Web Dynamics

Semantic Web

Dr. Marc Spaniol

Saarbrücken, July 15, 2010

Agenda

- Introduction to the Semantic Web
  - Visions of the Semantic Web
  - The Semantic Web architecture
- Semantic Web languages
  - Addressing languages: URI
  - Modeling languages: XML, RDF, OWL
  - Rule languages: RIF
  - Query languages: SPARQL
- Semantic Web by example
  - Creating RDF data
  - Merging RDF data
  - Querying RDF data

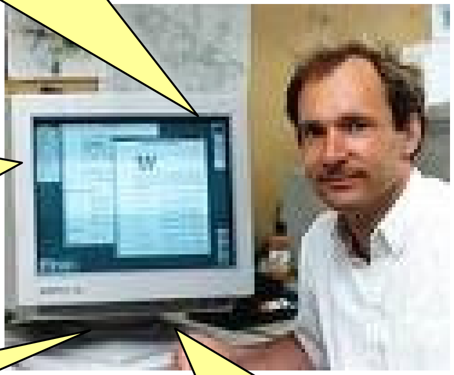
Semantic Web

Dr. Marc Spaniol

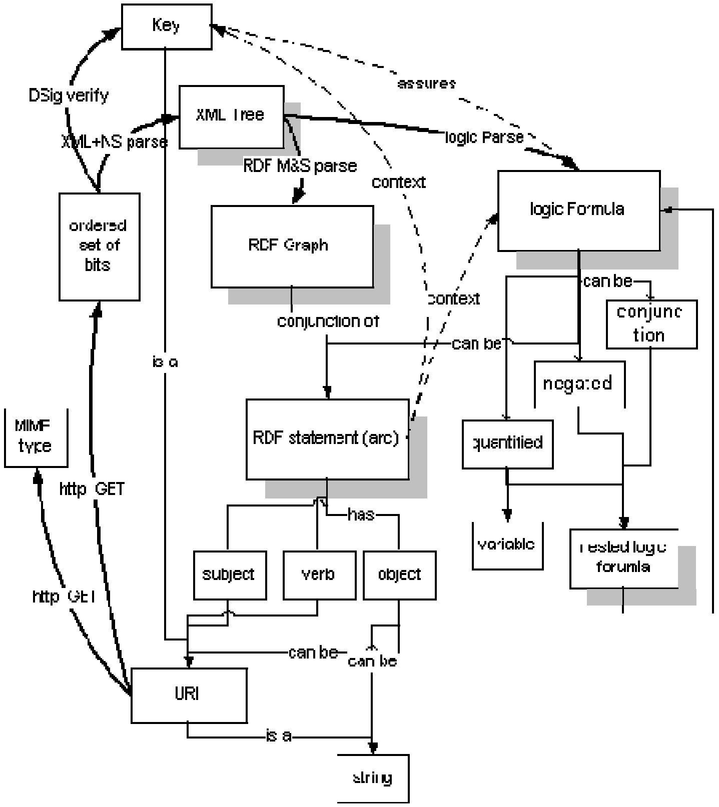
What is the Semantic Web?

The *Semantic Web* is a Web of data, in some ways like a global database
– Tim Berners-Lee

… a universal Web of semantic assertions



… an extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation



Source: http://www.w3.org/DesignIssues/Semantic.html

The Web was designed as an information space, with the goal that it should be useful not only for human-human communication, but also that machines would be able to participate and help.
– Tim Berners-Lee, Semantic Web road map

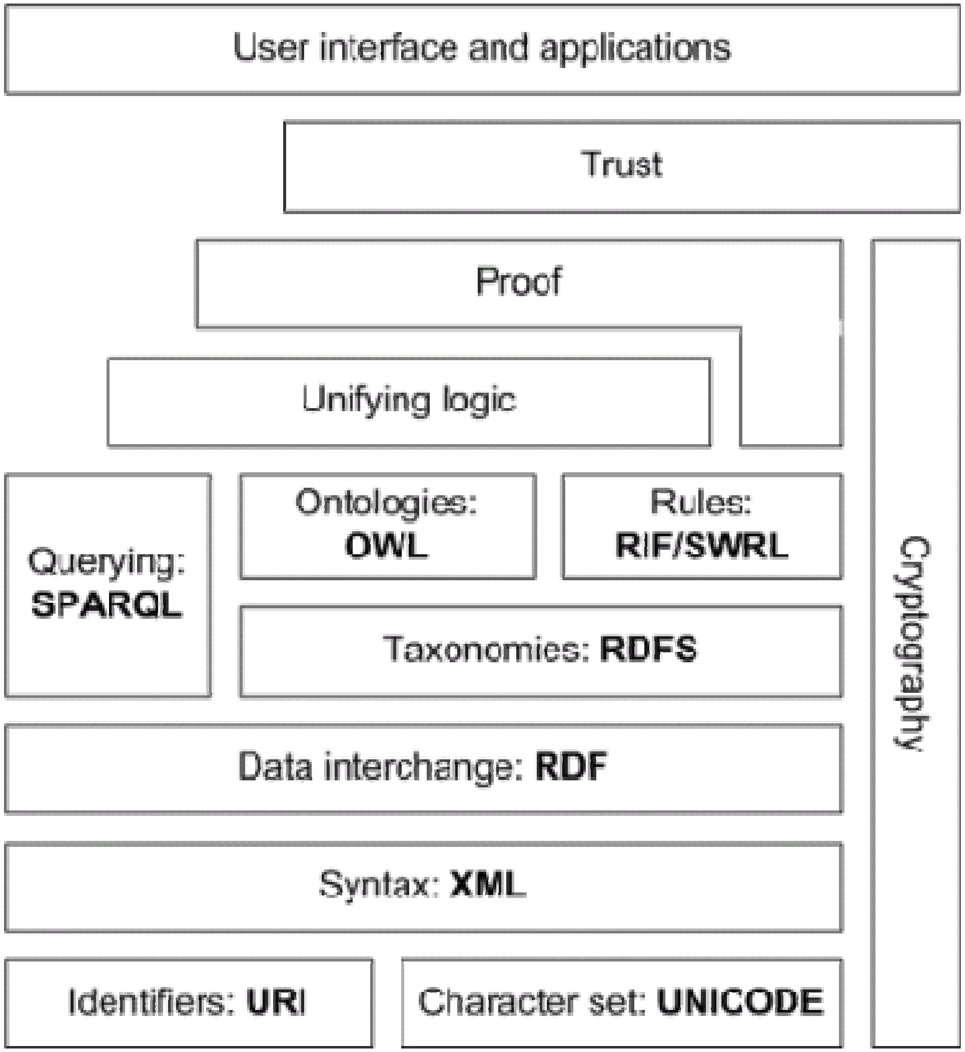
The Semantic Web Roadmap

- Step 1 – Text document and database records
  - Data are application based, while semantics are behind applications, e.g. product catalogue of Amazon
- Step 2 – XML documents with standard vocabulary
  - Data are independent from applications and can be exchanged within certain domains, e.g. Dublin Core
- Step 3 – RDF taxonomy and document with different vocabularies
  - Data are classified by hierarchical structured taxonomies, e.g. “RDF” is a “Semantic Web Language”
- Step 4 – OWL ontologies and automatic reasoning
  - New data can be generated through rule-based reasoning, e.g. automatic cross-domain document transformation

⇒ Goal: Applications are able to understand data

Semantic Web Architecture

- URI
- XML
- RDF and RDFS
- OWL
- RIF and SWRL
- SPARQL



User interface and applications

Trust

Proof

Unifying logic

Querying: **SPARQL**

Ontologies: **OWL**

Rules: **RIF/SWRL**

Taxonomies: **RDFS**

Data interchange: **RDF**

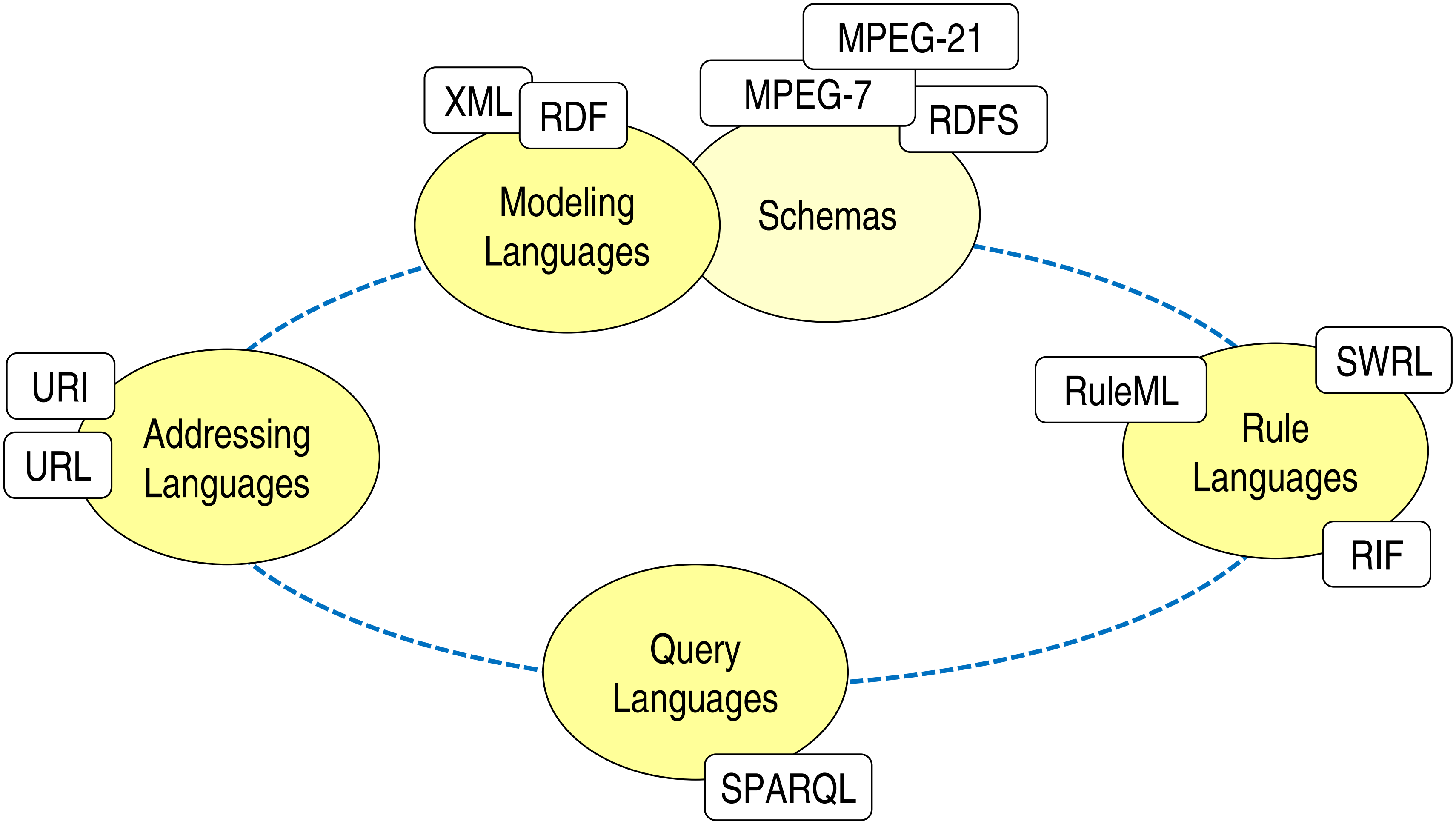
Syntax: **XML**

Identifiers: **URI**

Character set: **UNICODE**

Cryptography

Semantic Web Languages



Uniform Resource Identifier (URI)

- URI is used for naming and addressing

- Contemporary view of URI partitioning
  - An individual scheme does not need to be cast into one of a discrete set of URI types such as “URL”, “URN”, etc.
  - “http:”, “urn:”, are a kind of URI schemes, which define subspaces called namespaces

- Relationships to URLs
  - URL does not refer to a formal partition of URI space
  - URL is a type of URI
  - URL identifies a resource via a representation of its primary access mechanism
  - “http:” is a URI scheme and an http URI is a URL

XML

- A W3C standard to complement HTML http://www.w3.org/TR/REC-xml (2/98)
- Origins: structured text SGML
  - HTML describes presentation
  - XML describes content
- Possessing the characteristics of semi-structured data
  - Missing or additional attributes
  - Multiple valued attributes
  - Different types in different objects
  - Heterogeneous collections

Self-describing, irregular data, no a priori structure

XML

- Is a standardized *textual* notation for semi-structured data
- Is primarily aimed at semi-structured data which is a tree
- An arc in the tree with label L pointing at a node N in the semi-structured data is represented in the XML document as a pair of tags:

  <L> … </L>

  … is the XML description of the part of the semi-structured data for which N is the root
- Leaf nodes are represented by the data they contain

XML Terminology

- Tags: book, title, author, …
- Start tag: <book>, end tag: </book>
- Elements: <book>…</book>,<author>…</author>
- Elements are nested: elements and subelements
- Empty element: <red></red> abbrv. <red/>
- An XML document: single *root element*
- *Processing instructions* <? … ?>
- *Comments* <!-- … -->

XML Terminology

- Well-formed XML document:
  - Follows the syntax rules setup for XML by W3C
    - It consists of a correctly build Prolog, followed by exactly one Document Element
  - Each element can contain child elements or data. There is a tree with exactly one root element
  - Each element is correctly build, i.e. it has an opening `(<elementName>)` and a closing tag `(</elementName>)`
  - Tags of different elements do not overlap
  - Opening tags of an element may have multiple attributes with a unique name within this tag. The attributes have the following form: `<elementName attributeName="…">`

