Karlsruhe Institute of Technology

Semantic Web Modelling Languages (Part 1)

Tutorial at IJCAI2009
July 13, 2009

Forschungszentrum Karlsruhe
in der Helmholtz-Gemeinschaft

Universität Karlsruhe (TH)
Research University · founded 1825

Pascal Hitzler 

Markus Krötzsch 

Sebastian Rudolph 

AIFB, Universität Karlsruhe (TH)

http://www.pascal-hitzler.de
http://korrekt.org
http://www.sebastian-rudolph.de

Tutorial Outline

- RDF Sebastian (90 min)
- coffee break
- OWL Pascal (120 min)

Karlsruhe Institute of Technology

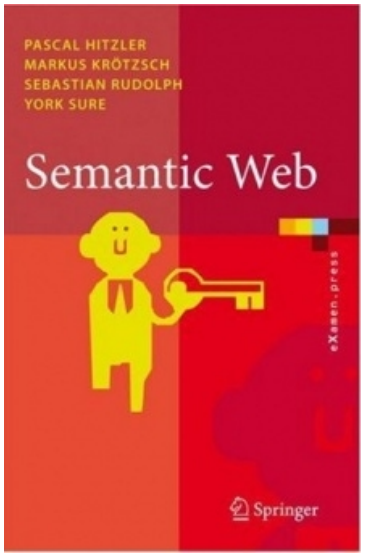
Full set of slides available from

http://www.semantic-web-book.org/page/IJCAI-09_Tutorial

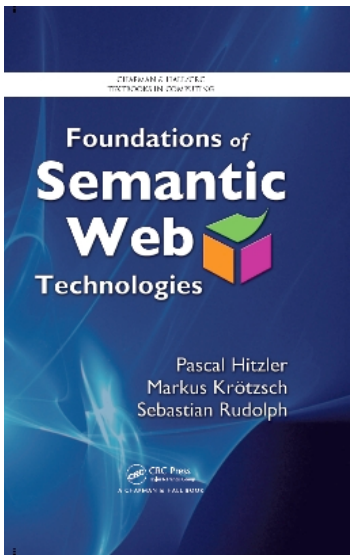
References

- W3C Specification: http://www.w3.org/RDF/

- Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, York Sure, Semantic Web – Grundlagen. Springer, 2008. http://www.semantic-web-grundlagen.de/ (In German.)



- Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, Foundations of Semantic Web Technologies. Chapman & Hall/CRC, 2009. http://www.semantic-web-book.org/wiki/FOST (Grab a flyer from us.)



Outline

- Graph-Based Knowledge Representation
- RDF Building Blocks and Turtle Syntax
- Model Theory for RDF
- RDF Schema
- RDFS Entailment
- Shortcomings of RDF

Why Semantic Web Modelling?

- Initially, the Web was made for humans reading webpages.
- But there‘s too much information out there to be entirely checked by a human with a specific information need.
- Machines can process large amounts of data.
- Normal Web data (such as HTML) is not suitable for content-sensitive machine processing (ambiguous, relies on background knowledge, etc.)
- Semantic Web is concerned with representing information distributed across the Web in a machine-interpretable way.

- So, why not use XML?

Shortcomings of (Pure) XML

- **Task: express ”The Book ’Foundations of Semantic Web Technologies' is published at CRC Press.”**
- **many options:**

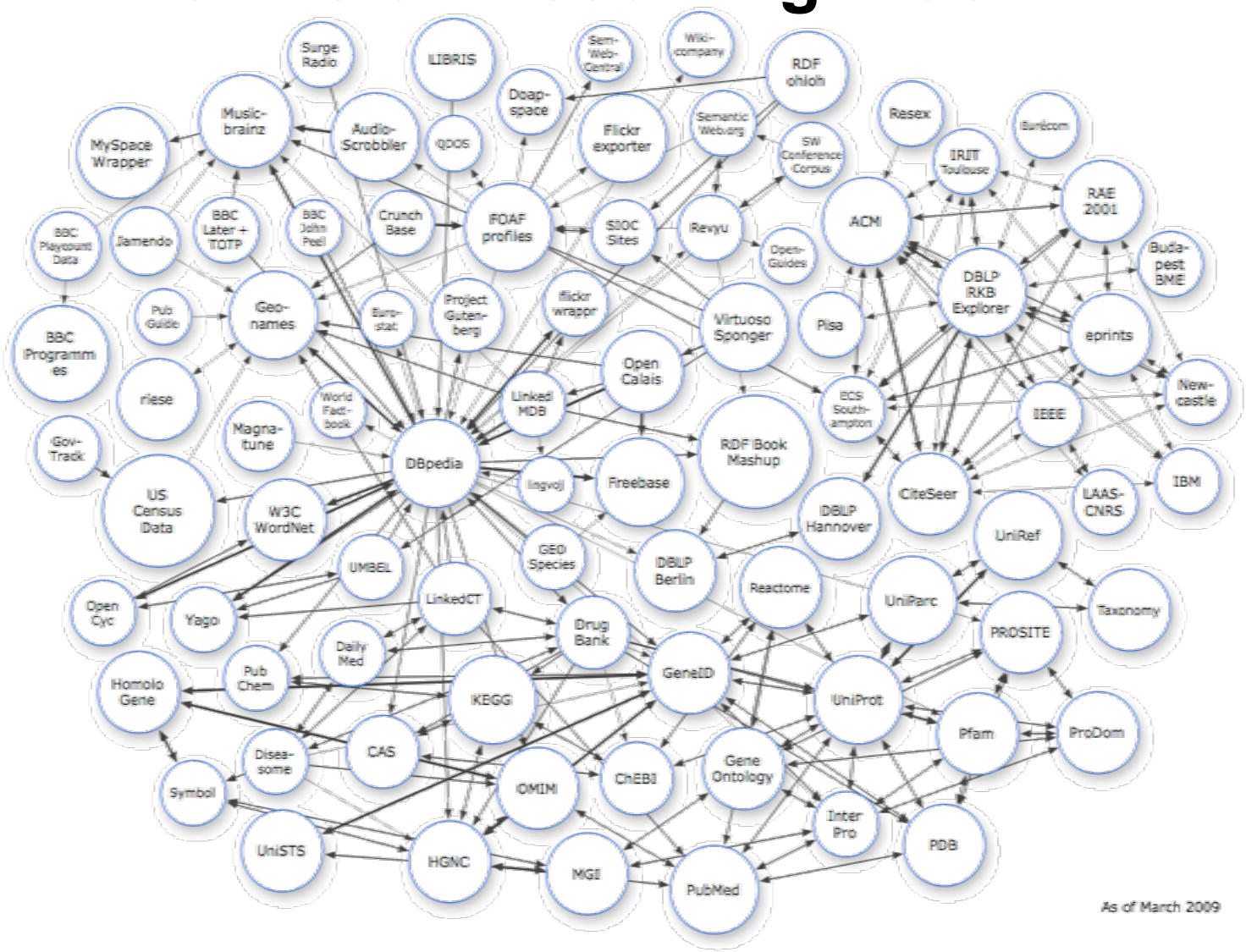
```
<published>
<publisher>CRC Press</publisher>
<book>Foundations of Semantic Web Technologies</book>
</published>
```

```
<publisher name="CRC Press">
<published book=”Foundations of Semantic Web Technologies/>
</publisher>
```

```
<book name="Foudations of Semantic Web Technologies">
<published publisher="CRC Press”/>
</book>
```

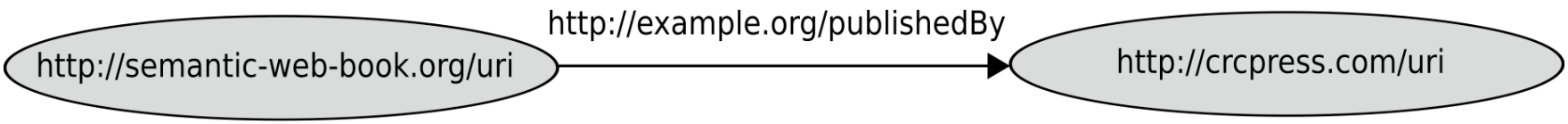
- **ambiguity and tree structure inappropriate for intended purpose**

