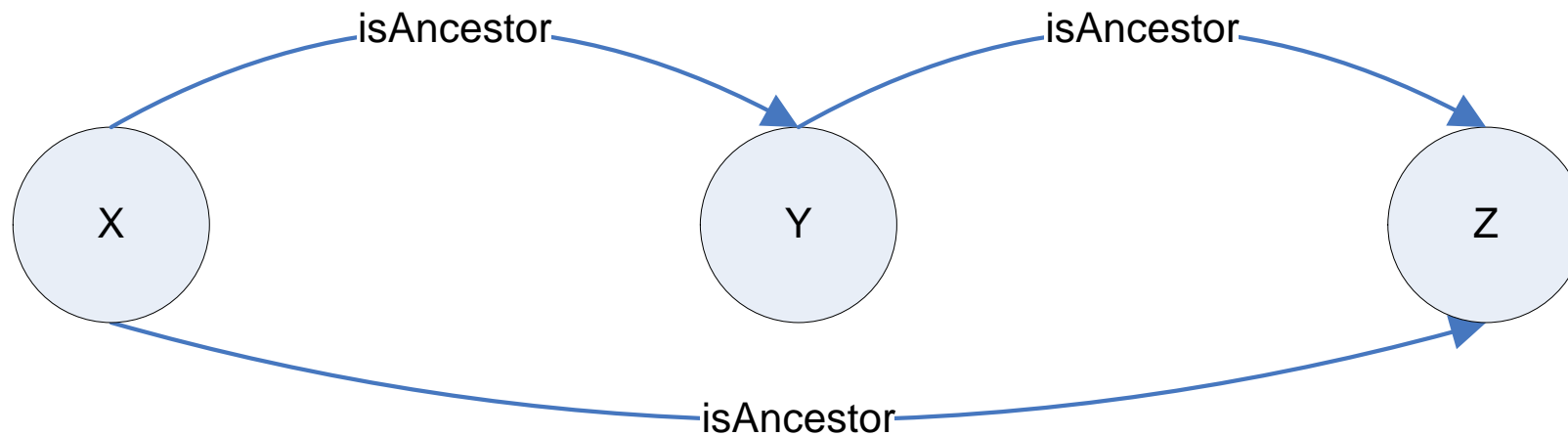
- $X \text{ eats } Y \text{ iff } Y \text{ isEatenBy } X$
- $X \text{ hasParent } Y \text{ iff } Y \text{ hasChild } X$

Reasoning on the Semantic Web II

Transitive Property



- If X-Y and Y-Z then X-Z



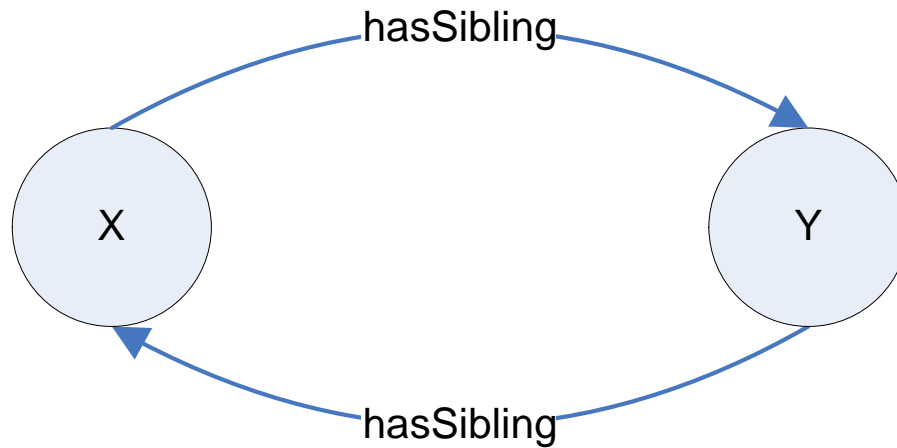
- X ancestor of Y, Y ancestor of Z, then X ancestor of Z