XML Terminology

- Valid XML document
  - Association with a Document Type Definition (DTD)
  - Comply with that DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="first.css"?>

<!DOCTYPE DOCUMENT [
  <!ELEMENT DOCUMENT (GREETING, MESSAGE)>
  <!ELEMENT GREETING (#PCDATA)>
  <!ELEMENT MESSAGE (#PCDATA)>
]>
<DOCUMENT>
  <GREETING> Hello from XML </GREETING>
  <MESSAGE> Welcome to Web Dynamics </MESSAGE>

</DOCUMENT>
```

XML Namespaces (1)

- http://www.w3.org/TR/REC-xml-names (1/99)

- Goal:
  - An XML document should contain elements and attributes that are defined for and used by multiple software modules (e.g. query processors)
  - Modularity: re-use well-understood markup vocabulary for which useful software exists

- Problem:

  Unique element names are not possible
  - Software modules need to recognize the tags and attributes for which they are designed to process

- Idea:

  Universal names extending the scope beyond the containing document

XML Namespaces (2)

- XML namespace is collection of names, identified by a URI reference, which are used as element names and attribute names
- Defined by special attribute

  ```
  xmlns : name ::= [prefix:]localpart
  ```

  is a globally unique name

  ```
  <book xmlns:isbn=“www.isbn-org.org/def”>
     <title> … </title>
     <number> 15 </number>
     <isbn:number> …. </isbn:number>
  </book>
  ```

- The namespace prefix isbn is defined for the element book and all its child elements

XML Namespaces (3)

- Default namespaces

```
<book xmlns='urn:loc.gov:books,
    xmlns:isbn='urn:ISBN:0-395-36341-6'>
      <title>Cheaper by the Dozen</title>
      <isbn:number>1568491379</isbn:number>
</book>
```

- Elements with no prefix are bound to the default namespace
- Default namespaces do not apply to attribute names

XML Namespaces (4)

- Scope of namespaces declaration applies
  - To the defining element
  - All sub-elements

- Multiple namespace prefixes can be declared in a single element

```
<bk:book xmlns:bk='urn:loc.gov:books'
          xmlns:isbn='urn:ISBN:0-395-36341-6'>
     <bk:title>Cheaper by the Dozen</bk:title>
     <isbn:number>1568491379</isbn:number>
</bk:book>
```

XML Namespaces (5)

- Uniqueness of attributes

  In an XML document no tag may contain two attributes which:

  1. Have identical names, or
  2. Have qualified names with the same local parts and with prefixes which have been bound to namespace names that are identical

```
<x xmlns:n1="http://www.w3.org
   xmlns:n2="http://www.w3.org" >
    <bad a="1"     a="2" />
    <bad n1:a="1"  n2:a="2" />
</x>
```

Illegal

```
<x xmlns:n1="http://www.w3.org"
   xmlns="http://www.w3.org" >
     <good a="1"     b="2" />
     <good a="1"     n1:a="2" />
</x>
```

legal

RDF

- http://www.w3.org/TR/REC-rdf-syntax (2/99)
- Goal: Interoperability on the Web
- Problem: Volume of information
  - Not possible to manage it manually
- Solution: Use metadata to describe data contained on the Web
- Metadata: “data about data”
  - Example: Library catalog – describes publications
- Example application areas:
  - *Resource discovery* to provide better search engine capabilities
  - *Cataloging* for describing the content and content relationships at a particular web site
  - Describing *intellectual property rights* of web pages

RDF

- RDF is a model for representing named properties and property values
  - “Attributes of resources”
- RDF supports common conventions of:
  - Semantics
  - Syntax
  - Structure
- RDF uses XML as a common syntax for the exchange and processing of metadata
- Semantics: edge-labeled graphs

Basic Entities of the RDF Data Model

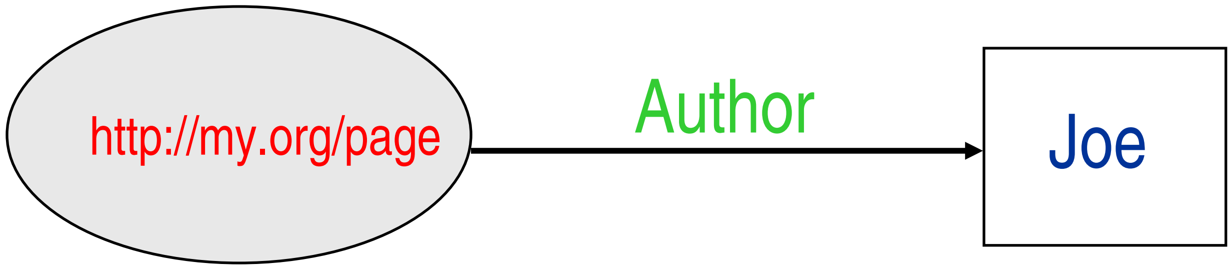
- Resources
  - All things being described by RDF expressions
  - Named by URIs (= resource identifier)
- Properties
  - Specific aspect, characteristic, attribute, or relation used to describe a resource
- Statement
  - Resource + property + property value (literal, another resource)

Subject (Resource)	http://my.org/page
Predicate (Property)	Title
Object (Literal)	„My first document“

Subject (Resource)	http://my.org/page
Predicate (Property)	Author
Object (Resource)	http://my.org/~joe

RDF Description

Statement: "The *author* of *http://my.org/page* is *Joe*."



- Subject (Resource) - *http://my.org/page*
- Predicate (Property) - *author*
- Object (Literal) - *Joe*

RDF Document

Complete XML document

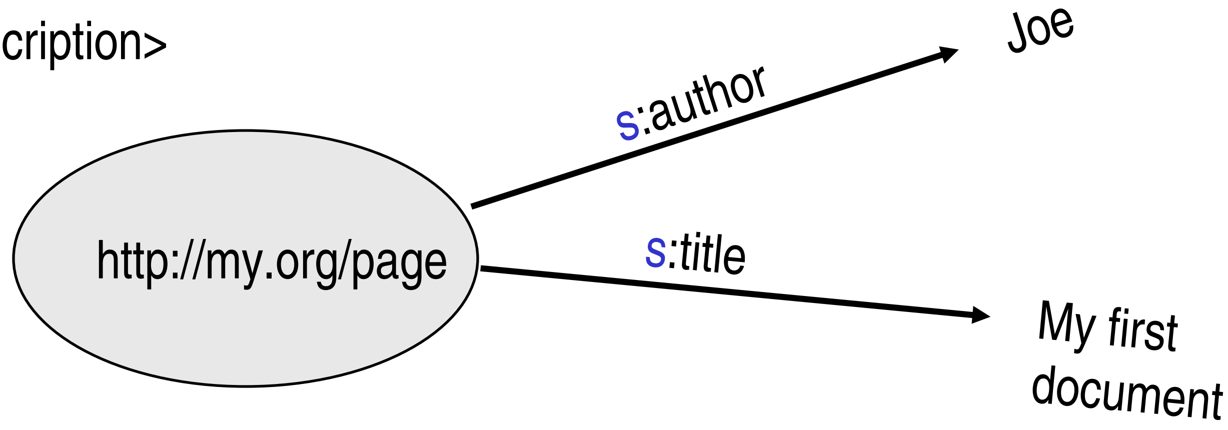
```
<?xml version="1.0"?>
    <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#“
                    xmlns:s="http://my.org/schema/">
                          <rdf:Description about=" http://my.org/page ">
                              <s:author>Joe</s:author>
                        </rdf:Description>
    </rdf:RDF>
```

Property names must be associated with a schema

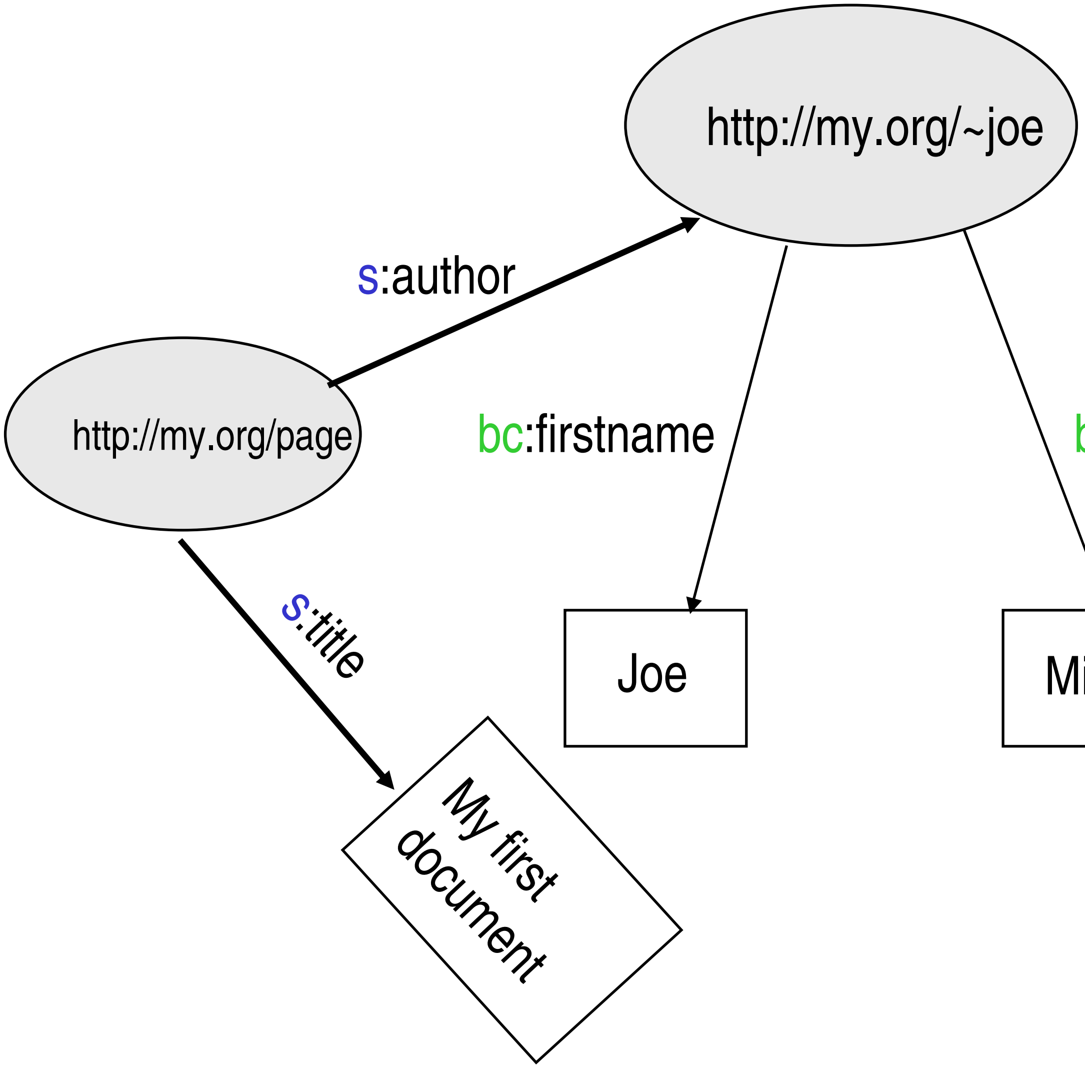
RDF Document Enhanced

Using the default namespace (rdf: prefix removed)

```
<?xml version="1.0"?>
     <RDF  xmlns = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
               xmlns:s="http://my.org/schema/">
                    <Description about="http://my.org/page">
                        <s:author>Joe</s:author>
                        <s:title> My first document </s:title>
                    </Description>
    </RDF>
```



Referencing other Resources



Referencing other Resources

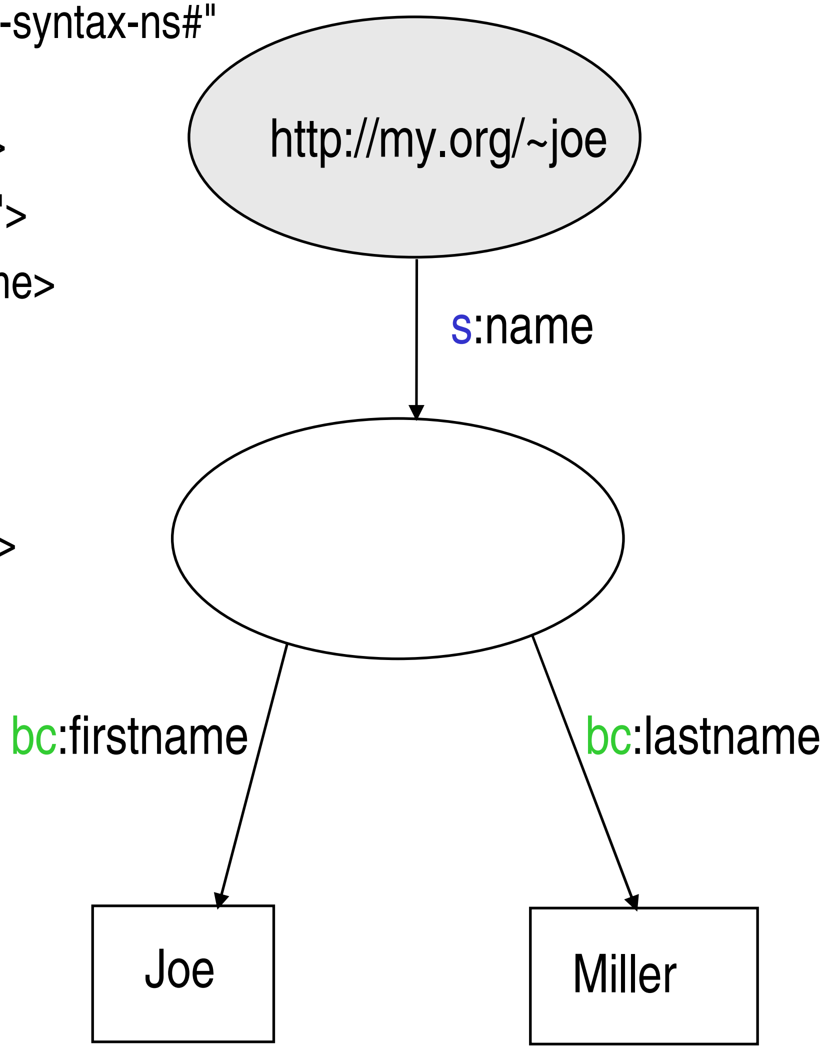
```
<?xml version="1.0"?>
     <RDF xmlns = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
               xmlns:s= "http://my.org/schema/"
               xmlns:bc = "http://my.org/businessCard/" >