Web-Wide Linked Open Data – The Vision Becoming True



RDF: Graphs instead of Trees

- **Solution: representation by directed graphs**

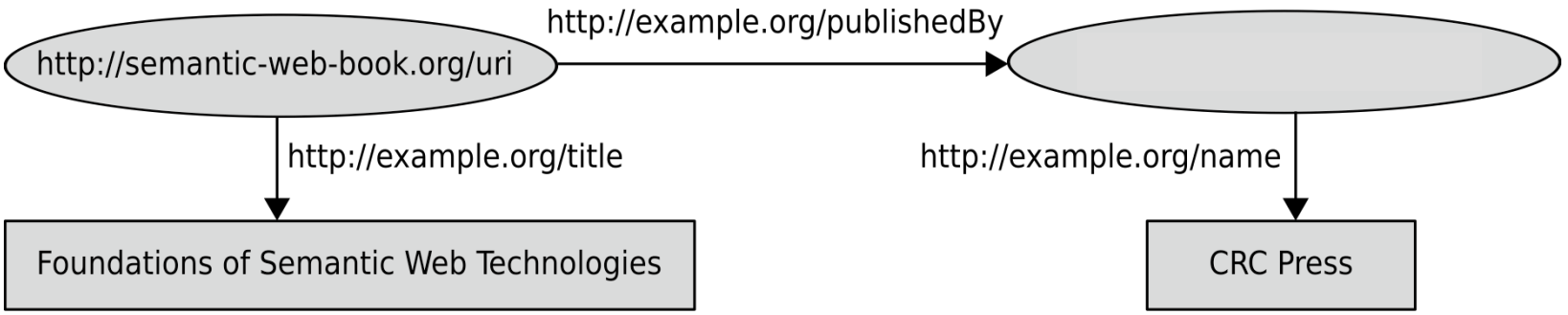


RDF

- “Resource Description Framework”
- W3C Recommendation (http://www.w3.org/RDF)
- RDF is a data model (not one specific syntax)
  - originally designed for providing metadata for Web resources, later used for more general purposes
  - encodes structured informationen
  - universal machine-readable exchange format

Building blocks for RDF Graphs

- URIs
- literals
- blank nodes (aka: empty nodes, bnodes)



URIs - Idea

- URI = Uniform Resource Identifier
- allow for denoting resources in a world-wide unambiguous way
- a resource can be any object that possesses a clear identity (within the context of a given application)
- examples: books, cities, humans, publishers, but also relations between those, abstract concepts, etc.
- already realized in some domains: e.g., ISBN for books

URIs - Syntax

- Builds on concept of URLs but not every URI refers to a Web document
  (but often the URL of a document is used as its URI)
- URI starts with so-called URI schema separated from the following part by ":" (e.g, **`http`**, **`ftp`**, **`mailto`**)
- mostly hierarchical internal structure

Self-defined URIs

- necessary if no URI exists (yet) for a resource (or it is not known)
- strategy for avoiding unwanted clashes: use **`http`** URIs of webspace you control
- this also allows you to provide some documentation about the URI
- How to distinguish URI of a resource from URI of the associated documents describing it?
- Example: URI for "Othello”
  - don’t use:
    `http://de.wikipedia.org/wiki/Othello`
  - rather use:
    `http://de.wikipedia.org/wiki/Othello#URI`

Literals

- used for representing data values
- written down as strings
- interpreted via assigned *datatype*
- literals without explicitly associated datatype are treated like strings

Bnodes

- **used to state existence of an entity the reference of which is not known**

Graphs as Triple Sets

- there are several ways for representing graphs
- in RDF we see graphs as set of vertex-edge-vertex triples

Graphs as Triple Sets

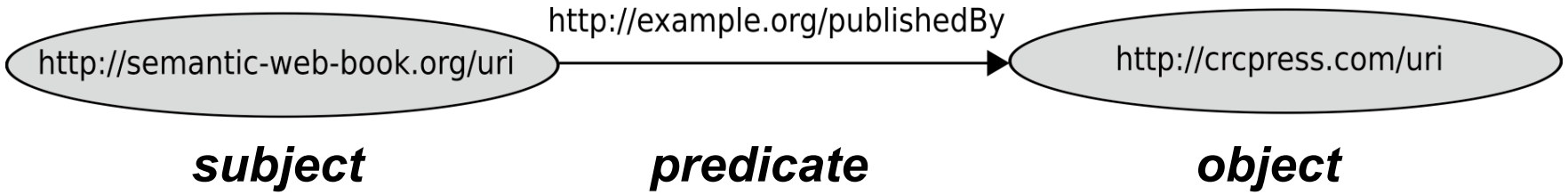
- there are several ways for representing graphs
- in RDF we see graphs as set of vertex-edge-vertex triples

Graphs as Triple Sets

- there are several ways for representing graphs
- in RDF we see graphs as set of vertex-edge-vertex triples

RDF Triples

- constitutents of an RDF triple



- terms inspired by linguistics but doesn't always coincide

- eligible instantiations:
  subject : URI or bnode
  predicate : URI
  object : URI or bnode or literal

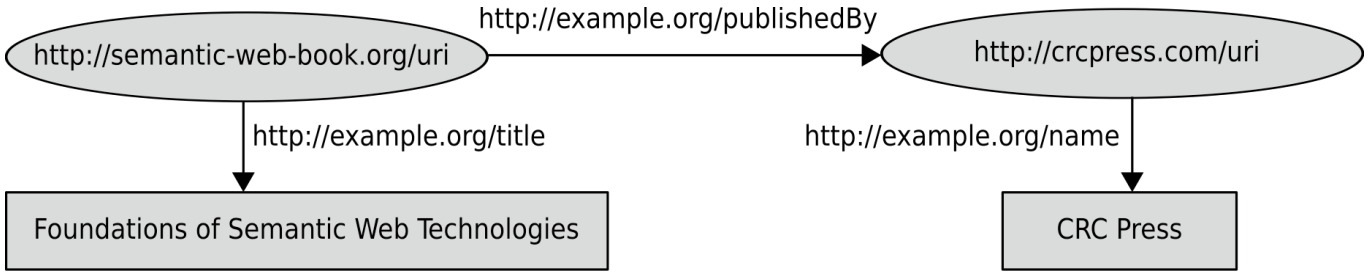
Turtle - An Easy Syntax for RDF

Karlsruhe Institute of Technology

Turtle notation:

- **unabbreviated URIs in <…>**
- **literals in "…"**
- **period at the end of each triple**
- **extra spaces and linebreaks outside of names irrelevant**

```
<http://semantic-web-book.org/uri>  <http://example.org/publishedBy>
                <http://crcpress.com/uri> .
<http://semantic-web-book.org/uri>  <http://example.org/title>
                "Foundations of Semantic Web Technologies" .
<http://crcpress.com/uri>  <http://example.org/name>  "CRC Press" .
```



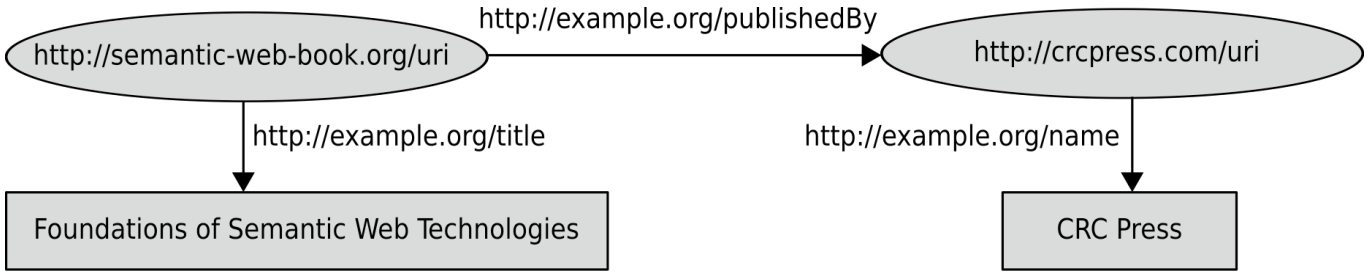
Turtle - An Easy Syntax for RDF

Karlsruhe Institute of Technology

Turtle notation:

- unabbreviated URIs in <…> but can be abbreviated by namespaces
- literals in "…"
- period at the end of each triple
- extra spaces and linebreaks outside of names irrelevant

```
@prefix book: <http://semantic-web-book.org/> .
@prefix ex: <http://example.org/> .
@prefix crc: <http://crcpress.com/> .
book:uri    ex:publishedBy    crc:uri .
book:uri    ex:title          "Foundations of Semantic Web Technologies" .
crc:uri     ex:name           "CRC Press" .
```



Turtle - An Easy Syntax for RDF

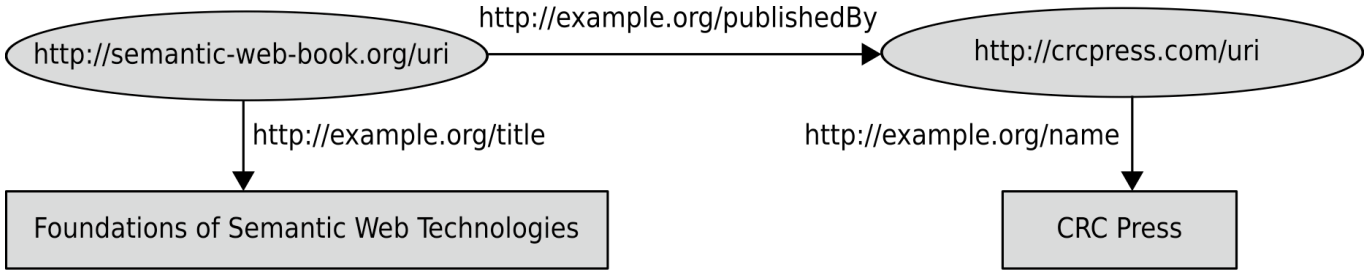
Karlsruhe Institute of Technology

Turtle notation:

- **unabbreviated URIs in <…> but can be abbreviated by namespaces**
- **literals in "…"**
- **period at the end of each triple**
- **extra spaces and linebreaks outside of names irrelevant**

```
@prefix book: <http://semantic-web-book.org/> .
@prefix ex: <http://example.org/> .
@prefix crc: <http://crcpress.com/> .
book:uri      ex:publishedBy    crc:uri ;
              ex:title          "Foundations of Semantic Web Technologies" .
crc:uri       ex:name           "CRC Press" .
```

repeated subjects may be left out



Turtle - An Easy Syntax for RDF

Karlsruhe Institute of Technology

Turtle notation:

- **unabbreviated URIs in <…> but can be abbreviated by namespaces**
- **literals in "…"**
- **period at the end of each triple**
- **extra spaces and linebreaks outside of names irrelevant**

```
@prefix book: <http://semantic-web-book.org/> .
@prefix ex: <http://example.org/> .
@prefix crc: <http://crcpress.com/> .
book:uri      ex:publishedBy    crc:uri ;
              ex:title          "Foundations of Semantic Web Technologies" ;
              ex:author         book:Hitzler, book:Krötzsch, book:Rudolph .
crc:uri       ex:name           "CRC Press" .
```

repeated subjects may be left out

several objects can be assigned to the same subject-predicate pairs

XML-Syntax of RDF

- there is also an XML syntax for RDF
- it's for machines, so we don't deal with it here

```
<rdf:Description rdf:about="http://semantic-web-book.org/uri">
    <ex:title>Foundations of Semantic Web Technologies</ex:title>
    <ex:publishedBy>
        <rdf:Description rdf:about="http://crcpress.com/uri">
            <ex:name>CRC Press</ex:name>
        </rdf:Description>
    </ex:publishedBy>
</rdf:Description>
```

Datatypes in RDF

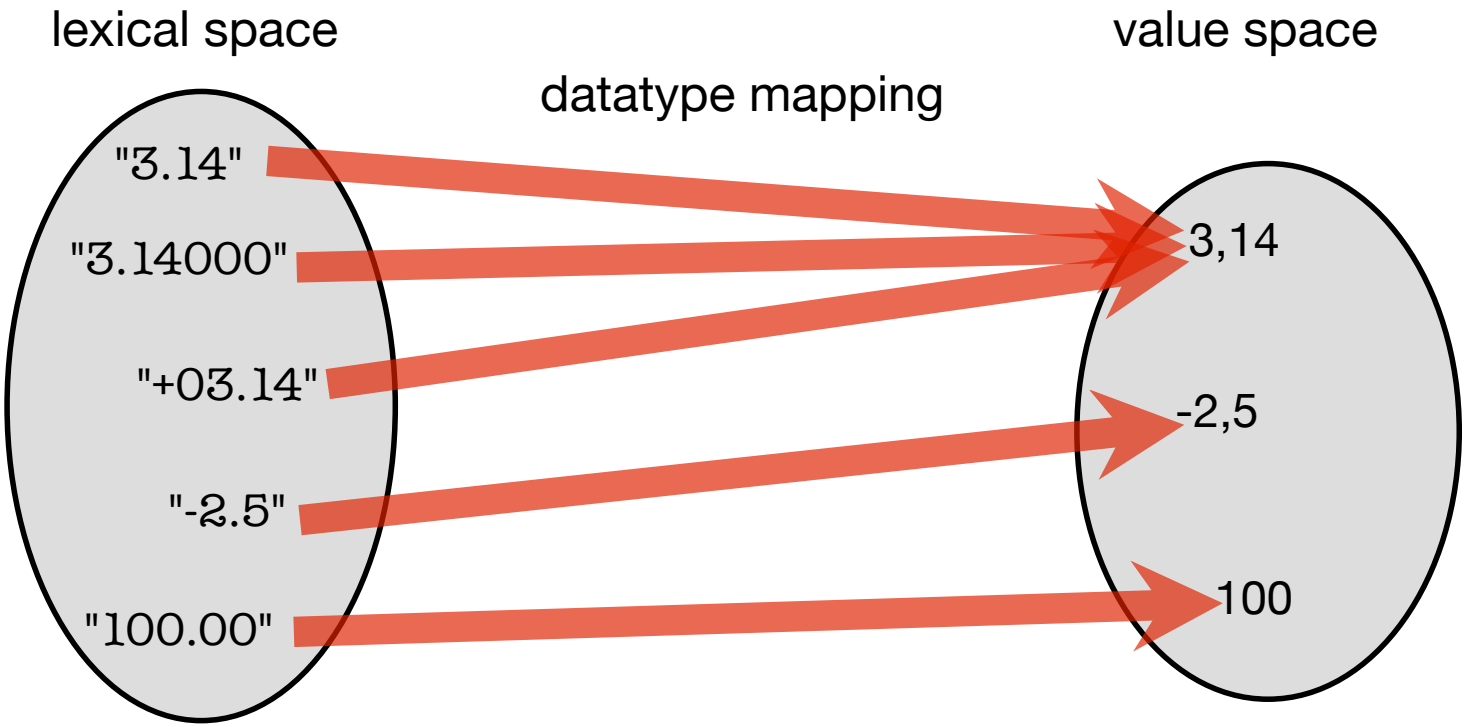
- by now: literals were untyped, interpreted as strings (i.e.: "02"<"100"<"11"<"2")
- typing literals with datatypes allows for more adequate (semantic = meaning-appropriate) treatment of values
- datatypes denoted by URIs and can be freely chosen
- frequently: xsd datatypes from XML
- syntax of typed literal:

  "datavalue"^^datatype-URI

- rdf:XMLLiteral is the only datatype that is part of the RDF standard, denotes arbitrary balanced XML “snippets”