Reasoning on the Semantic Web II

Symmetric Property



- $X-Y$ iff $Y-X$



- X hasSibling Y iff Y hasSibling X

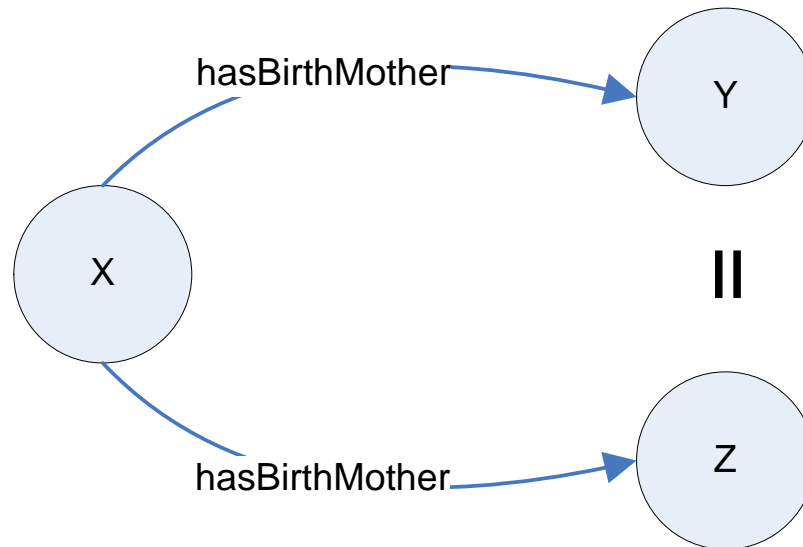
Reasoning on the Semantic Web II

Functional Property



Single valued property

- Only one individual can be related to it
- If X - Y and X - Z then $Y=Z$



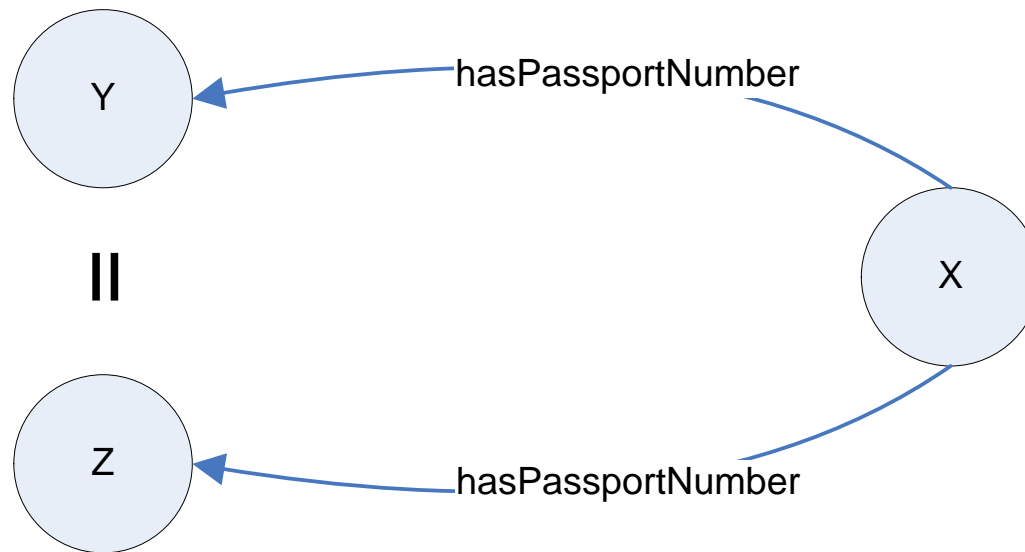
- If X hasBirthMother Y and X hasBirthMother Z then $Y=Z$