          <Description about="http://my.org/page">
               <s:author resource="http://my.org/~joe/“/>
               <s:title> My first document </s:title>
          </Description>

           <Description about="http://my.org/~joe">
               <bc:firstname> Joe </bc:firstname>
               <bc:lastname> Miller </bc:lastname>
          </Description>

     </RDF>
```

Internal Identifier (“Blank Nodes”)

```
<?xml version="1.0"?>
     <RDF  xmlns = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
               xmlns:s= "http://my.org/schema/"
               xmlns:bc = "http://my.org/businessCard/" >
                   <Description about="http://my.org/~joe">
                       <s:name>rdf:nodeID="A234"</s:name>
                   </Description>
                   <Description rdf:nodeID="A234">
                       <bc:firstname> Joe </bc:firstname>
                       <bc:lastname> Miller </bc:lastname>
                   </Description>
    </RDF>
```

<http://my.org/~joe> s:name _:A234.

_:A234 bc:firstname “Joe”.



RDF Container

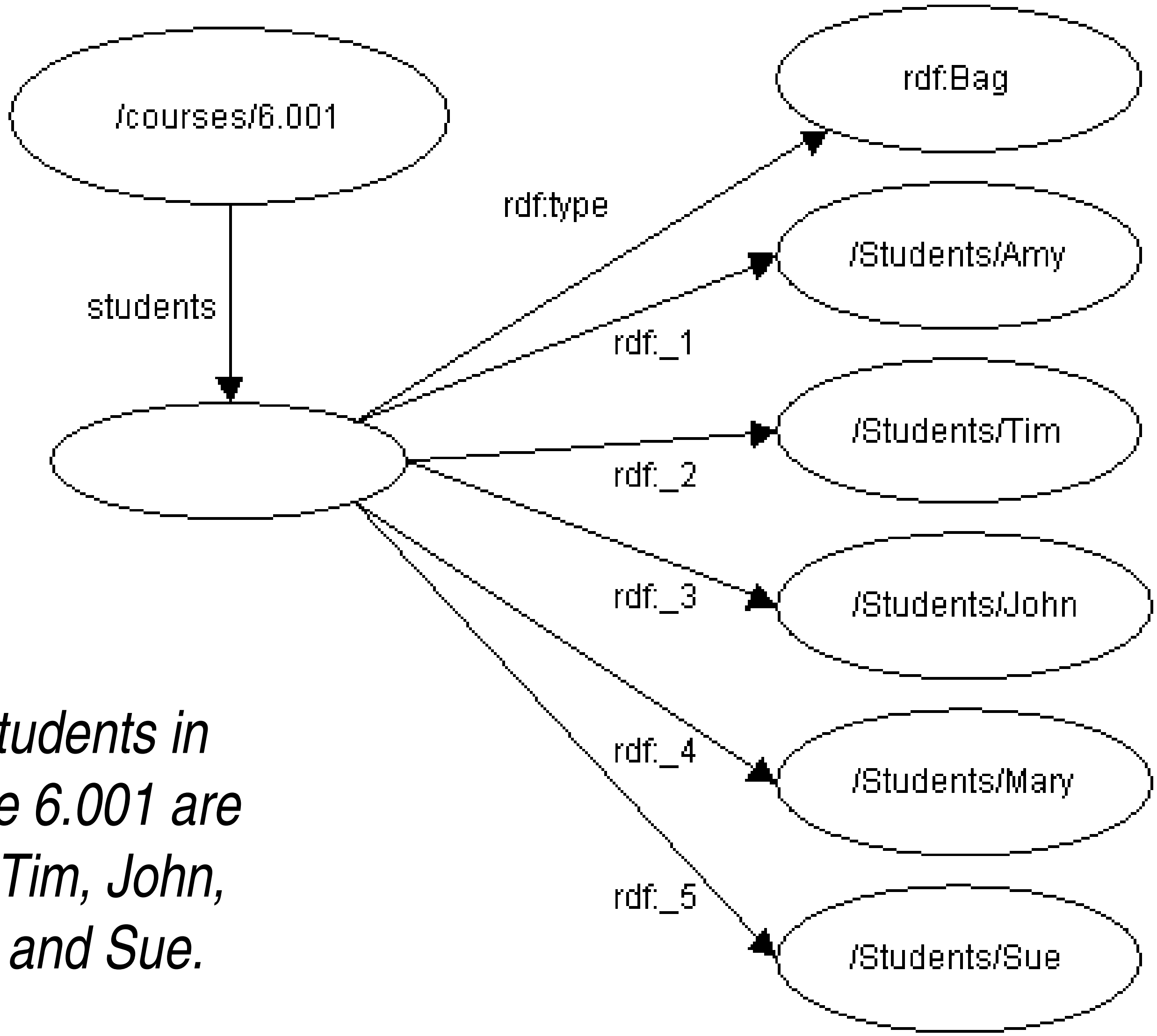
- Goal: reference to a collection of resources or literals

- Bag
  - Unordered list of resources or literals

- Sequence
  - Ordered list of resources or literals

- Alternative
  - List of resources or literals that represent alternatives for the (single) value of a property
    - Title in different languages
    - Different download sites

RDF Bag Example



The students in course 6.001 are Amy, Tim, John, Mary, and Sue.

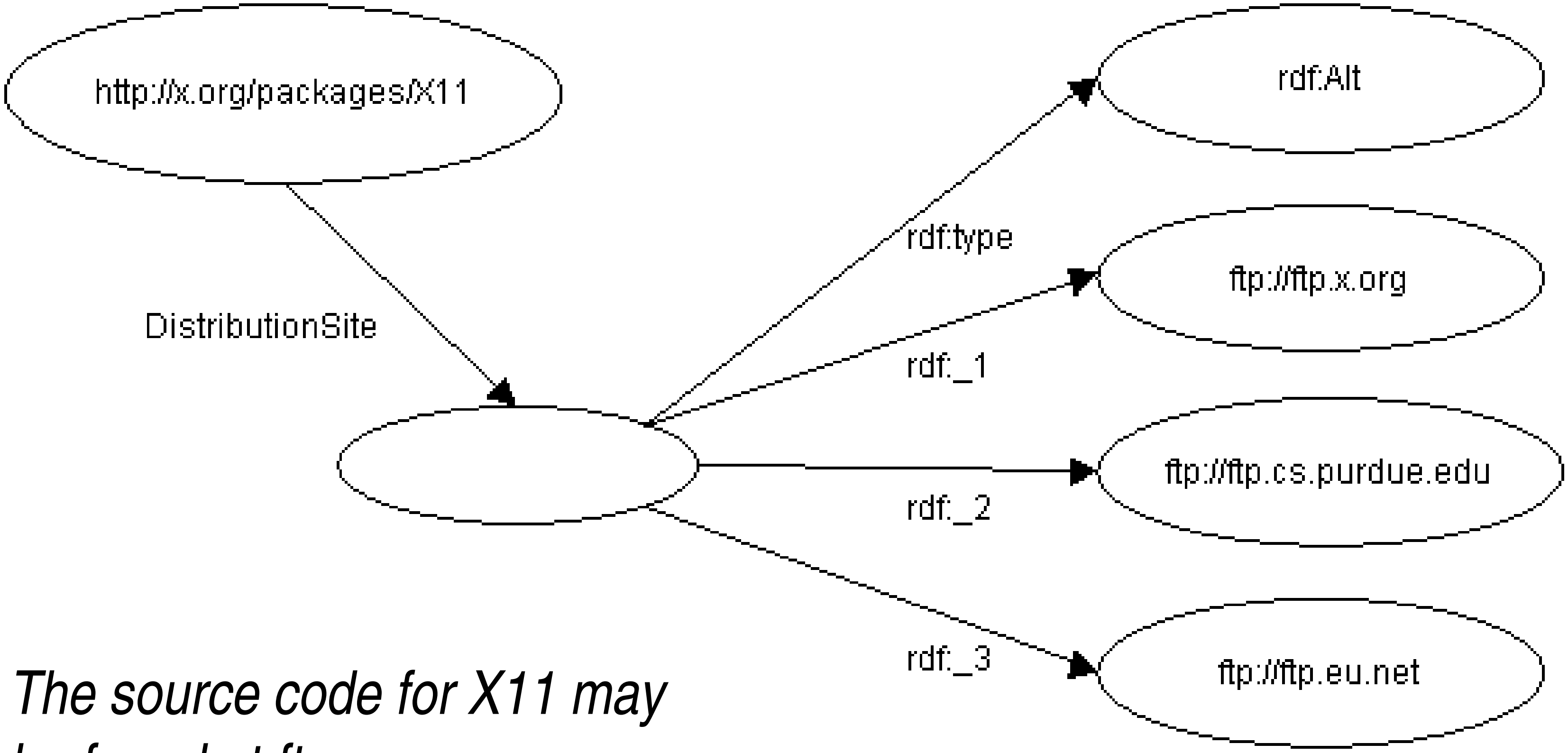
http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/

RDF Bag Example

```
<rdf:RDF>
    <rdf:Description about="http://mycollege.edu/courses/6.001">
          <s:students>
             <rdf:Bag>
                    <rdf:li resource="http://mycollege.edu/students/Amy"/>
                    <rdf:li resource="http://mycollege.edu/students/Tim"/>
                    <rdf:li resource="http://mycollege.edu/students/John"/>
                    <rdf:li resource="http://mycollege.edu/students/Mary"/>
                    <rdf:li resource="http://mycollege.edu/students/Sue"/>
             </rdf:Bag>
          </s:students>
    </rdf:Description>
</rdf:RDF>
```

http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/

RDF Alternative Example



The source code for X11 may be found at ftp.x.org, ftp.cs.purdue.edu, or ftp.eu.net.

http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/

RDF Alternative Example

```
<rdf:RDF>
   <rdf:Description about="http://x.org/packages/X11">
          <s:DistributionSite>
                <rdf:Alt>
                        <rdf:li resource="ftp://ftp.x.org"/>
                        <rdf:li resource="ftp://ftp.cs.purdue.edu"/>
                        <rdf:li resource="ftp://ftp.eu.net"/>
                </rdf:Alt>
        </s:DistributionSite>
   </rdf:Description>
</rdf:RDF>
```

http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/

Statements about the Members of a Container

```
<rdf:Bag ID="pages">
    <rdf:li resource="http://my.org/intro.html" />
    <rdf:li resource="http://my.org/main.html" />
</rdf:Bag>

<rdf:Description about="#pages">
    <s:creator>Joe Miller</s:creator>
</rdf:Description>

<rdf:Description aboutEach="#pages">
    <s:creator>Joe Miller</s:creator>
</rdf:Description>
```

?

Distributive referent

RDF Distributive Referent

- Using a distributive referent on a container is the same as making all the statements about each of the members separately

- No explicit graph representation of distributive referents is defined

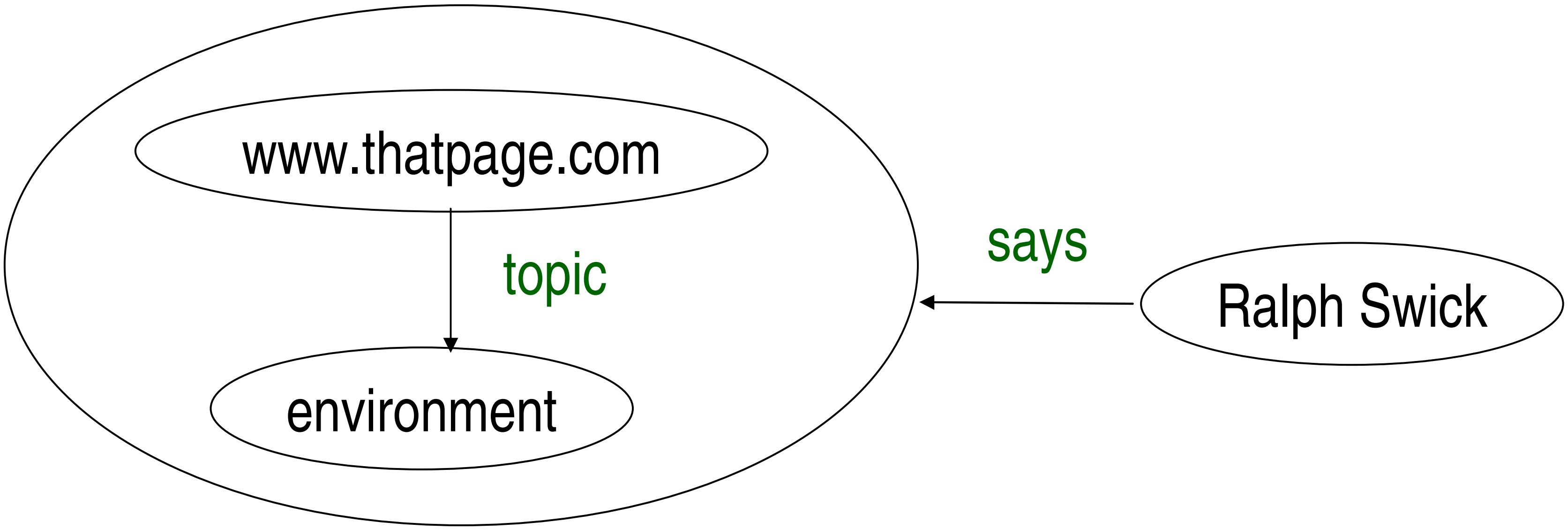
- The distributive referent statement is equivalent to:

```
<rdf:Description about=" http://my.org/intro.html">
    <s:creator> Joe Miller </s:creator>
</rdf:Description>

<rdf:Description about=" http://my.org/main.html">
    <s:creator> Joe Miller </s:creator>
</rdf:Description>
```

RDF Higher Order Statements

“Ralph Swick says:

‘the topic of www.thatpage.com is environment’ “



RDF uses *reification*

RDF Reification

- Modeling the original statement as a resource we need four properties

- Subject
  - The *subject* property identifies the resource being described by the modeled statement

- Predicate
  - The *predicate* property identifies the original property in the modeled statement

- Object
  - The *object* property identifies the property value in the modeled statement

- Type
  - The value of the *type* property describes the type of the new resource

An Example of RDF Reification

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:a="http://description.org/schema/">
    <rdf:Description>
      <rdf:subject resource="www.thatpage.com"/>
      <rdf:predicate resource="http://my.org/schema/topic"/>
      <rdf:object>environment</rdf:object>
      <rdf:type resource="http://www.w3.org/1999/02/22-rdf-syntax-
                  ns#Statement"/>
      <s:attributedTo>Ralph Swick</s:attributedTo>
    </rdf:Description>
</rdf:RDF>
```

Vocabularies

- Data integration needs agreements on
  - Terms
    - “Translator”, “author”
  - Categories used
    - “Person”, “literature”
  - Relationships among those
    - “An author is also a Person…”, “historical fiction is a narrower term than fiction”
    - I.e., new relationships can be deduced

- There is a need for “languages” to define such vocabularies
  - To define those vocabularies
  - To assign clear “semantics” on how new relationships can be deduced

“Core” Vocabularies

- There are a number of “core” vocabularies
  - Dublin Core: about information resources, digital libraries, with extensions for rights, permissions, digital right management
  - FOAF: about people and their organizations
  - DOAP: on the descriptions of software projects
  - SIOC: Semantically-Interlinked Online Communities
  - vCard in RDF
  - …

⇒ Ontologies/vocabularies must be shared and reused!

RDF Schema (RDFS)

- http://www.w3.org/TR/rdf-schema/

- Interoperability requires that writer and reader of a statement understand the same meaning for the terms used
  - E.g. Author, Title, Firstname, Lastname etc.

- Meaning in RDF is expressed through reference to a schema
  - Kind of dictionary:
    - Defines the terms
    - Gives specific meaning to them
  - Definitions and restrictions of usage for properties are defined

- Namespaces are used to tie a specific use of a word to the dictionary (schema) where the intended definition is to be found

RDF Schema (RDFS)

- RDF Schema = RDF Vocabulary Description Language
  - RDFS defines a vocabulary for RDF
  - RDFS describes how to use RDF
- RDFS specifies a data model on which RDF statements are based on
  - Abstract data types (Classes)
  - Hierarchical class models and inheritance
  - Syntax for data interexchange
- RDFS vocabulary
  - RDFS classes: <rdfs:Class>, <rdfs:Resource> etc.
  - RDFS properties: <rdfs:subClassOf>, <rdfs:subPropertyOf> etc.

Limitations of RDFS

- Characterization of properties
- Identification of objects with different URI-s
- Disjointness or equivalence of classes
- Construct classes, not only name them
- More complex classification schemes
- Can a program reason about some terms? E.g.:
  - "if «Person» resources «A» and «B» have the same «foaf:email» property, then «A» and «B» are identical"
- etc.

⇒ Ontologies

What is an Ontology?

- Ontology is a specification of a conceptualization that is designed for reuse across multiple applications and implementations [Karp00]

- An ontology is an explicit, formal specification of a shared conceptualization [Grub93]

- Modeling for knowledge representation
  - Relations exist between classes
  - Relations are represented in rules

Web Ontology Language (OWL)

- http://www.w3.org/TR/owl-features/

- Why OWL?
  - An ontology language is required to describe the meaning of terminology used in Web documents formally
  - OWL adds more vocabulary for describing properties and classes
  - OWL facilitates greater machine interpretability of Web content

- OWL sublanguages
  - OWL Lite provides a classification hierarchy and simple constraints
  - OWL DL enables maximum expressiveness (DL: description logics)
  - OWL Full supports maximum expressiveness and the syntactic freedom of RDF
  - OWL Lite $\subseteq$ OWL DL $\subseteq$ OWL Full

OWL Lite

- OWL Lite features are related to RDF schema
  - A *Class* defines a group of individuals that belong together
  - Individuals that share the same properties
  - Furthermore: rdfs:subClassOf, rdfs:subPropertyOf, rdfs:domain, rdfs:range, rdf:Property

- Equality and Inequality
  - equivalentClass: two classes are equivalent
  - equivalentProperty: two properties are equivalent
  - sameAs/differentFrom: two individuals are the same/different
  - AllDifferent: a number of individuals are mutually distinct

OWL Lite

- Property characteristics
  - inverseOf, TransitiveProperty, SymmetricProperty, FunctionalProperty, InverseFunctional Property
- Property restrictions
  - allValuesFrom/someValuesFrom: is stated on a property with respect to a class
- Restricted cardinality
  - Constrains the cardinality of a property on instances of a class
- Class intersection
  - Allows intersections of names classes and restrictions
- Furthermore: OWL datatypes, header information, annotation properties, and versioning

RIF (Rule Interchange Format)

- The goals of the RIF work:
  - Define simple rule language(s) for the (Semantic) Web
  - Define interchange formats for rule based systems
- RIF defines several “dialects” of languages
- RIF is not bound to RDF only
  - E.g., relationships may involve more than 2 entities
  - There are dialects for production rule systems
- RIF Core simplest RIF “dialect”
  - Core document contains
    - Directives like import, prefix settings for URI-s, etc
    - A sequence of logical implications
  - Expressivity of RIF Core
    - Definite Horn without function symbols → “Datalog” [e.g. p(a,b,c), but not p(f(a),b,c)]
    - Built-in datatypes and predicates
    - “Local” symbols, a bit like blank nodes

Applying RIF on RDF Data

- Typical scenario:
  - The “data” of the application is available in RDF
  - Rules on that data is described using RIF
  - The two sets are “bound” (eg, RIF “imports” the data)
  - A RIF processor produces new relationships
- Making RIF working on RDF
  - RDF triples have to be represent able in RIF
  - Various constructions (typing, datatypes, lists) should be aligned
  - The semantics of the two worlds should be compatible
- There is a separate document that brings these together

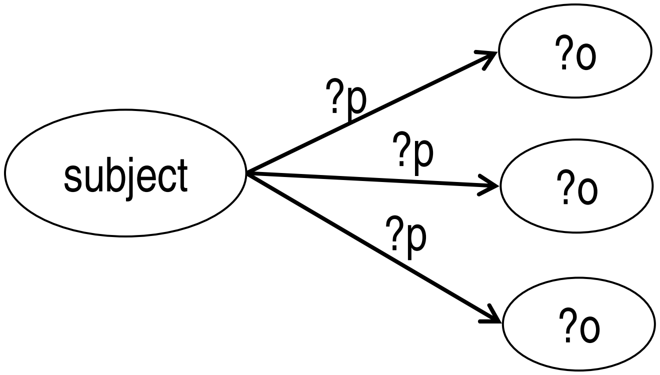
RIF vs. OWL

- The expressivity of the two is fairly identical
  - The emphasis are a bit different
- Using rules vs. ontologies may largely depend on
  - Available tools
  - Personal technical experience and expertise
  - Taste…

SPARQL

- SPARQL protocol and RDF Query Language
- http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/
- Based on SQL

```
Select ?p, ?o
Where subject ?p ?o
```



- The query results corresponds to the ways in which the query’s graph pattern matches the data
- Multiple matches

```
Prefix dc: <http://purl.org/dc/elements/1.1/>
Select ?title
Where { <http://example.org/book/book1> dc: title ?title . }
```

Semantic Web

Dr. Marc Spaniol

SPARQL

- Graph patterns in SPARQL
  - Basic graph pattern can be grouped by { }
  - The keyword “optional” can be applied to a graph pattern

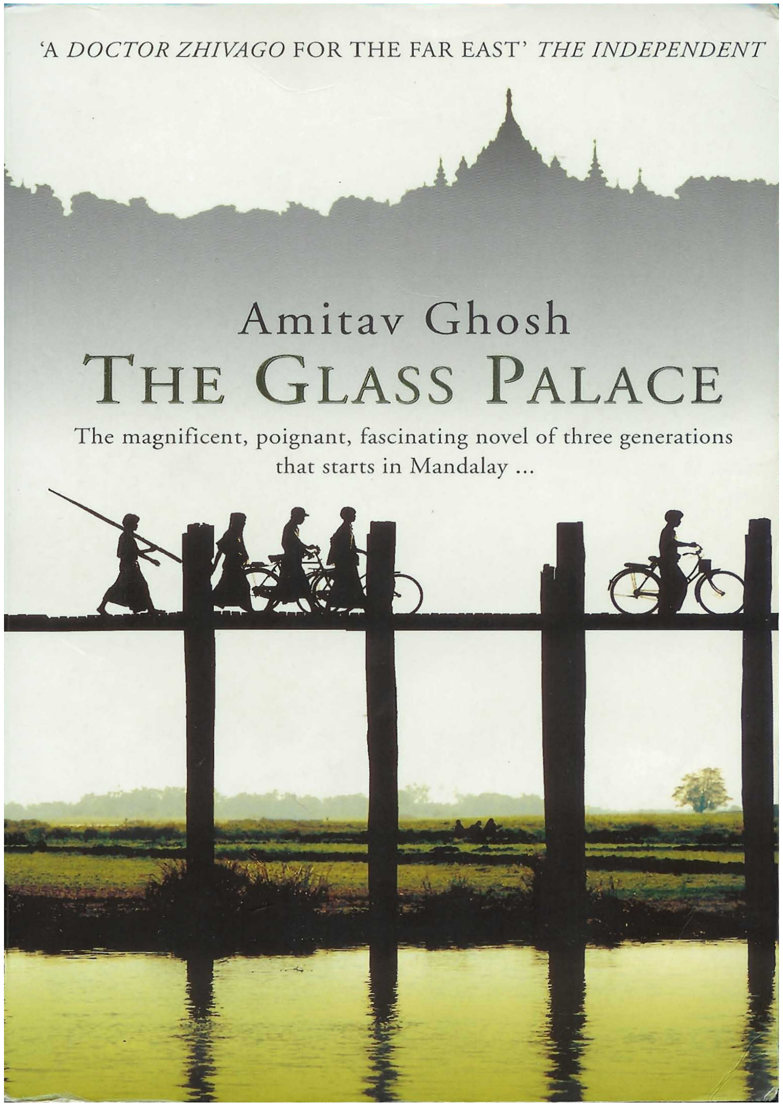
- Filter for group graph pattern

```
Prefix  dc: <http://purl.org/dc/elements/1.1/>
Select ?book where
{ ?book dc:publishedBy <http://springer.com> .
?book dc:Price ?price filter (?price <35) }
```

Semantic Web

Dr. Marc Spaniol

The Semantic Web by Example


'A DOCTOR ZHIVAGO FOR THE FAR EAST' THE INDEPENDENT
Amitav Ghosh
THE GLASS PALACE
The magnificent, poignant, fascinating novel of three generations that starts in Mandalay ...