Datatypes – the Abstract View

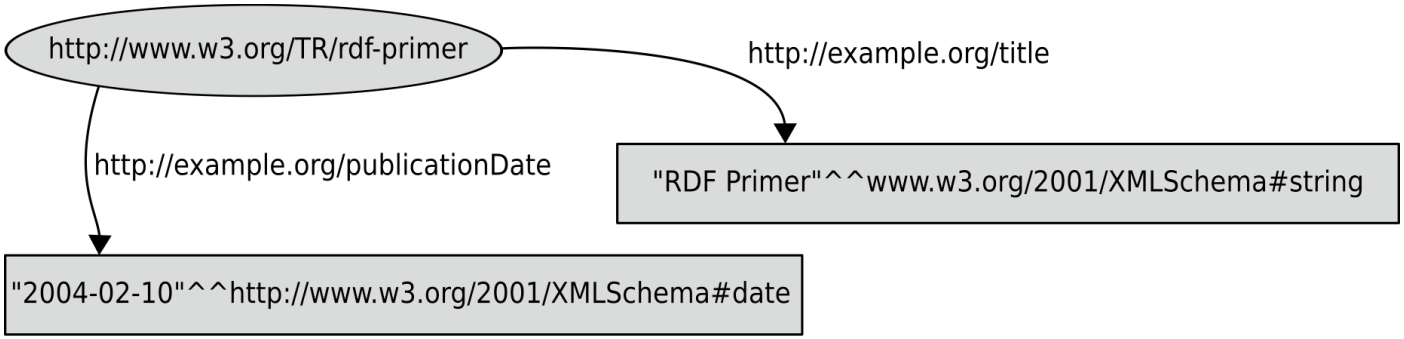
- **Example: xsd:decimal**



"3.14"="+03.14" holds for xsd:decimal **but not for** xsd:string

Datatypes in RDF – Example

- **Graph:**



- **Turtle:**

```
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
<http://www.w3.org/TR/rdf-primer>
<http://example.org/title> "RDF Primer"^^xsd:string ;
<http://example.org/publicationDate> "2004-02-10"^^xsd:date .
```

Language Settings and Datatypes

- **language settings only applicable to untyped literals**

<http://www.w3.org/TR/rdf-primer> <http://example.org/title>
"Initiation à RDF"@fr, "RDF Primer"@en .

- **distinct types or language settings – distinct literals**

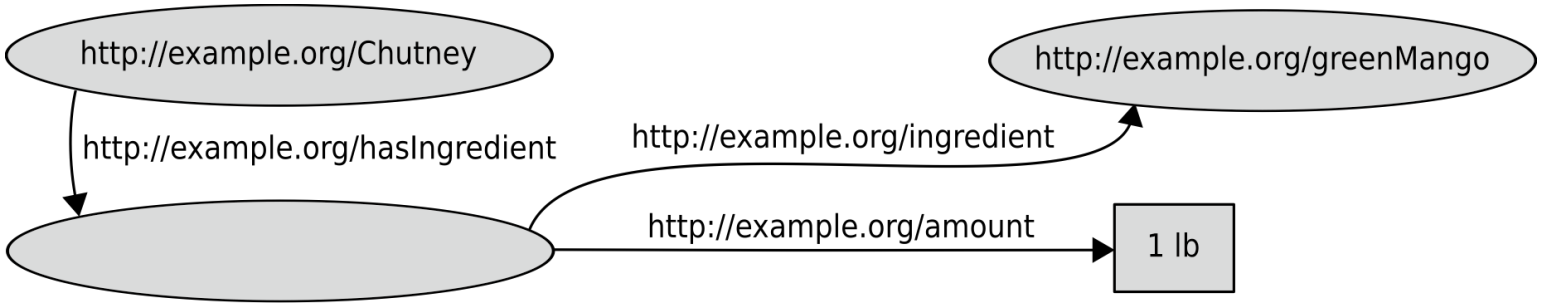
<http://crcpress.com/uri> <http://example.org/Name>
"CRC Press" ,
"CRC Press"@en ,
"CRC Press"^^xsd:string .

n-ary Relationships

- **Cooking with RDF:**
  "*For the preparation of Chutney, we need the following: 1 lb green mango, 1 tsp. Cayenne pepper, ...*"

dish	ingredient	amount
chutney	green mango	1 lb
chutney	cayenne pepper	1 tsp.

- **solved by auxiliary nodes (may be blank)**



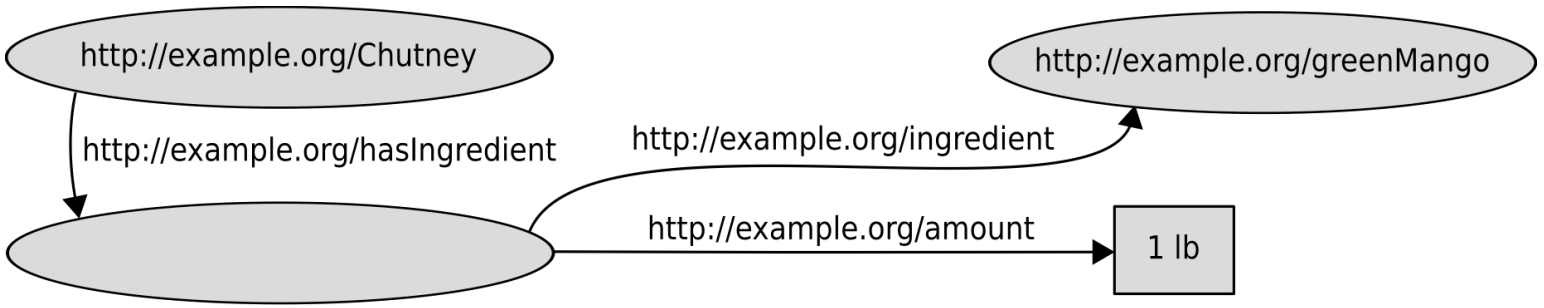
n-ary Relationships

- **Turtle version 1:**

```
@prefix ex: <http://example.org/> .
ex:Chutney ex:hasIngredient _:id1 .
_:id1 ex:ingredient ex:greenMango; ex:amount "1lb" .
```

- **Turtle version 2:**

```
@prefix ex: <http://example.org/> .
ex:Chutney ex:hasIngredient
                [ ex:ingredient ex:greenMango; ex:amount "1lb" ] .
```

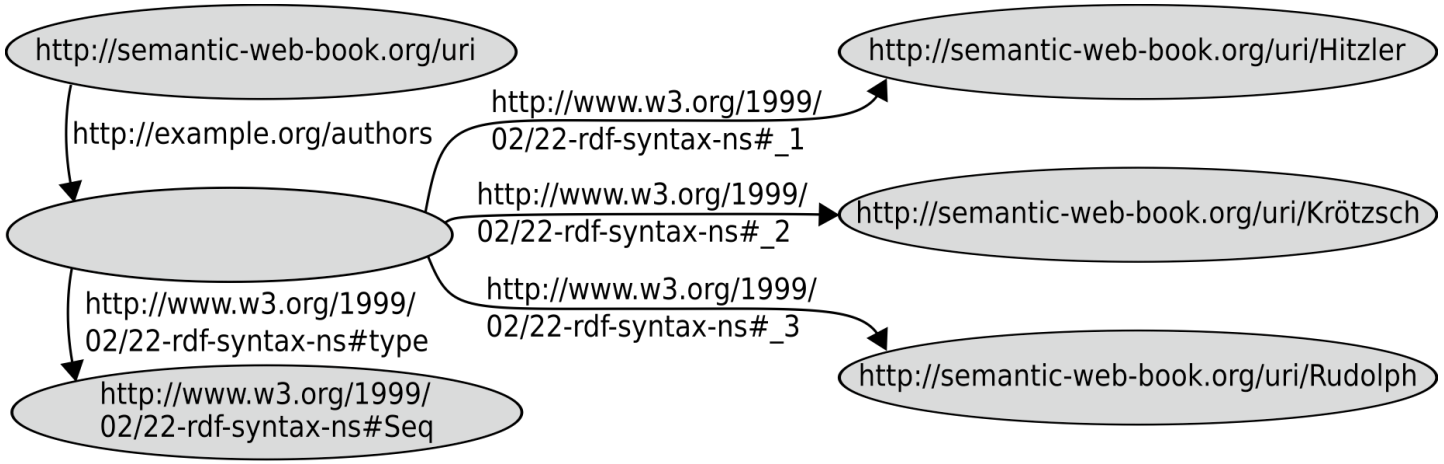


Special Datastructures in RDF

- open lists (containers)
- closed lists (collections)
- reified triples

Open Lists (Container)