Reasoning on the Semantic Web II

Inverse Functional Property



- $Y-X$ and $Z-X$ then $Y=Z$



- Y hasPassportNumber X and Z hasPassportNumber X then $Y=Z$

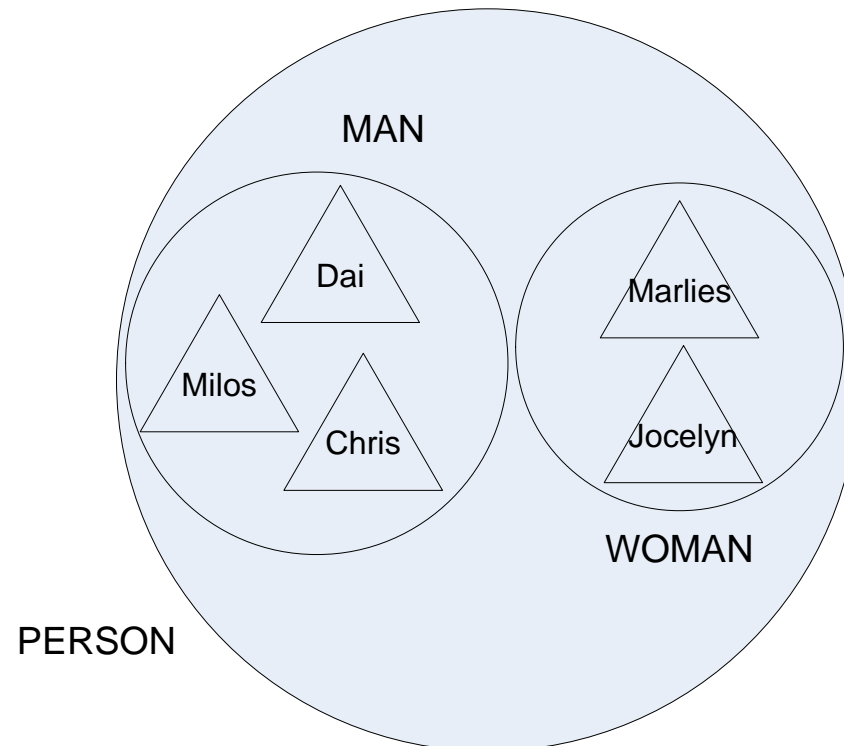
Reasoning on the Semantic Web II

Disjointness (I)



If two classes are disjoint, then an instance cannot be in both classes at the same time.

- E.g. Man and Woman are disjoint
- E.g. Researcher and FootballPlayer are not disjoint





Example

- A cow and a Sheep are animals
 - $Cow \subseteq Animal \sqcap Vegetarian$
 - $Sheep \subseteq Animal$
 - Vegetarian = anything that eats plants
 - $Vegetarian \equiv \forall \text{ eats.Plant}$
 - A MadCow is a cow and eats sheeps
 - $MadCow \subseteq Cow \sqcap \exists \text{ eats.Sheep}$
- MadCow is consistent !!!

Reasoning on the Semantic Web II

Disjointness (& III)



By default, it is assumed that individuals may belong to more than one class.

If we want to specify the contrary, then we need to explicitly say it

Example again:

- A cow and a Sheep are animals
 - $Cow \subseteq Animal \sqcap Vegetarian$
 - $Sheep \subseteq Animal$
 - Vegetarian = anything that eats plants
 - $Vegetarian \equiv \forall \text{ eats.Plant}$
 - A MadCow is a cow and eats sheeps
 - $MadCow \subseteq Cow \sqcap \exists \text{ eats.Sheep}$
 - Animal and Plant are disjoint
 - $Animal \subseteq \neg Plant$
- MadCow is inconsistent !!!