Databases and
Information Systems
Prof. Dr. G. Weikum

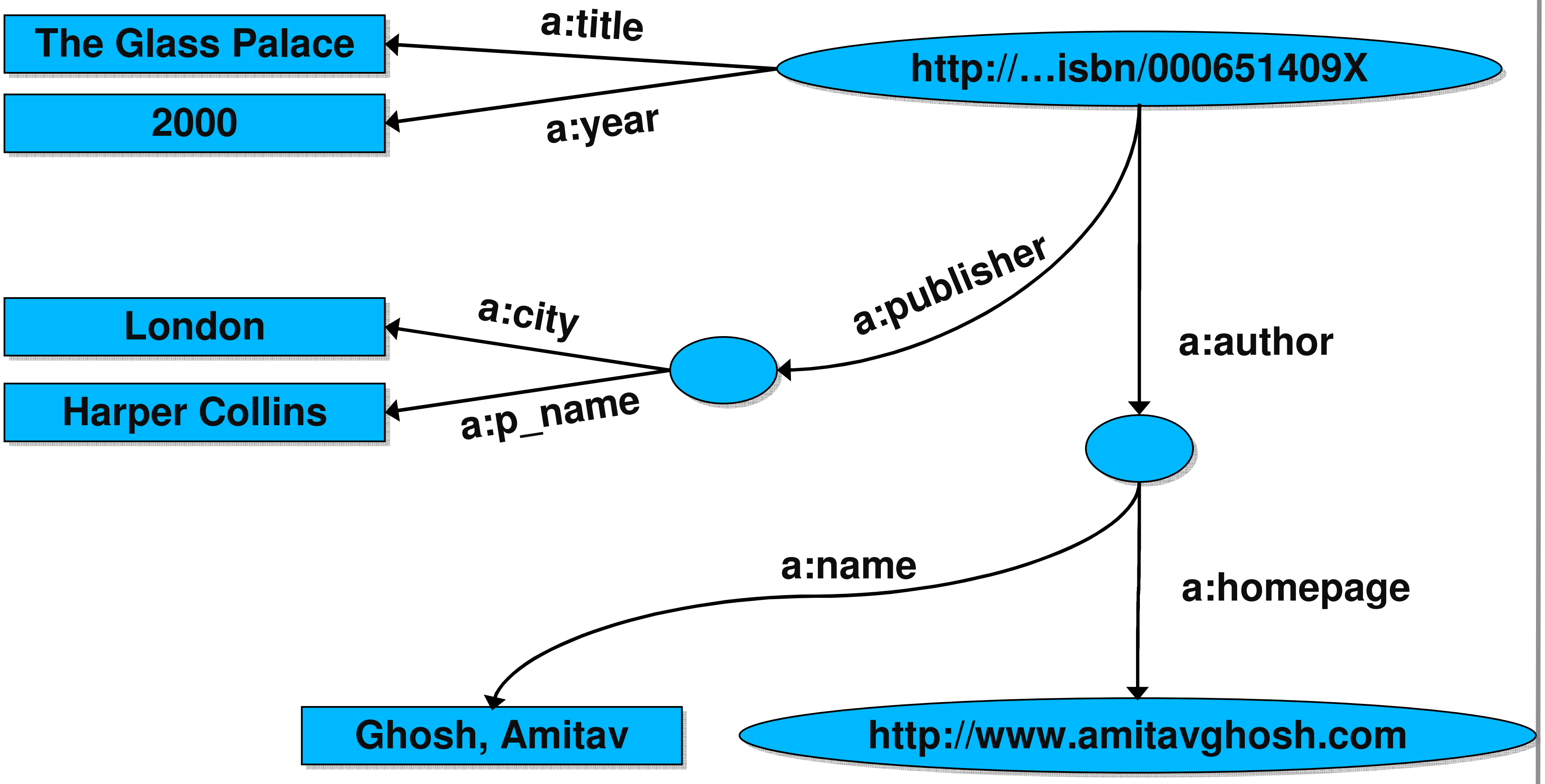
Simplified Bookstore Data (dataset “A”)

ID	Author	Title	Publisher	Year
ISBN 0-00-6511409-X	id_xyz	The Glass Palace	id_qpr	2000

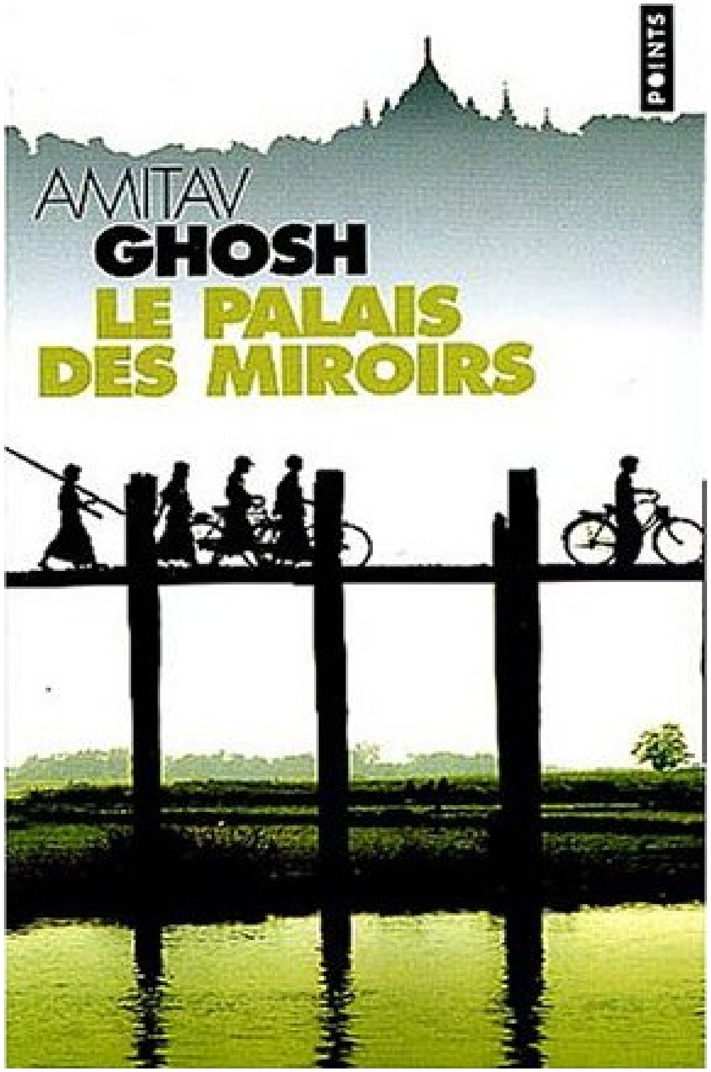
ID	Name	Homepage
id_xyz	Ghosh, Amitav	http://www.amitavghosh.com

ID	Publisher’s name	City
id_qpr	Harper Collins	London

Creating RDF Data from Store "A" as a Set of Relations



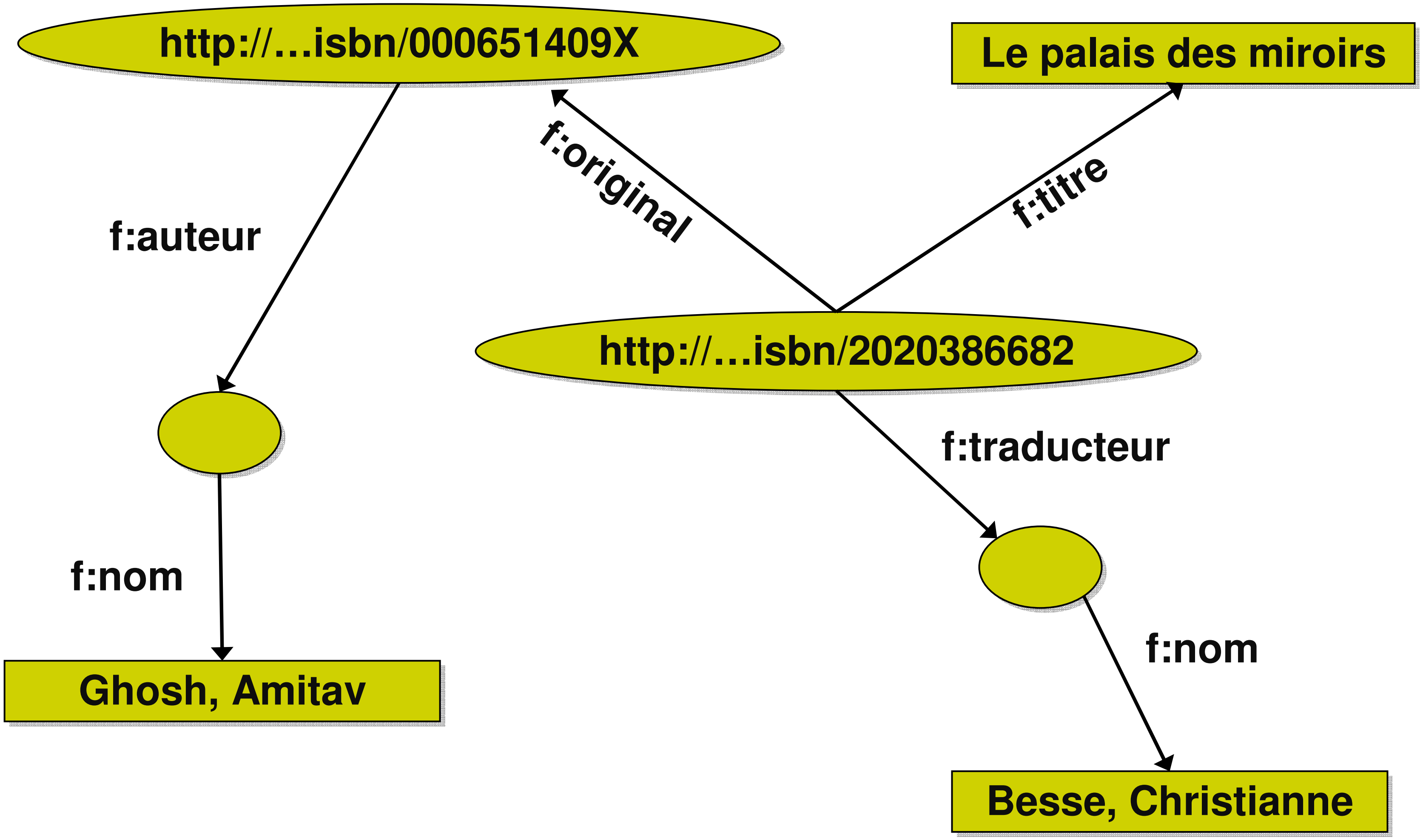
Same book in French...



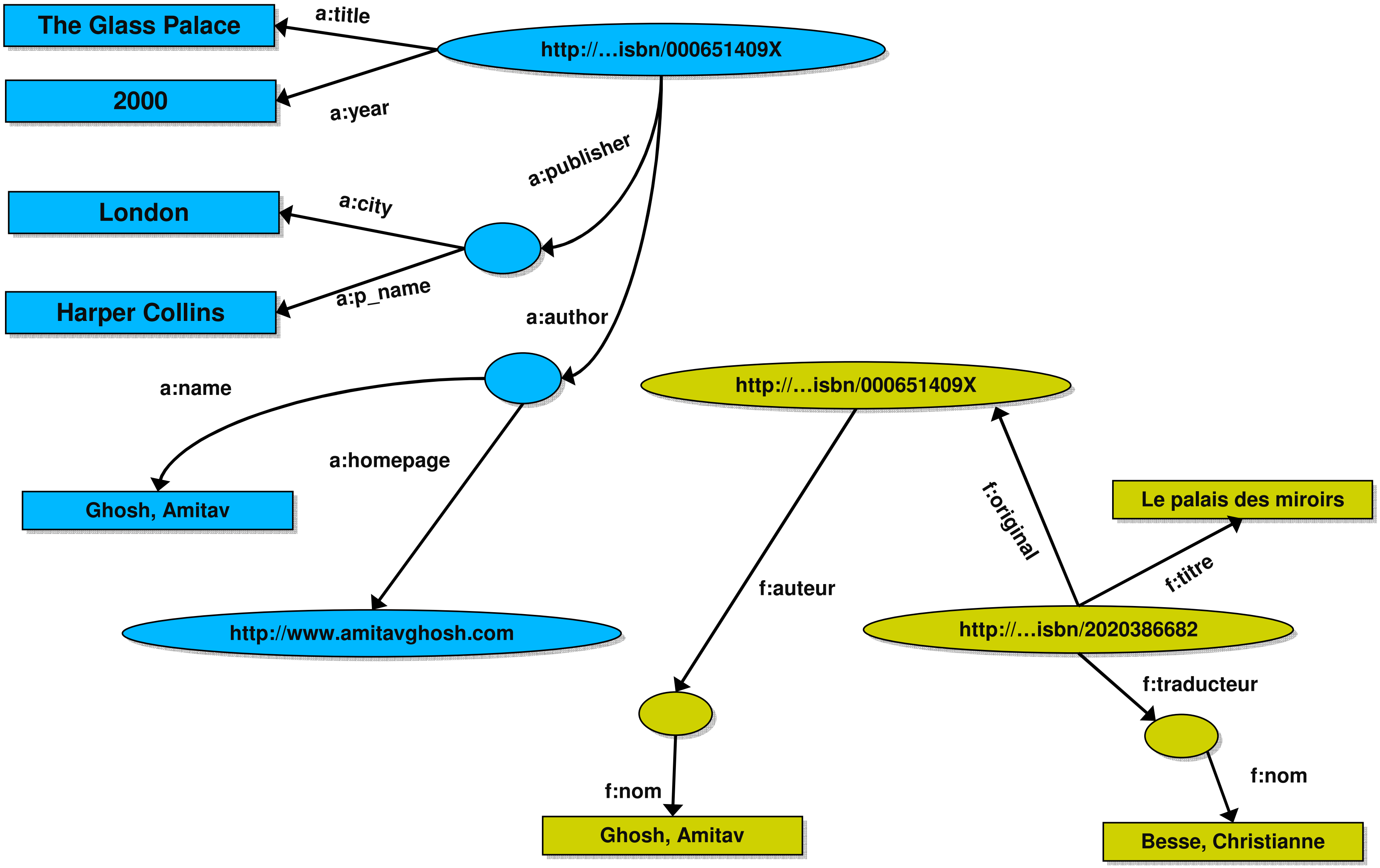
Another Bookstore Data (dataset "F")

	A	B	C	D
1	**ID**	**Titre**	**Traducteur**	**Original**
2	ISBN 2020286682	Le Palais des Miroirs	$A12$	ISBN 0-00-6511409-X
3				
4				
5				
6	**ID**	**Auteur**		
7	ISBN 0-00-6511409-X	$A11$		
8				
9				
10	**Nom**			
11	Ghosh, Amitav			
12	Besse, Christianne			