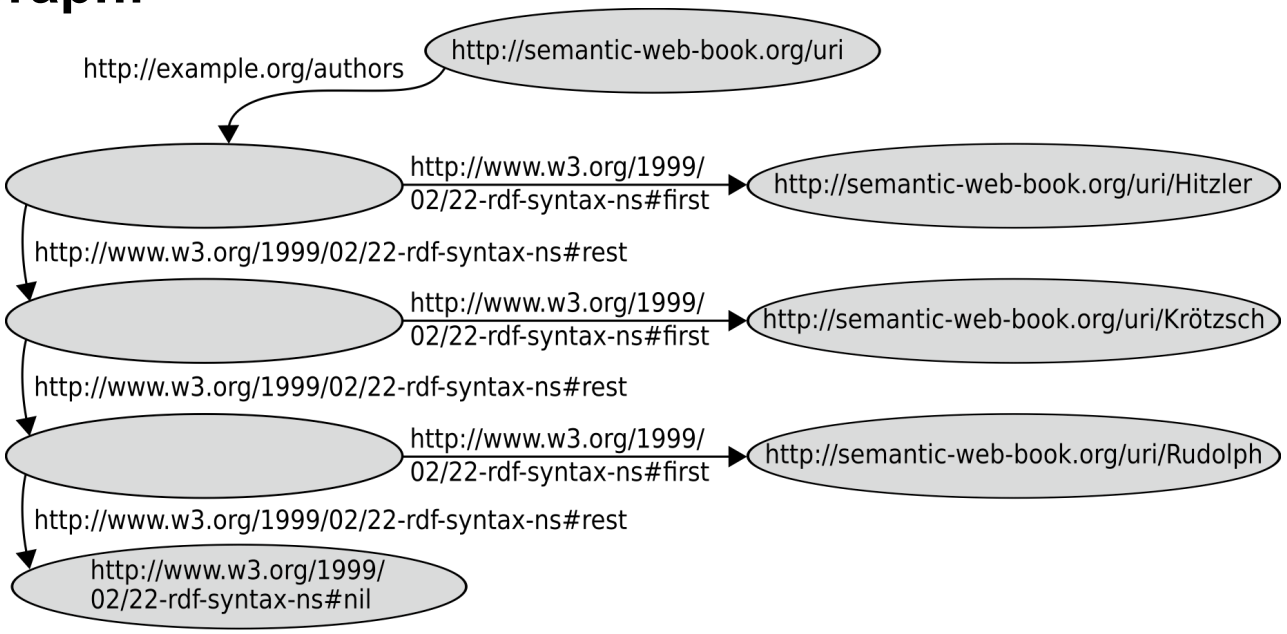
- **Graph:**



- **by rdf:type we assign a list type to the root node**
  - **rdf:Seq – ordered liste (sequence)**
  - **rdf:Bag – unordered list**
  - **rdf:Alt – set of alternatives or choices**

Closed Lists (Collections)

- **Graph:**

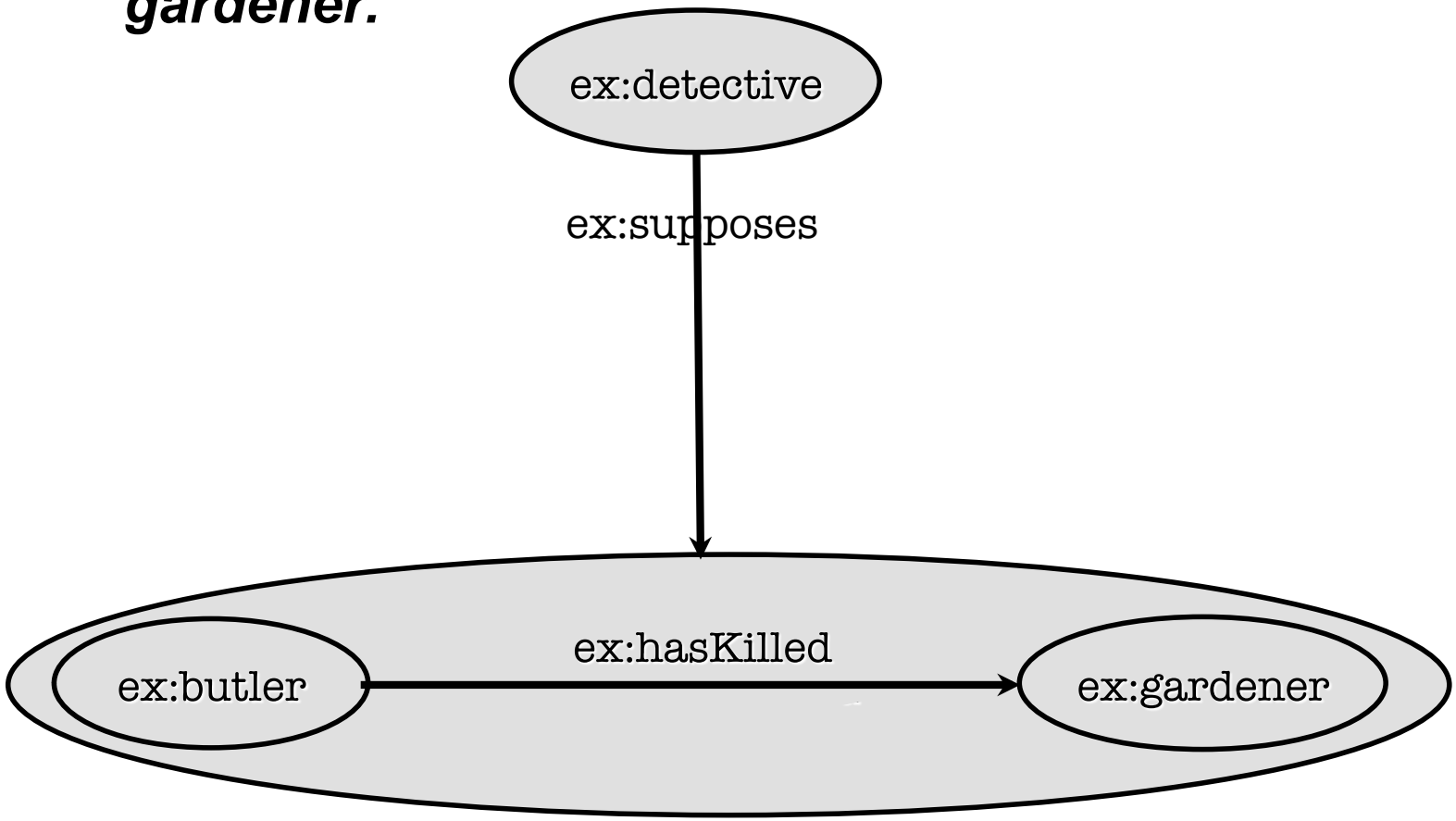


- **Abbreviation for Turtle:**

```
@prefix book: <http://semantic-web-book.org/> .
book:uri <http://example.org/authors>
   ( book:uri/Hitzler book:uri/Krötzsch book:uri/Rudolph ) .
```