- **Cardinality: a class has at least, at most or exactly a number of relationships**
 - **Minimum cardinality**
 - Football game has at least 1 referee (≥ 1 hasReferee)
 - **Maximum cardinality**
 - Football team has at most 11 players (≤ 11 hasPlayer)
 - **Cardinality restriction**
 - A person has exactly one mother (≥ 1 hasMother \sqcap ≤ 1 hasMother)

- **hasValue**
 - The filler is an individual
 - $\text{Spanish} \sqsubseteq \exists \text{ citizenOf.}\{\text{"Spain"}\}$

Reasoning on the Semantic Web II

Common mistakes



Mistake 1

- A universal restriction is satisfied if no relation exists
- E.g. $\text{DogLover} \equiv \forall \text{hasPet.Dog}$
 - Note!! If no relation exists, the restriction is still satisfied

Mistake 2

- Property is functional (cardinality exactly 1)
- $X\text{-}Y \text{ and } X\text{-}Z \rightarrow Y = Z$
- If $X \text{ hasBirthMother } Y$ and $X \text{ hasBirthMother } Z$ then $Y=Z$

Mistake 3

- Domain and range are not requirements of membership.
 - If property has domain or range C
 - an instance (class) I owns the property or is the range of it
 - I is member (subclass) of C
 - $\text{eats has domain Animal, } X \text{ eats } Y \rightarrow X \subseteq \text{Animal}$

Reasoning on the Semantic Web I

OWL reasoning Languages



- **OWL Lite**
 - Classification Hierarchy and simple constraints

- **OWL DL**
 - Expressiveness of Descriptions Logics
 - retaining computational completeness and decidability

- **OWL Full**
 - Maximum expressiveness and syntactic freedom of RDF
 - No computational guarantees

- **OWL Lite < OWL DL < OWL Full**



Fact: Alice can access file “presentation.pdf”

■ **Closed World Assumption (CWA)**

- If something has not been stated true then it is assumed that is false
- Alice can access image “party.jpg”? → False

■ **Open World Assumption (OWA)**

- something cannot be assumed false because it has not been stated true
- Alice can access image “party.jpg”? → We do not know
- There may be somewhere another fact Alice can access image “party.jpg”

Reasoning on the Semantic Web I

Classification of statements



The terms ABox and TBox are used to describe two different types of statements in ontologies

■ TBOX

- Describe a system in terms of controlled vocabularies, for example, a set of classes and properties. A "terminological component"
- Associated with object-oriented classes
- $\text{Carnivore} \equiv \text{Animal} \sqcap \exists \text{ eats. Animal}$

