



An Introduction to Semantic Web (Tutorial)

17th International World Wide Web Conference
Beijing, China, 21st April, 2008

Ivan Herman (郝易文), W3C

> Towards a Semantic Web



- The current Web represents information using
 - natural language (English, Hungarian, Chinese,...)
 - graphics, multimedia, page layout
- Humans can process this easily
 - can deduce facts from partial information
 - can create mental associations
 - are used to various sensory information
 - (well, sort of... people with disabilities may have serious problems on the Web with rich media!)

> Towards a Semantic Web



- Tasks often require to combine data on the Web:
 - hotel and travel information may come from different sites
 - searches in different digital libraries
 - etc.
- Again, humans combine these information easily
 - even if different terminologies are used!

> However...



- However: machines are ignorant!
 - partial information is unusable
 - difficult to make sense from, e.g., an image
 - drawing analogies automatically is difficult
 - difficult to combine information automatically
 - is `<foo:creator>` same as `<bar:author>`?
 - ...

> Example: automatic airline reservation



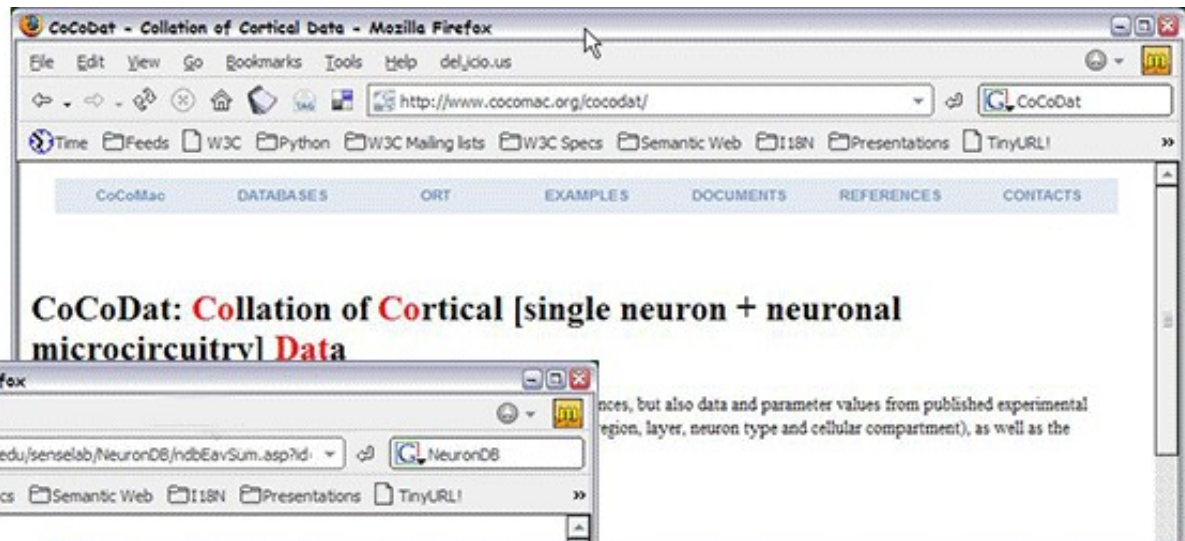
- Your automatic airline reservation
 - knows about your preferences
 - builds up knowledge base using your past
 - can combine the local knowledge with remote services:
 - airline preferences
 - dietary requirements
 - calendaring
 - etc
- It communicates with remote information (i.e., on the Web!)
 - (M. Dertouzos: The Unfinished Revolution)