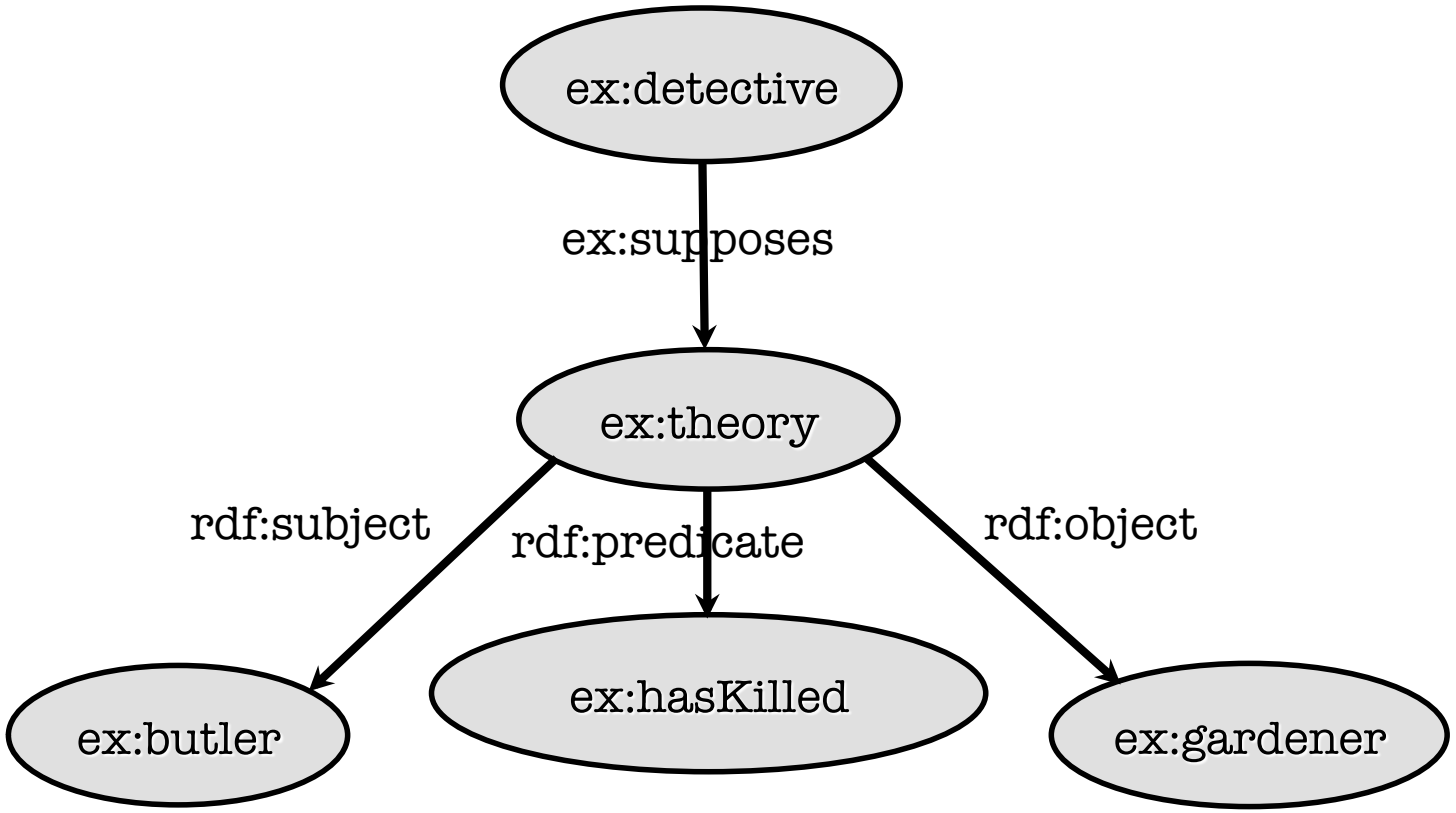
Reification

- **How to model propositions about propositions such as: *„The Detective supposes that the butler killed the gardener.“***



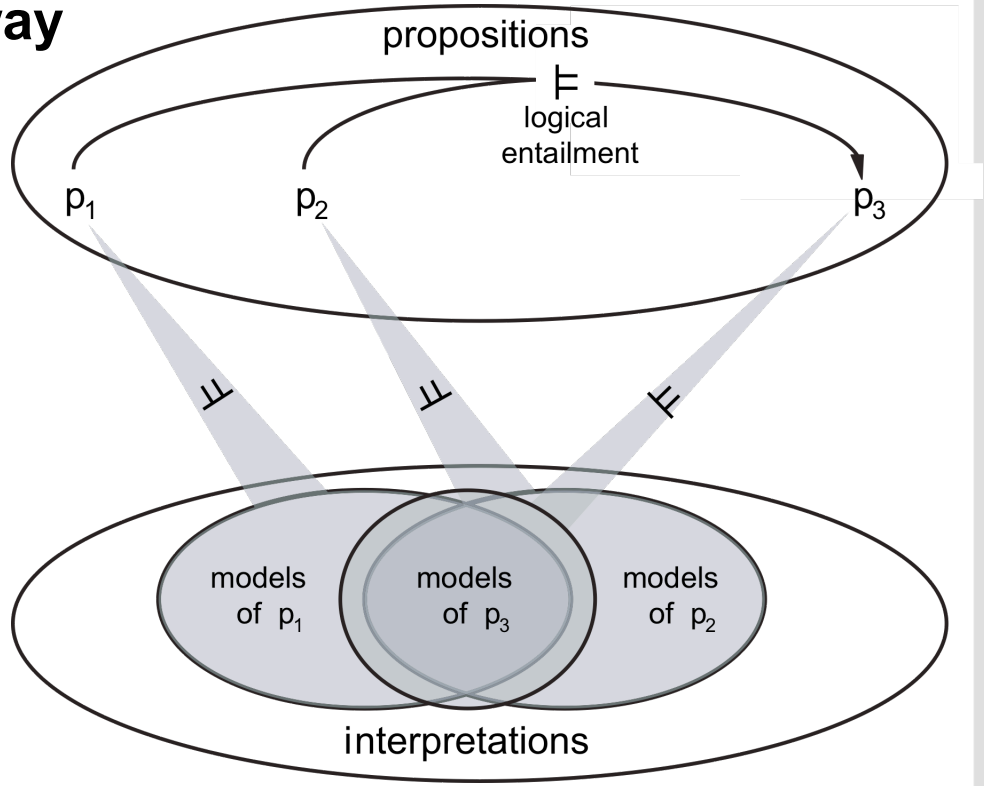
Reification

- **Solution: auxiliary node for nested proposition**



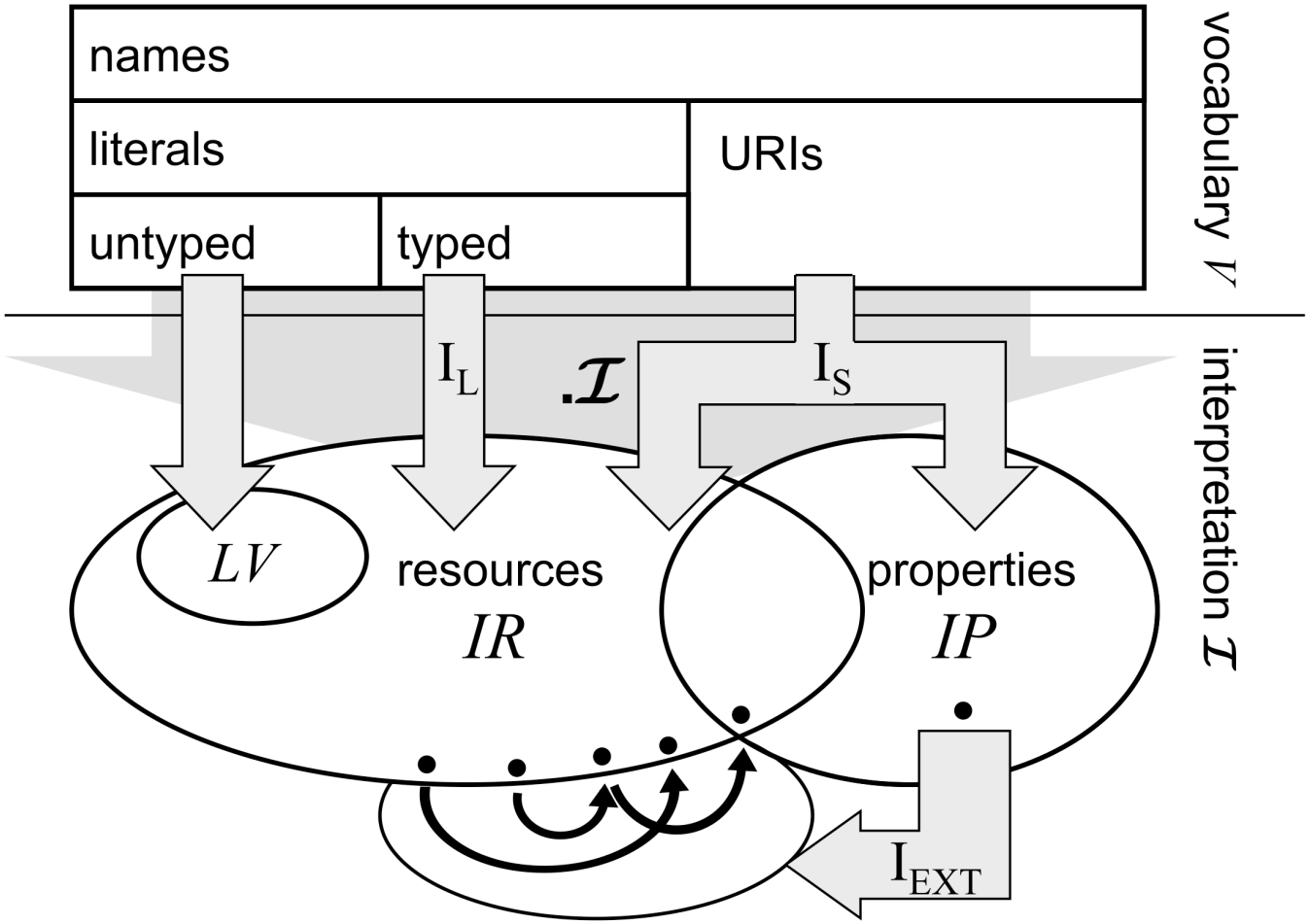
Simple Semantics

- RDF is focused on information exchange and interoperability
- answers of RDF tools to entailment queries should coincide
- therefore, formal semantics needed
- defined in a model-theoretic way