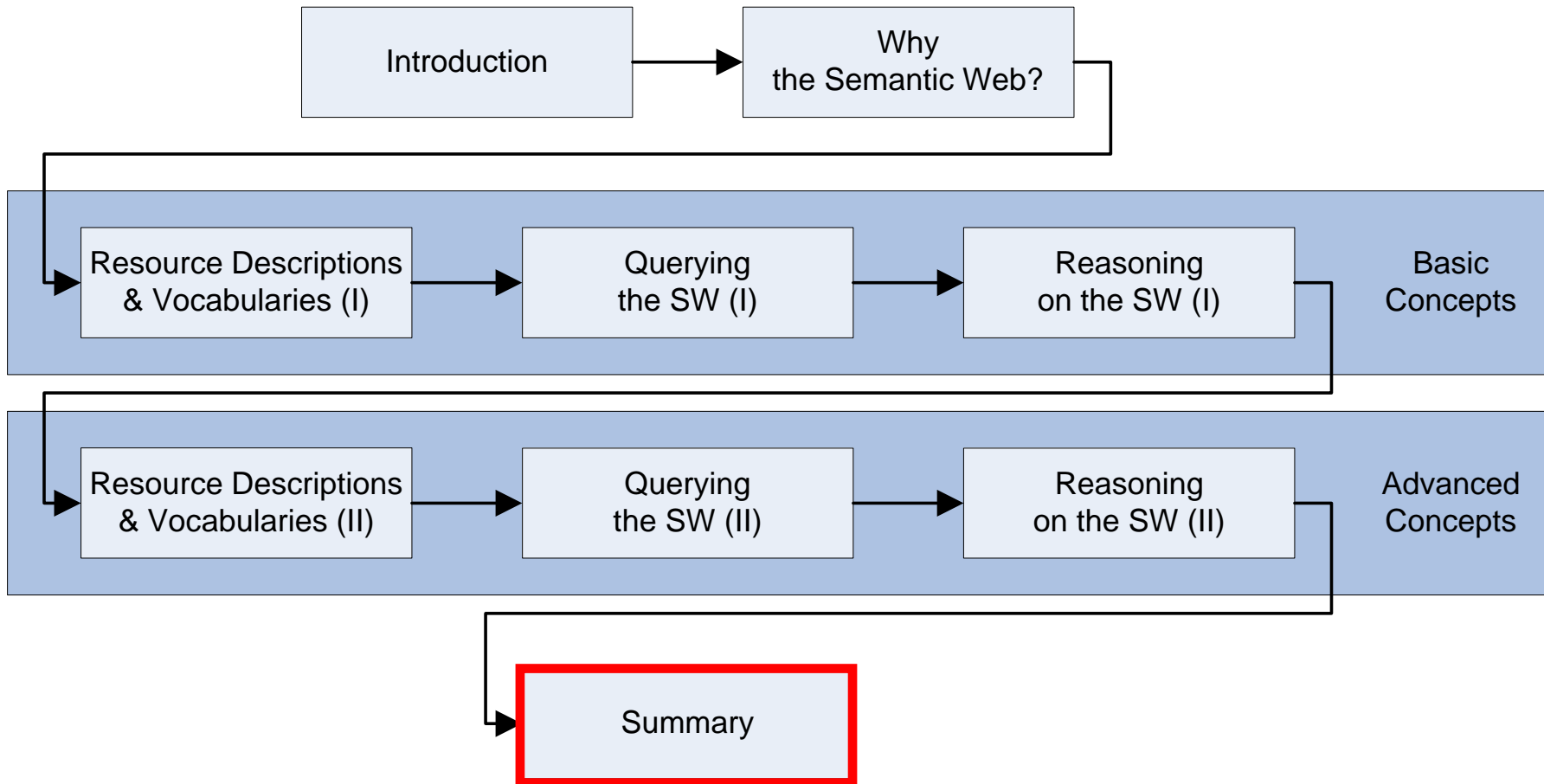
■ ABOX

- "assertion component"—a fact associated with a terminological vocabulary
- associated with instances of classes
- $\text{myTiger} \in \text{Carnivore}$

Together ABox and TBox statements make up a knowledge base.

Outline

Lecture overview



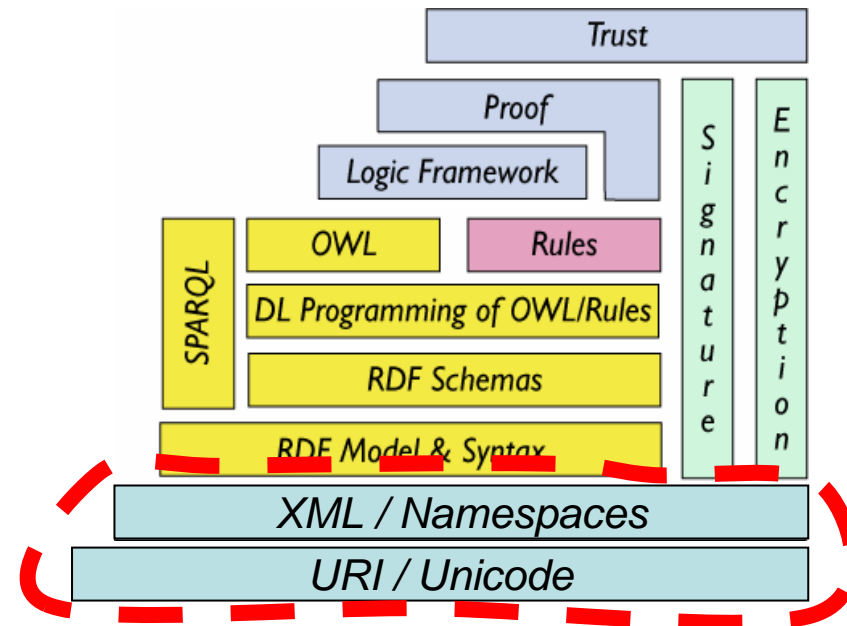
Summary

RDF foundations



Share basic syntax with other Web standards

- URI: unique identifiers
- Namespaces: organize/group identifiers
- XML: reuse syntax and data types