Creating RDF Data from Store “F” as a Set of Relations



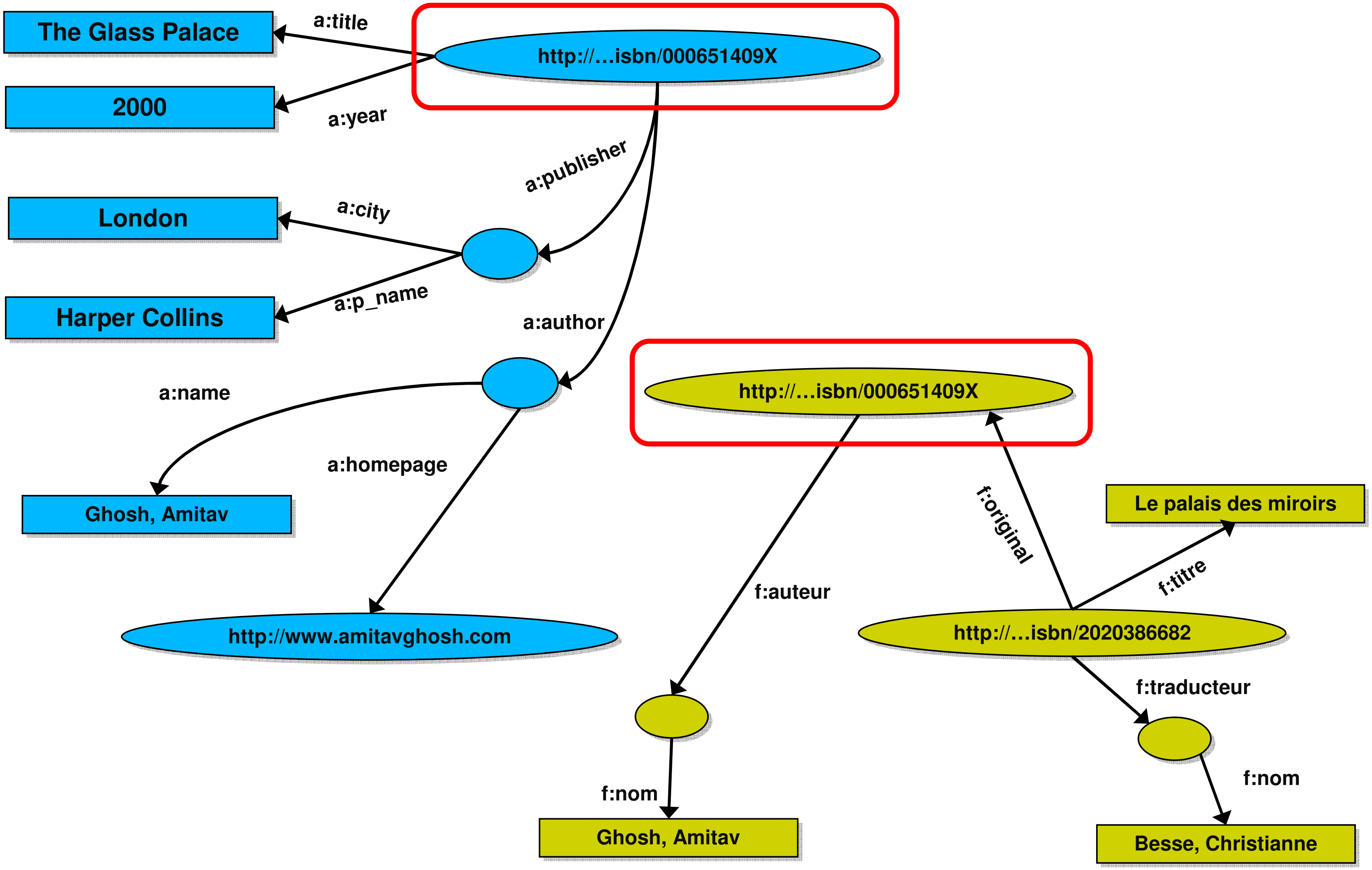
Merging Data from Store “A” and “F”



Semantic Web

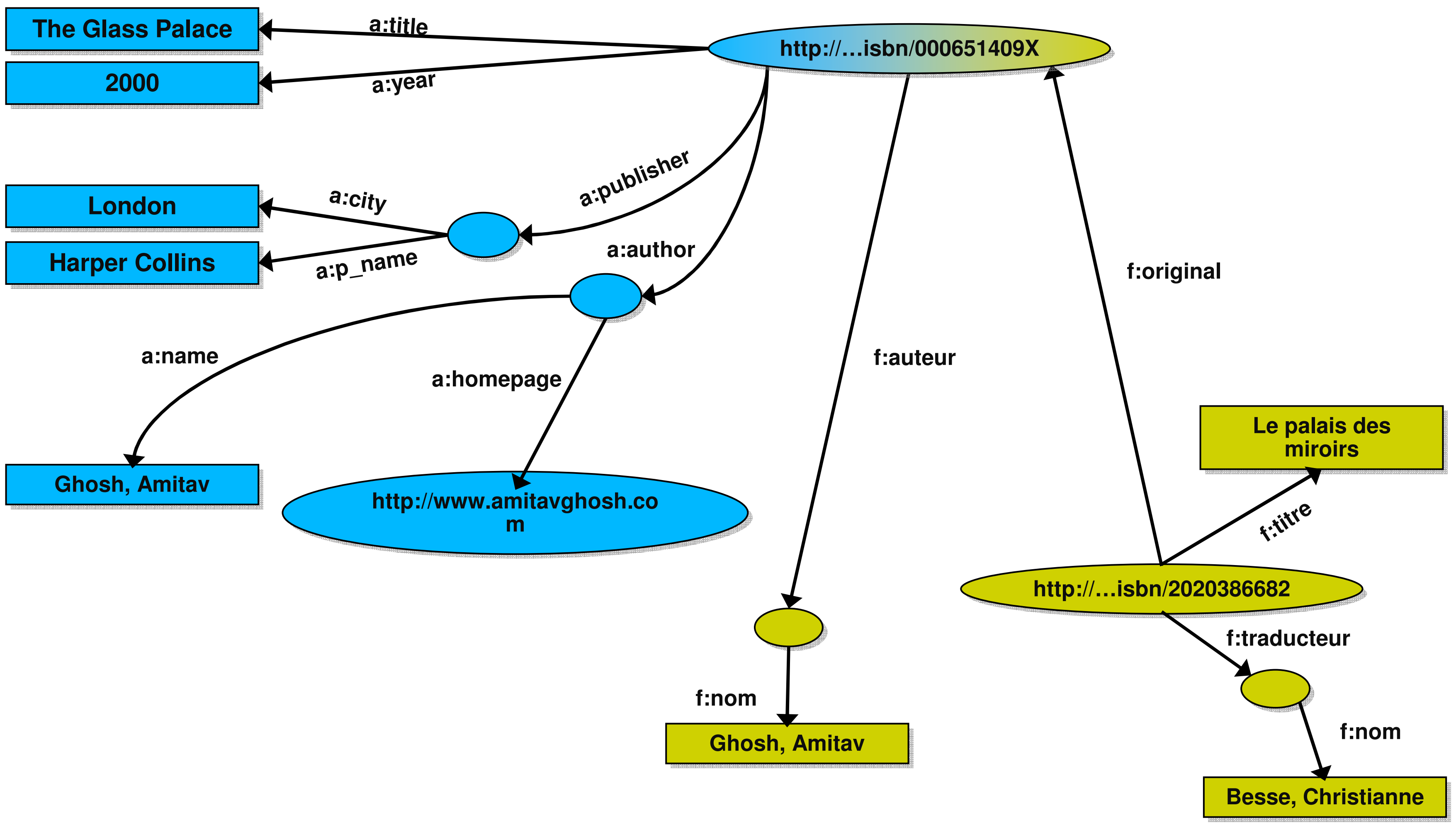
Dr. Marc Spaniol

Merging Data (continued)



Databases and Information Systems
Prof. Dr. G. Weikum

Merged Data

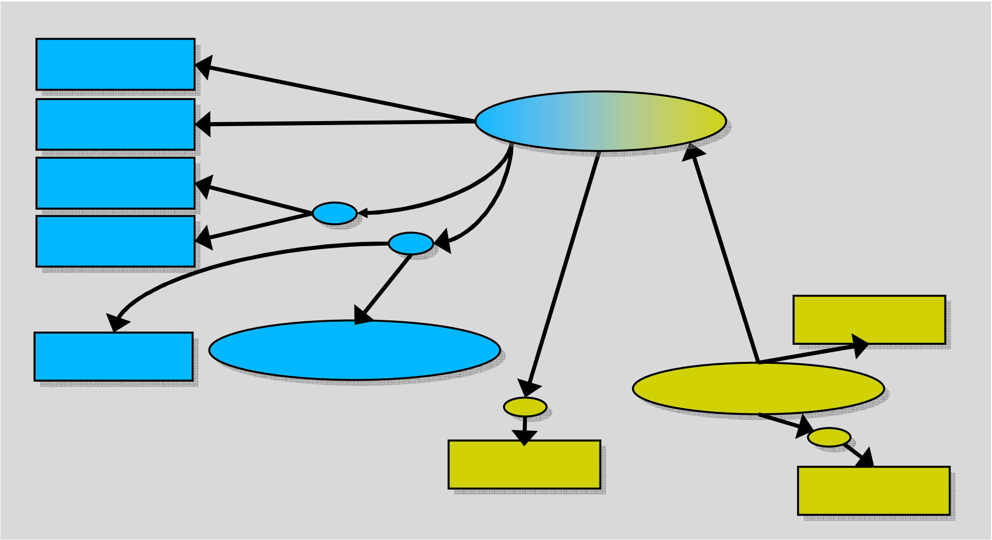

The Glass Palace
a:title
2000
a:year
http://…isbn/000651409X
a:publisher
London
a:city
Harper Collins
a:p_name
a:author
a:name
a:homepage
Ghosh, Amitav
http://www.amitavghosh.co
m
f:auteur
f:original
Le palais des
miroirs
f:titre
http://…isbn/2020386682
f:traducteur
f:nom
Ghosh, Amitav
f:nom
Besse, Christianne

Querying the Data

- User of data “F” can now ask queries like:
  - “give me the title of the original”
    well, … « donnes-moi le titre de l’original »

- This information is not in the dataset “F”…

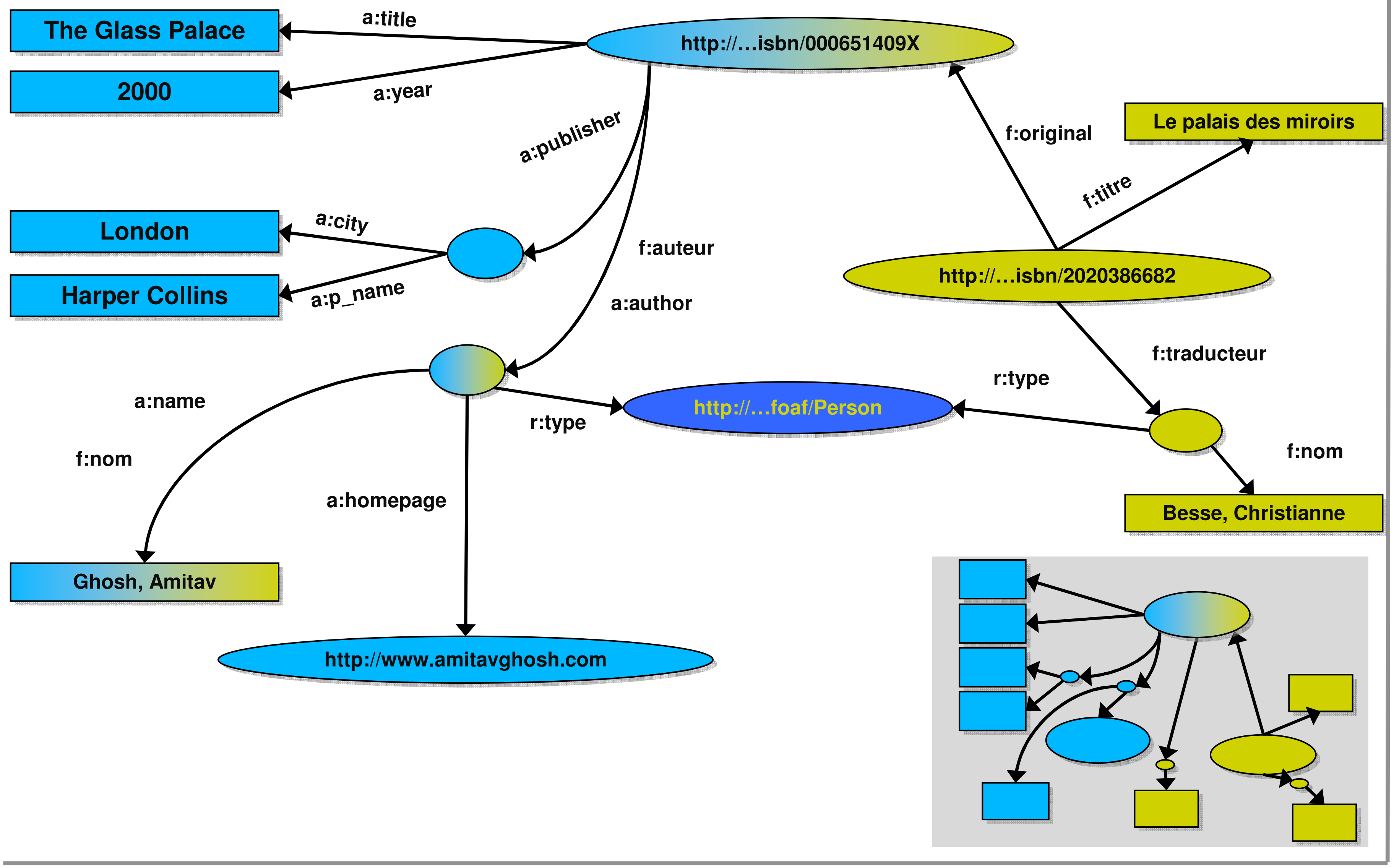
- …but can be retrieved by merging with dataset “A”!