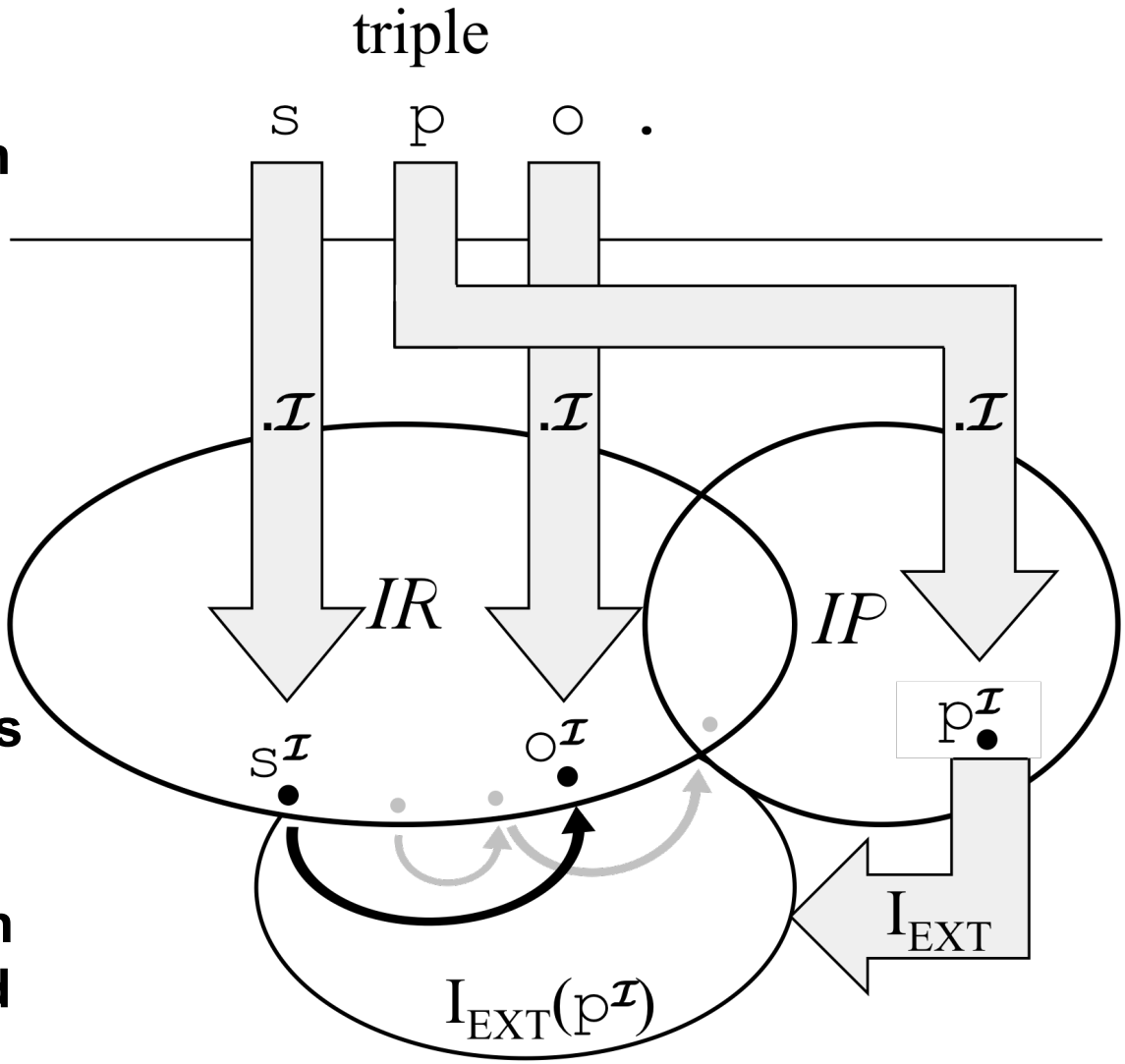
Simple Semantics

- **Interpretation in RDF:**



Simple Semantics

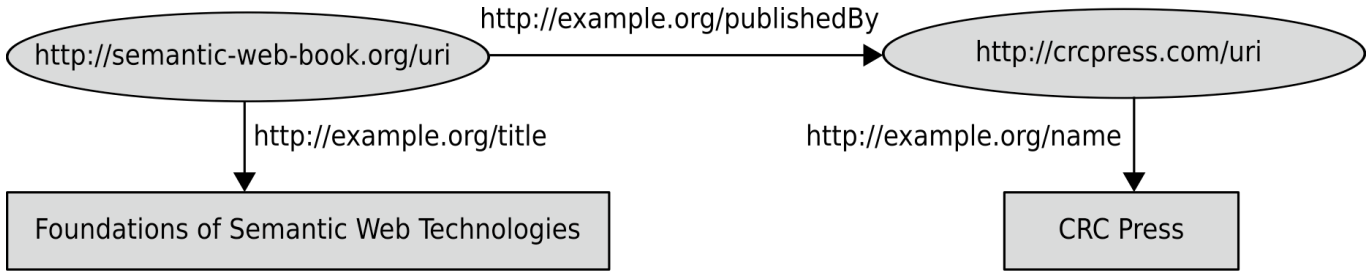
- when is a triple valid in an interpretation?
- a graph is valid, if all its triples are
- this settles the case for „grounded“ graphs
- graph with blank nodes is valid if they can be mapped to elements such that the condition on the right is satisfied



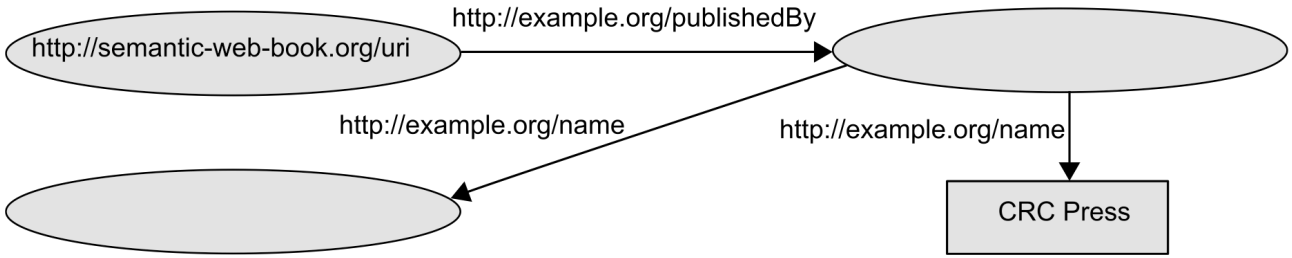
Simple Entailment

- this model theory defines simple entailment
- this is essentially graph matching with bnodes being wildcards

Example: the graph

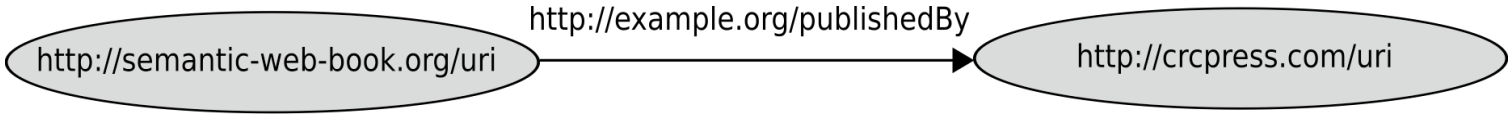


simply entails the graph



Schema Knowledge with RDF(S)