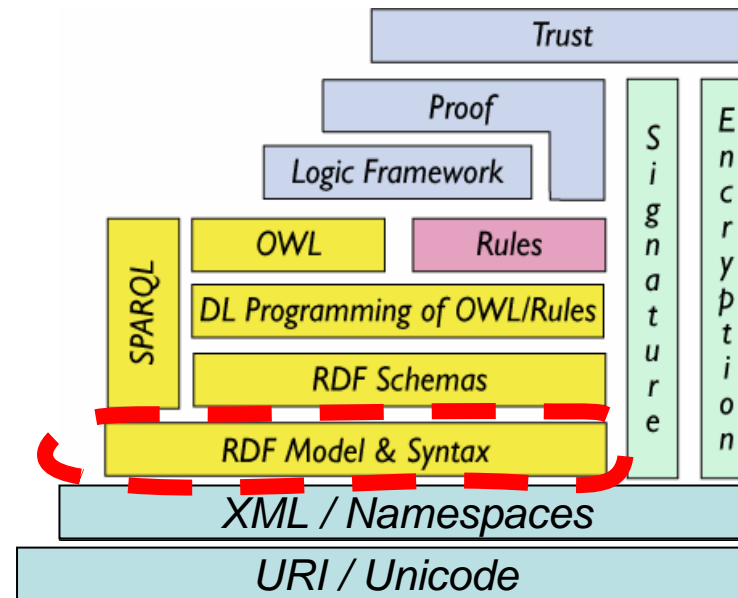
Summary

RDF Model



Data model facility

- Evolution of hyperlinks
- Open, extensible (open world assumption)
- Graph model
- Easy interconnection of distributed data