Achieving more

- We “feel” that a:author and f:auteur should be the same
- But an automatic merge doest not know that!
- Let us add some extra information to the merged data:
    - a:author same as f:auteur
    - Both identify a “Person”
    - A term that a community may have already defined:
        - A “Person” is uniquely identified by his/her name and, say, homepage
        - It can be used as a “category” for certain type of resources

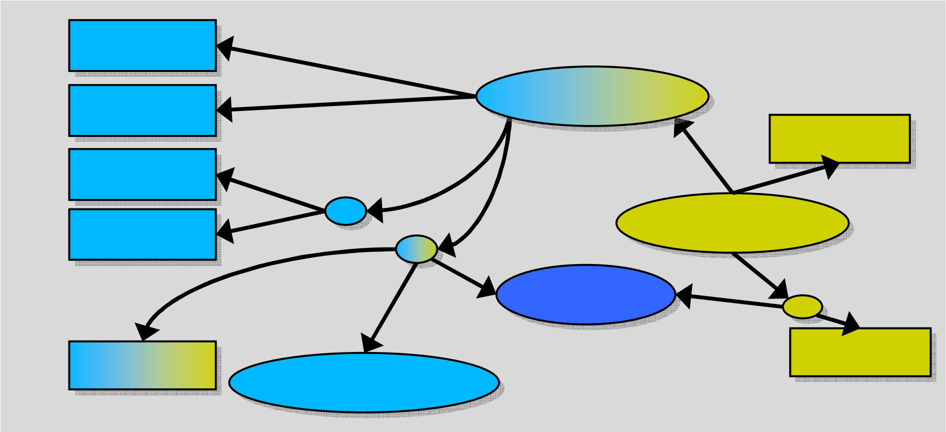
Incorporating Extra Knowledge



Richer Queries

- User of dataset “F” can now query:
  - “donnes-moi la page d’accueil de l’auteur de l’original” well… “give me the home page of the original’s ‘auteur’”
- The information is not in datasets “F” or “A”…

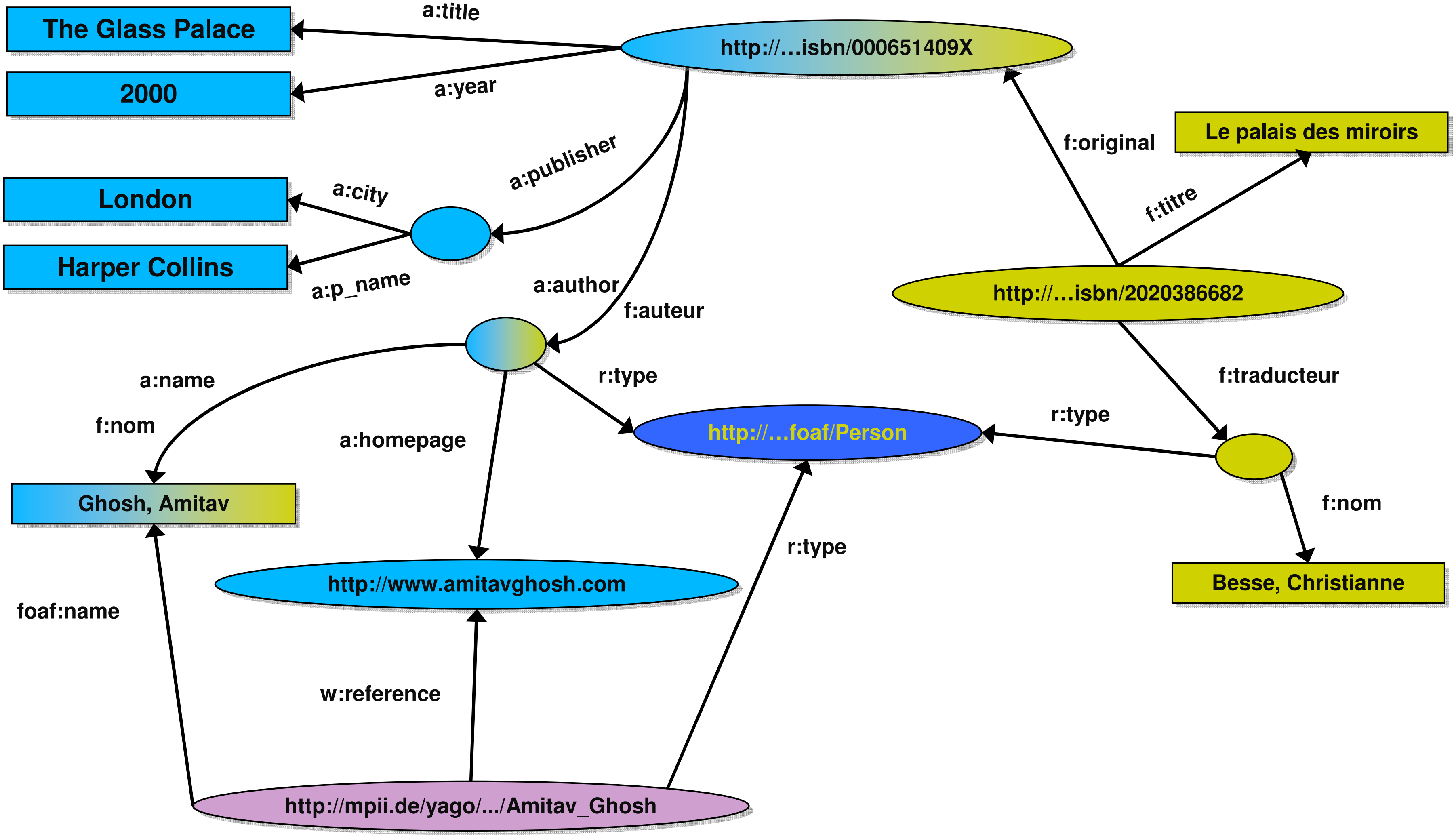
  …but was made available by:
  - merging datasets “A” and datasets “F”
  - adding three simple extra statements as an extra “glue”



Semantic Web

Dr. Marc Spaniol

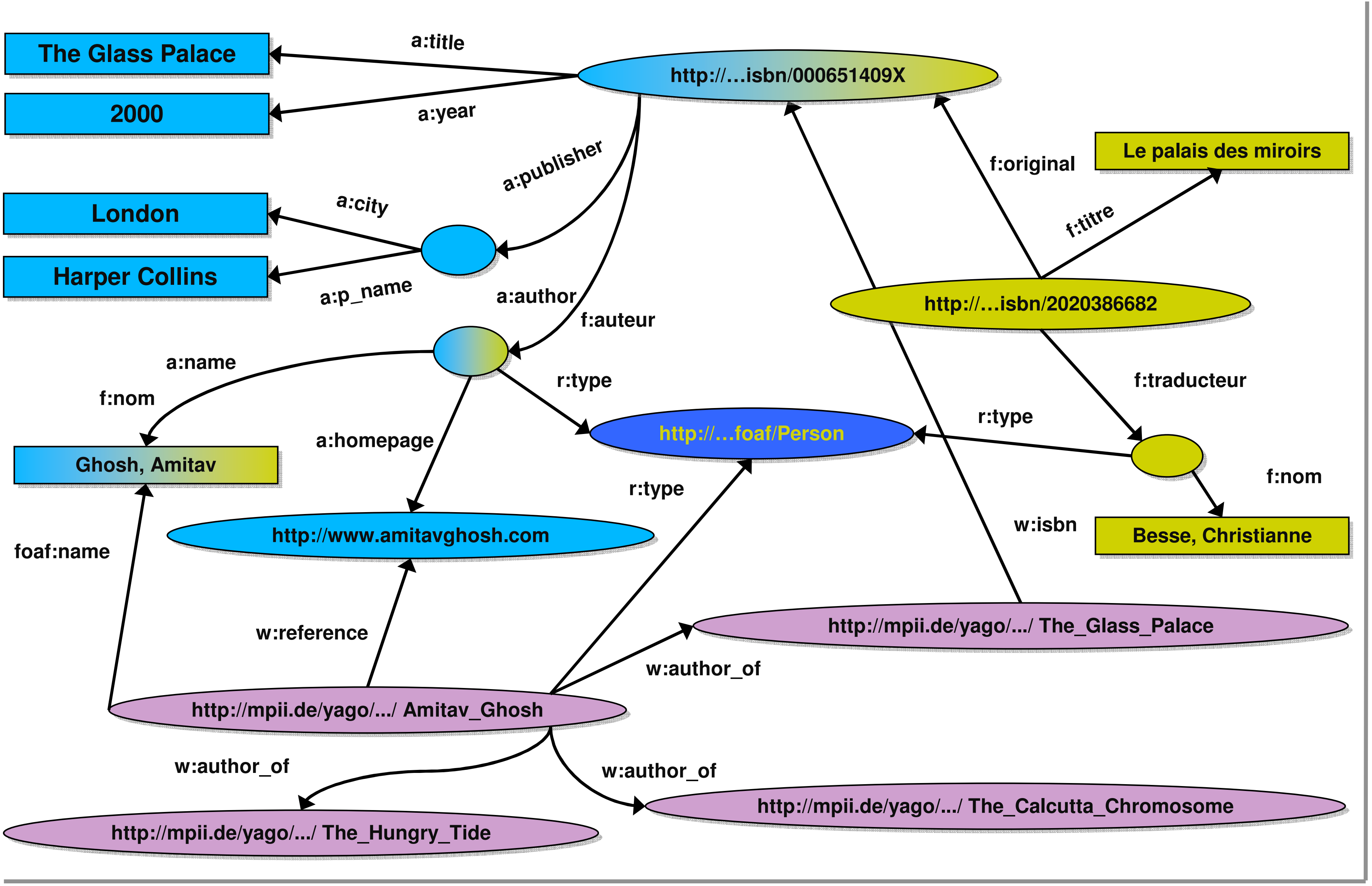
Merging with Wikipedia Data



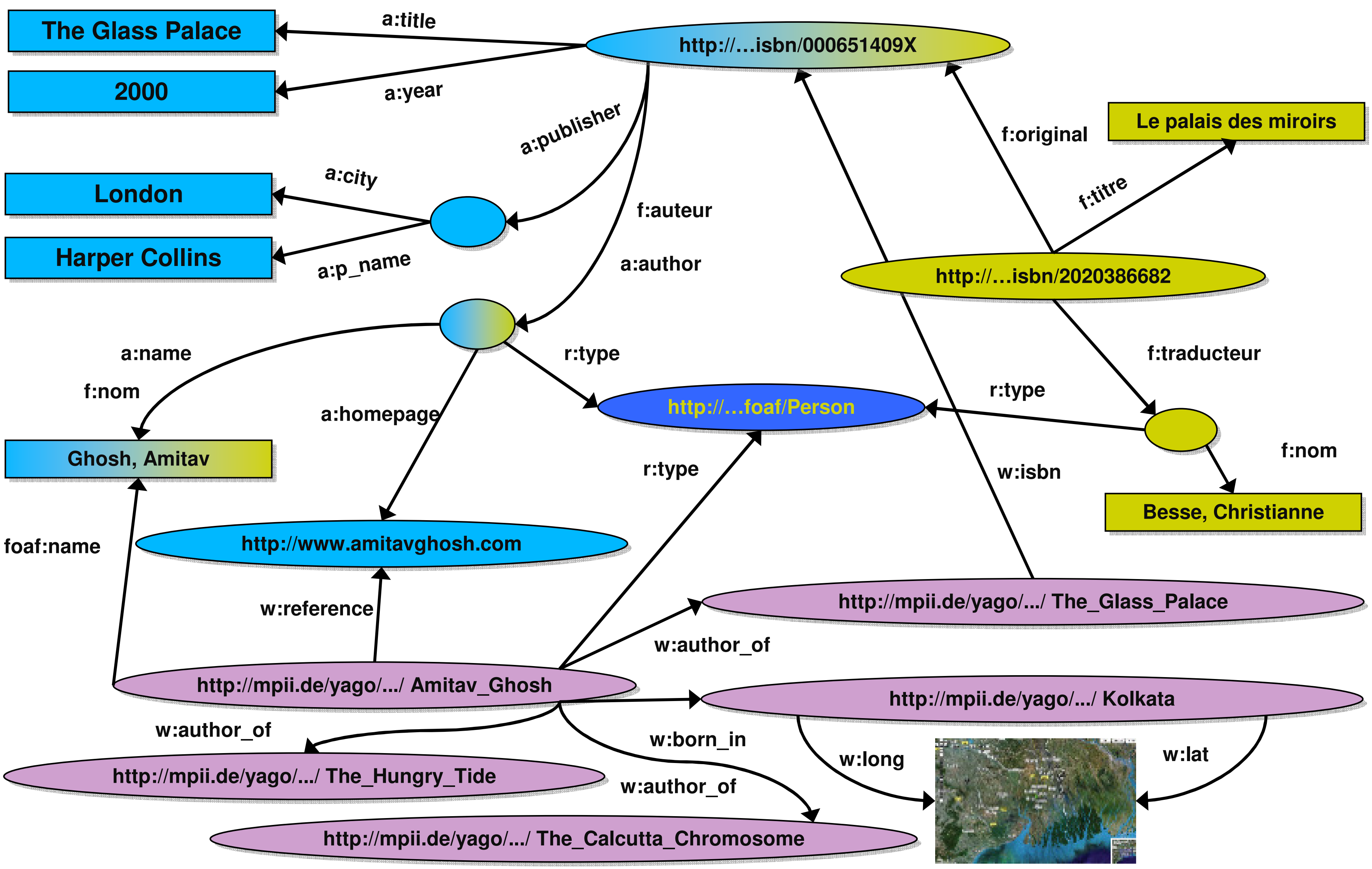
Semantic Web

Dr. Marc Spaniol

Merging with Wikipedia Data (cont)