- RDF allows for specification of factual data



- = propositions about single resources (individuals) and their relationships
- desirable: propositions about generic groups of individuals, such as the class of publishers, of organizations, or of persons
- in database terminology: *schema knowledge*
- RDF Schema (RDFS): part of the RDF W3C recommendation

Classes and Instances

```
book:uri  rdf:type  ex:Textbook .
```

- **characterizes the specific book as an instance of the (self-defined) class of textbooks**
- **class-membership not exclusive:**

```
book:uri  rdf:type  ex:Enjoyable .
```

- **URIs can be typed as class-identifiers:**

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

Subclasses

- **we want to express that every textbook is a book, e.g., that every instance of the class** ex:Textbook **is "automatically" recognized as an instance of the class** ex:Book
- **realized by** rdfs:subClassOf **property:**

```
ex:Textbook  rdfs:subClassOf  ex:Book .
```

- rdfs:subClassOf **is defined to be transitive and reflexive**
- **rule of thumb:**

rdf:type	**means**	$\in$
rdfs:subClassOf	**means**	$\subseteq$

Properties

- technical term for Relations, Correspondencies
- Property names usually occur in predicate position in factoid RDF triples
- characterize, how two resources are related
- mathematically: set of pairs:
  maried_with = {(Adam,Eve),(Brad,Angelina),...}
- URI can be marked as property name by typing it accordingly:

```
ex:publishedBy  rdf:type  rdf:Property  .
```

Subproperties

- **in analogy to subclass relationships**
- **representation in RDFS via** rdfs:subPropertyOf **e.g.:**

```
ex:happilyMarriedWith  rdf:subPropertyOf  rdf:marriedWith  .
```

- **then, given**

```
ex:Markus  ex:happilyMarriedWith  ex:Anja  .
```

we can deduce

```
ex:Markus  ex:marriedWith  ex:Anja  .
```

Property Restrictions

- **properties may give hints what types the linked resources have, e.g. we know that** ex:publishedBy **connects publications with publishers**

- **i.e., for all URIs a, b where we know**

  a ex:publishedBy b .

  we want to automatically follow:

  a rdf:type ex:Publication .
  b rdf:type ex:Publisher .

- **this generic correspondency can be encoded in RDFS:**

  ex:publishedBy rdfs:domain ex:Publication .
  ex:publishedBy rdfs:range ex:Publisher .

Property Restrictions

- with property restrictions, semantic interdependencies between properties and classes can be specified
- Caution: property restrictions are interpreted globally and conjunctively, e.g.

```
ex:authorOf  rdfs:range  ex:Cookbook  .
ex:authorOf  rdfs:range  ex:Storybook  .
```

  means: everything which is authored by somebody is both a cookbook and a storybook
- thus: always use most generic classes for domain/ range statements

Additional Information

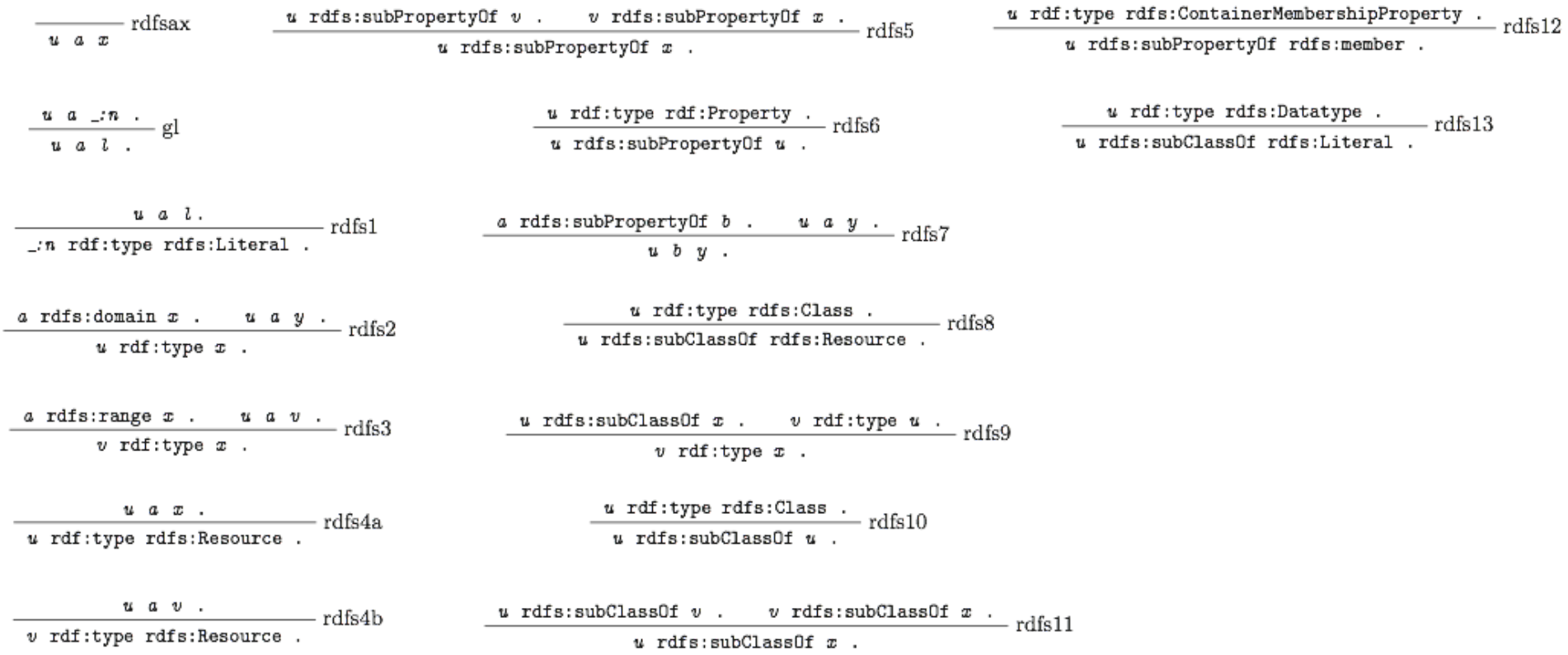
- used to add human-readable information (comments or names)
- for compatibility reasons graph-based representation recommended; set of properties for that purpose:
  - rdfs:label assigns an alternative name (encoded as literal) to an arbitrary ressource
  - rdfs:comment assigns a more comprehensive comment (also literal)
  - rdfs:seeAlso, rdfs:definedBy refer to resources (URIs!) containing further information about the subject resource

RDFS Entailment

- RDFS interpretations take care of RDF(S)-specific vocabulary by imposing additional conditions on simple interpretations:
  - all URIs and bnodes are of type rdf:Resource
  - triple predicates are of type rdf:Property
  - all well-typed and untyped literals are of type rdf:Literal
  - types of triple subjects/objects correspond to rdfs:domain/ rdfs:range statements
  - rdfs:subClassOf and rdfs:subPropertyOf are interpreted reflexive and transitive and “inheriting“
  - well-formed XML-Literals are mapped into LV, ill-formed ones go somewhere else
  - ...and many more

RDFS Entailment – Automation

- **RDFS entailment can be decided via rule-like deduction calculus (NP-complete)**



Deployment of RDF

- today there is a variety of RDF tools
- software libraries for virtually every programming language
- freely available systems for handling large sets of RDF data (so-called RDF stores or triple stores)
- increasingly supported by commercial actors (e.g. Oracle)
- basis for several data formats: RSS 1.0, XMP (Adobe), SVG (vector graphics format)

RDF(S) as Ontology Language?

- RDFS language features allow for modeling certain semantic aspects of a domain of interest
- hence, RDFS can be seen as a *lightweight* ontology language

RDF(S) as Ontology Language?

Karlsruhe Institute of Technology

Shortcomings of RDF(S):

- **"weak" semantics:**

```
ex:speaksWith  rdfs:domain      ex:Homo .
ex:Homo        rdfs:subClassOf  ex:Primates .
```

does not entail

```
ex:speaksWith  rdfs:domain      ex:Primates .
```

- **expressivity: no negative information can be specified, no cardinality, no disjunction…**

Karlsruhe Institute of Technology

Thanks!

http://semantic-web-grundlagen.de/wiki/ESWC09_Tutorial