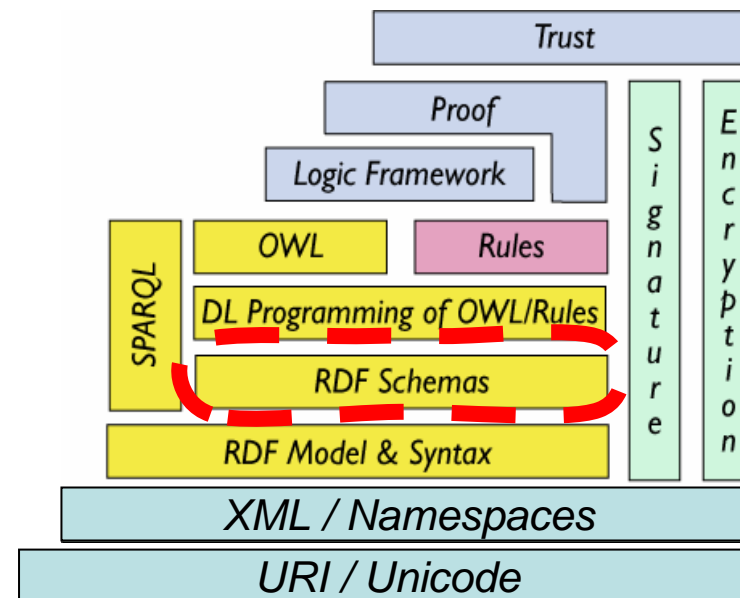
Summary

RDF Schema



Facility for shared vocabulary

- Properties to share link types
- Classes to share resource types

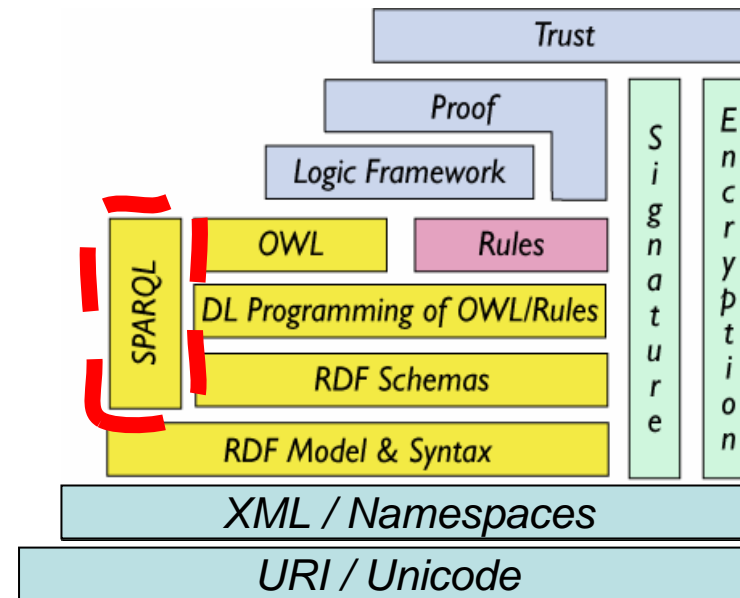


Summary SPARQL



Querying facility

- Flexible pattern matching
- No reasoning
 - Some reasoning support via entailment regime