Wikipedia “enriched” Data



The Glass Palace

2000

London

Harper Collins

Ghosh, Amitav

http://…isbn/000651409X

http://…isbn/2020386682

Le palais des miroirs

http://…foaf/Person

Besse, Christianne

http://www.amitavghosh.com

http://mpii.de/yago/…/ The_Glass_Palace

http://mpii.de/yago/…/ Amitav_Ghosh

http://mpii.de/yago/…/ Kolkata

http://mpii.de/yago/…/ The_Hungry_Tide

http://mpii.de/yago/…/ The_Calcutta_Chromosome

a:title

a:year

a:publisher

a:city

a:p_name

f:auteur

a:author

f:original

f:titre

a:name

f:nom

r:type

a:homepage

f:traducteur

r:type

r:type

f:nom

w:isbn

foaf:name

w:reference

w:author_of

w:author_of

w:born_in

w:author_of

w:long

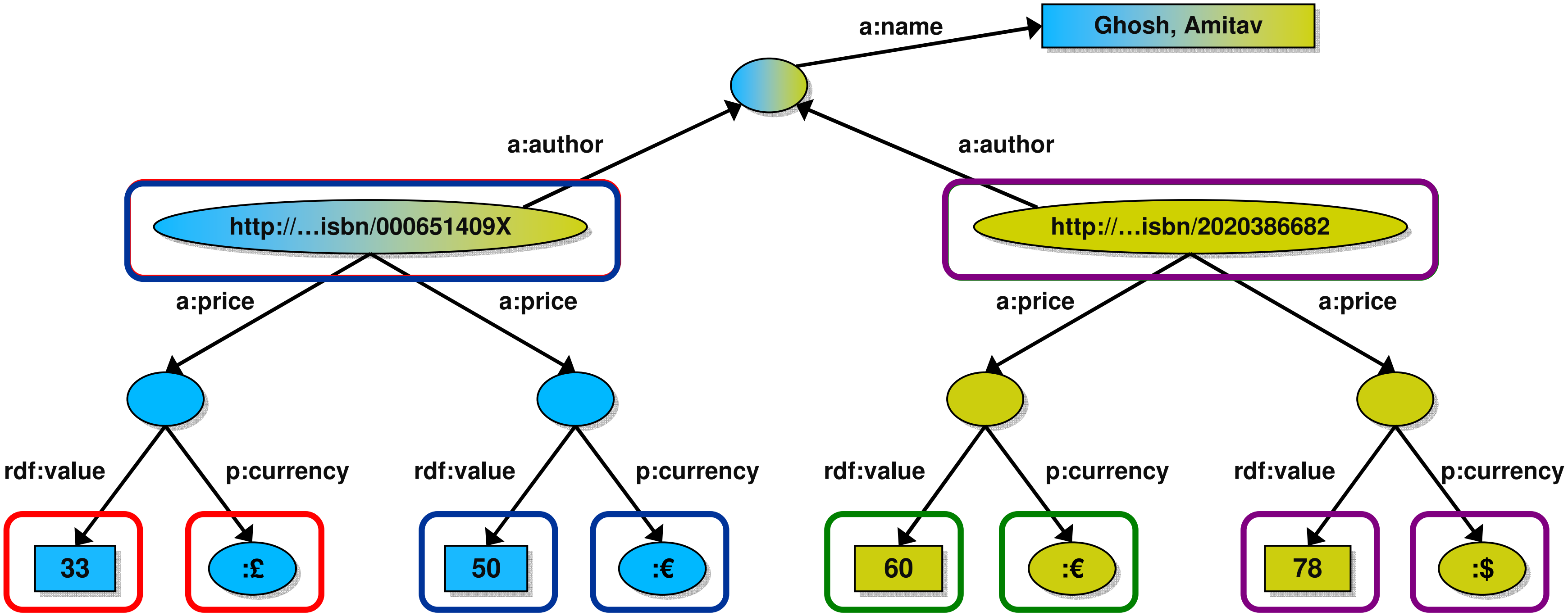
w:lat

Querying the Data with SPARQL

```
SELECT ?isbn ?price ?currency
WHERE {?isbn a:price ?x. ?x rdf:value ?price.
?x p:currency ?currency.}
```

Returns:

[<…409X>,33,:£], [<…409X>,50,:€], [<…6682>,60,:€], [<…6682>,78,:$]

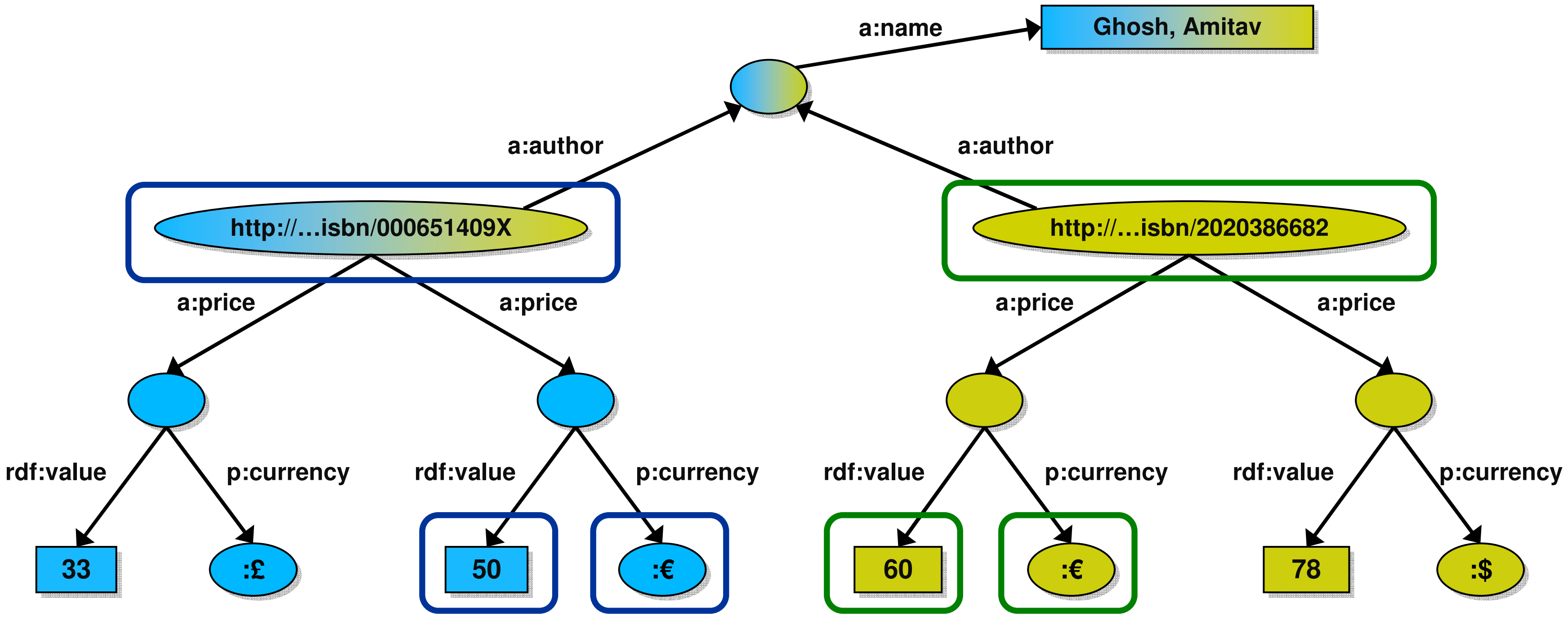


Pattern Constraints

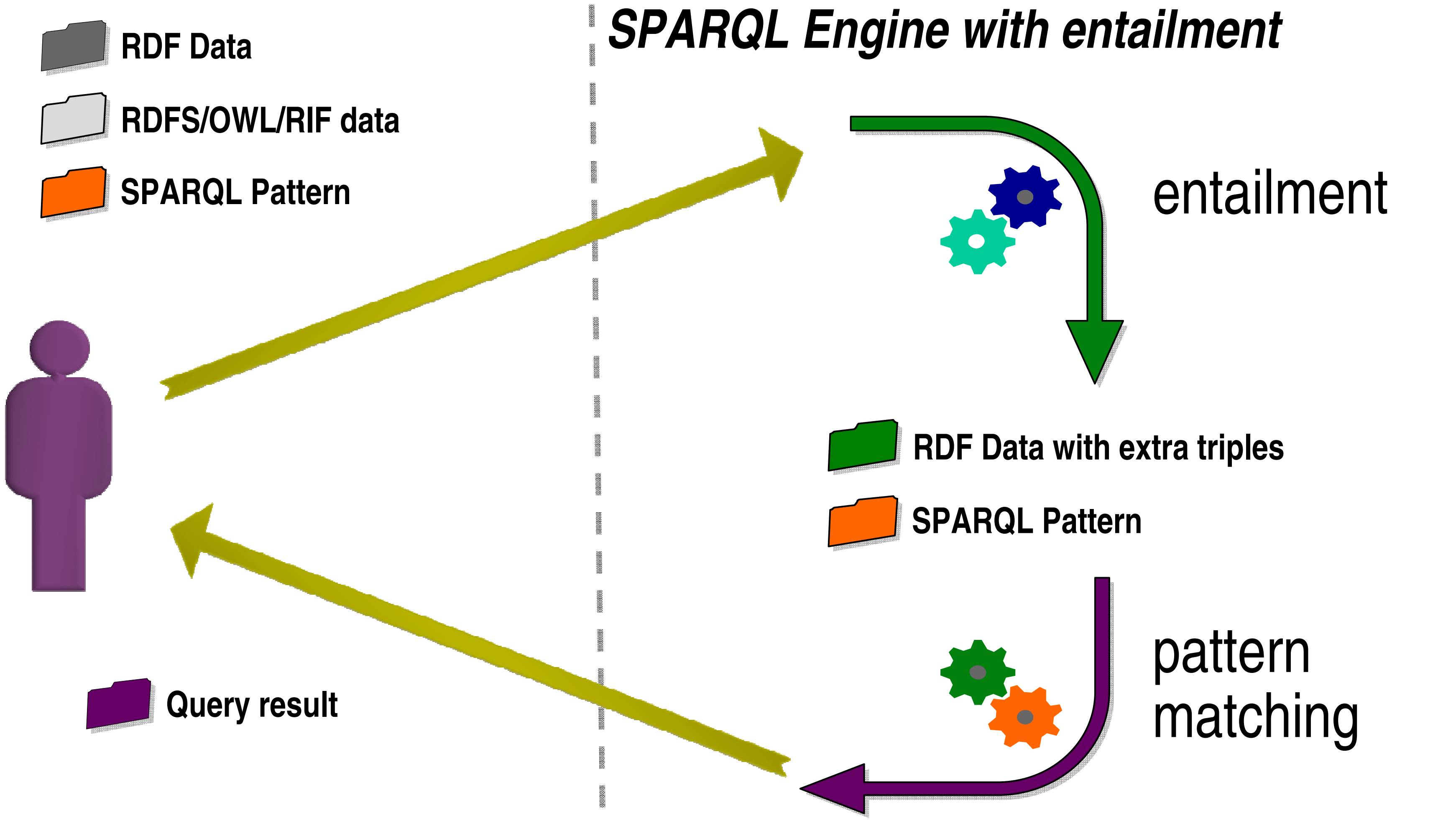
```
SELECT ?isbn ?price ?currency
WHERE { ?isbn a:price ?x. ?x rdf:value ?price.
?x p:currency ?currency. FILTER(?currency == :€) }
```

Returns:

[<…409X>,50,:€], [<…6682>,60,:€]



SPARQL 1.1 and RDFS/OWL/RIF



RDF Data

RDFS/OWL/RIF data

SPARQL Pattern

SPARQL Engine with entailment

entailment

RDF Data with extra triples

SPARQL Pattern

pattern matching

Query result

Summary

- The Semantic Web is a
  - Great vision
  - Place where machines will be able to participate and help
  - “Means” of automatically doing what’s done every day by Web users

- The Semantic Web needs
  - Plenty of machine-readable (RDF) data
  - Rules, taxonomies and/or ontologies to interpret the data

- The Semantic Web can add extra knowledge to merged datasets
  - E.g., a full classification of various types of library data
  - Geographical information

⇒ The Semantic Web allows “complex” queries to be asked/answered

References

[AlHe08] D. Allemang and J. Hendler: “Semantic Web for the Working Ontologist”, 2008.

[AnHa08] G. Antoniu and F. van Harmelen: “Semantic Web Primer”, 2nd edition, 2008.

[Herm10] Ivan Herman: “Introduction to Semantic Web Technologies (tutorial)”, Semantic Technology Conference, San Francisco, CA, USA, 2010.
http://www.w3.org/2010/Talks/0622-SemTech-IH/
[last access: July 14, 2010]

[HSKr09] P. Hitzler, R. Sebastian, M. Krötzsch: “Foundation of Semantic Web Technologies”, 2009.

[W3C10] W3C Semantic Web Activity: “Home page”, 2010.
http://www.w3.org/2001/sw/
[last access: July 14, 2010]