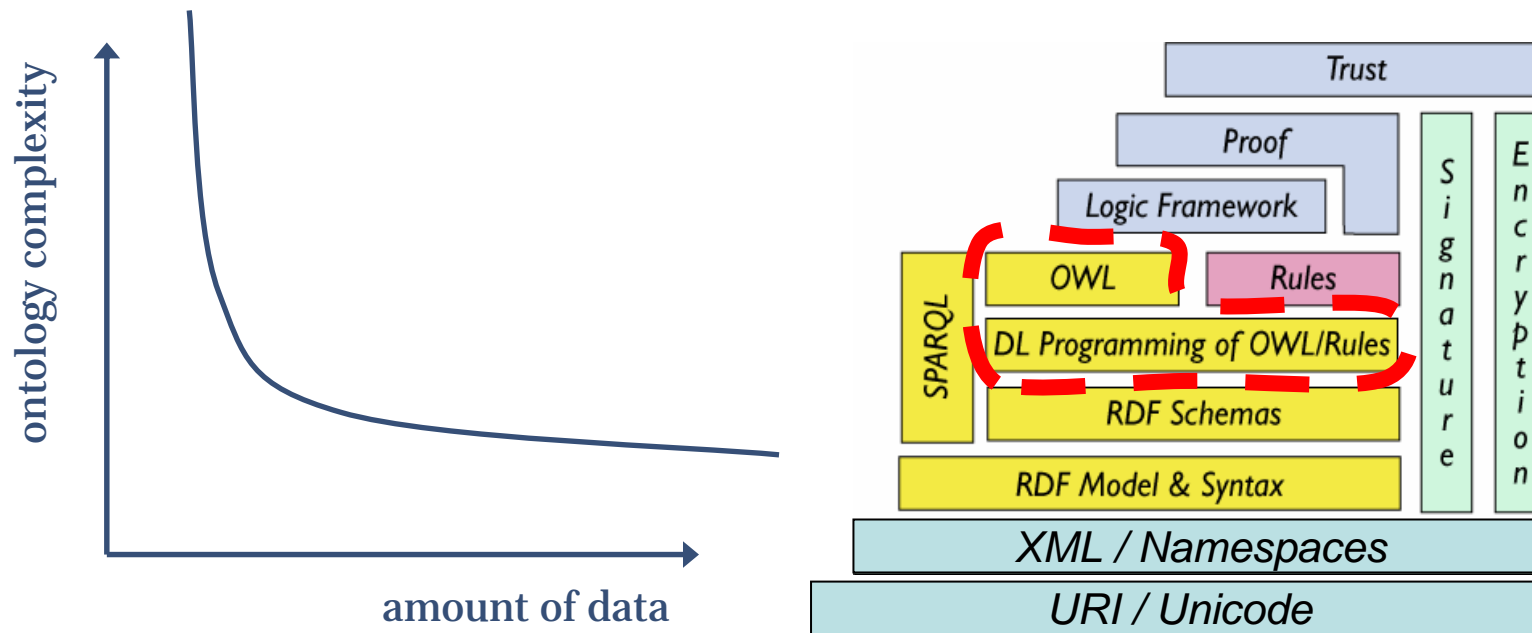
Summary

OWL / Description Logic



Reasoning facility

- Support complex ontology models
- Reasoning on class and instance level



Summary

Comparison to other techniques



	Rel. DB	XML	SemWeb
Data model	Table	Tree	Graph
Maturity	++	+	o
Scalability	++	+	o/-
Schema expressivity	o	+	+ (RDFS) ++ (OWL/DL)
Schema extensibility	-	+	++
Data inter-connection	- (CWA)	- (CWA)	+ (OWA)

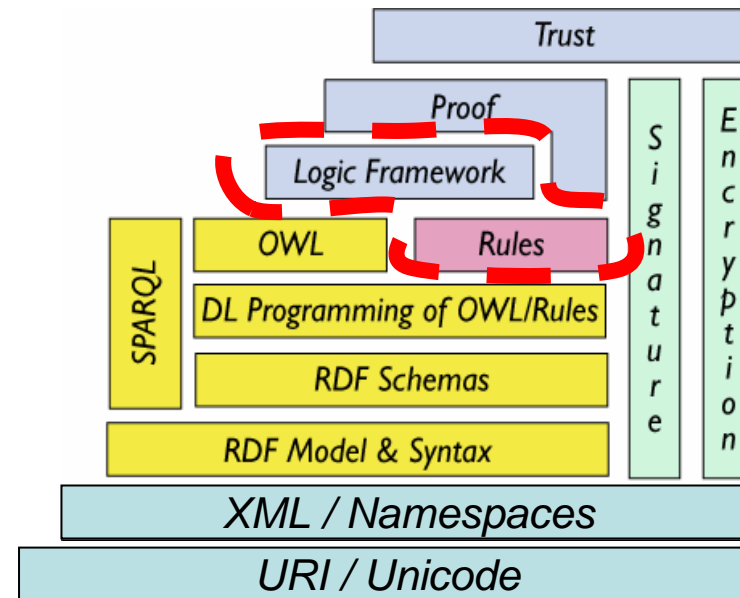
Summary

Future Developments



Support rules + ontologies

Support ,model context‘



References

Specifications



■ RDF Primer

http://www.w3.org/TR/rdf_primer/

■ RDF Vocabulary Description Language 1.0: RDF Schema

http://www.w3.org/TR/rdf_schema/

■ OWL Web Ontology Language Overview

http://www.w3.org/TR/owl_features/

■ SPARQL Query Language for RDF

http://www.w3.org/TR/rdf_sparql_query/



- An introduction to Description Logics

 - www.inf.unibz.it/~franconi/dl/course/dlhb/dlhb-01.pdf

- A Practical Guide to Building OWL Ontologies Using the Protege-OWL Plugin and CO-ODE Tools

 - <http://www.co-ode.org/resources/tutorials/ProtegeOWLTutorial.pdf>

- Some Other Tutorials

 - Nicola Henze's Semantic Web course

 - <http://www.kbs.uni-hannover.de/~henze/semweb05/>

 - Jos de Bruijn's Semantic Web Technologies course

 - <http://www.debruijn.net/teaching/swt/>

Thanks!



Questions?

olmedilla@L3S.de – <http://www.L3S.de/~olmedilla/>
siberski@L3S.de – <http://www.L3S.de/~siberski/>