> Example: data(base) integration

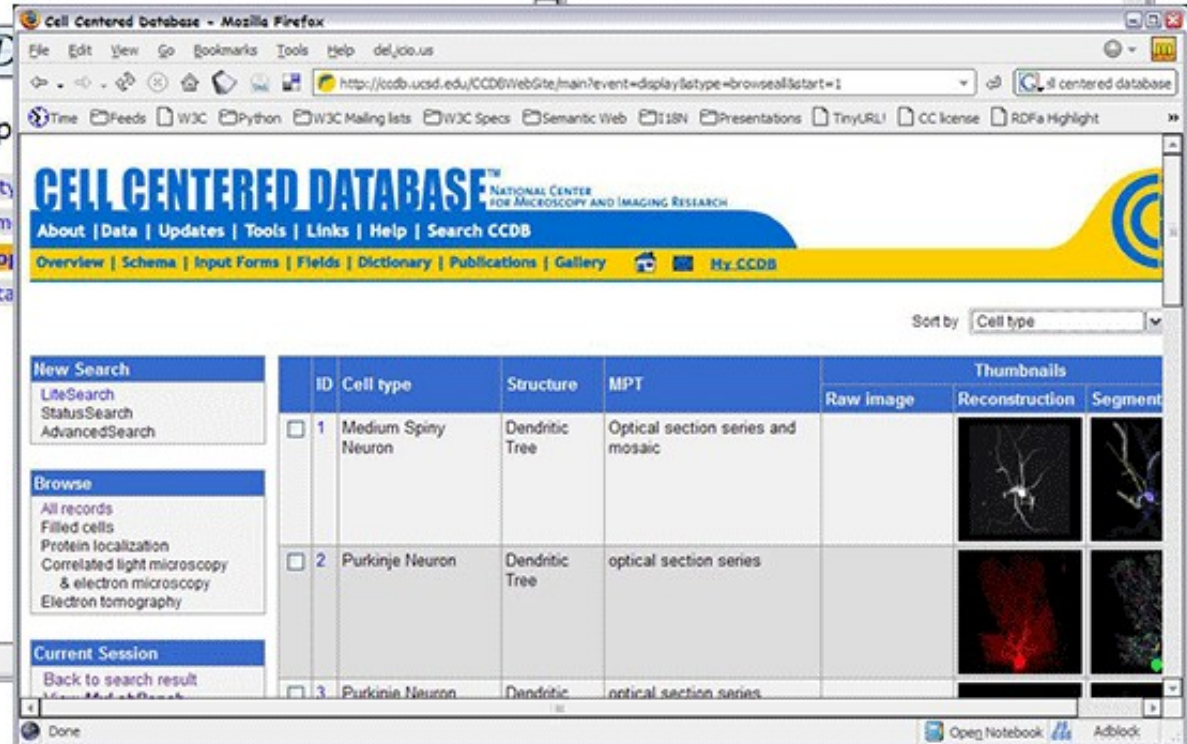
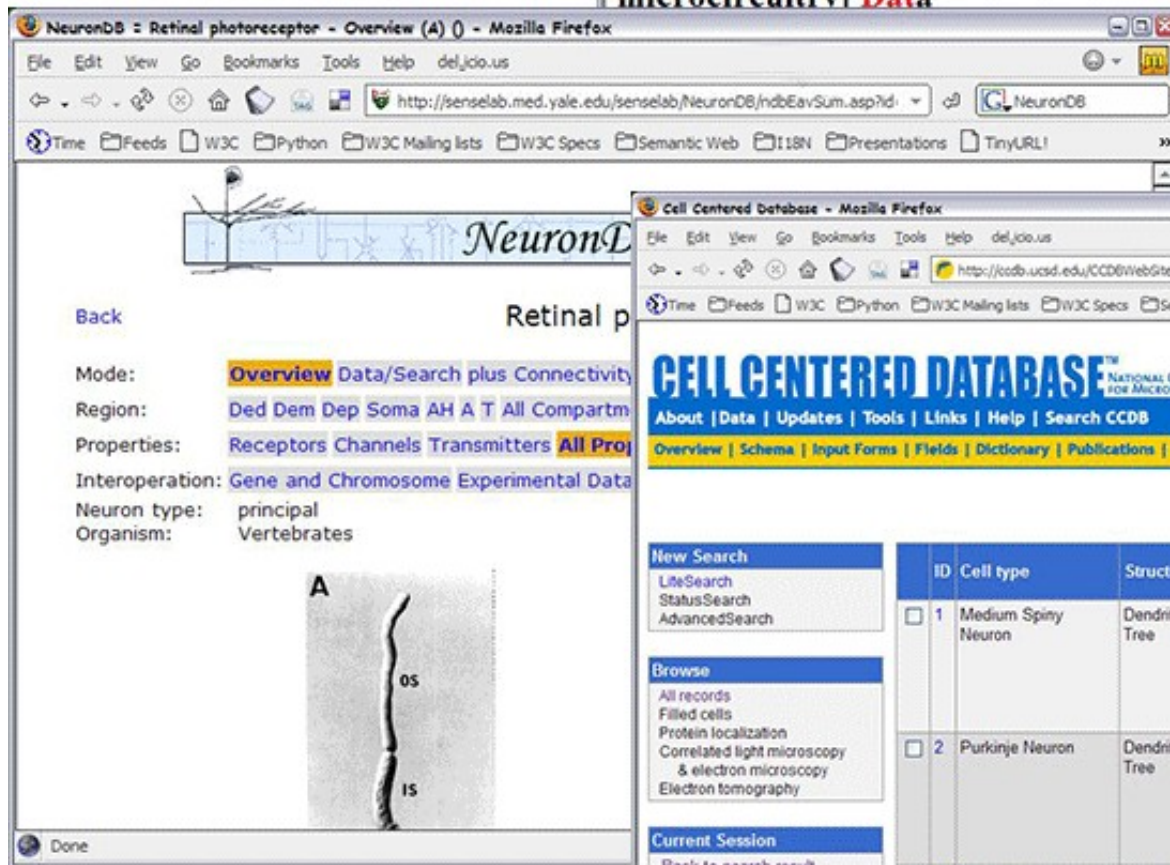


- Databases are very different in structure, in content
- Lots of applications require managing several databases
 - after company mergers
 - combination of administrative data for e-Government
 - biochemical, genetic, pharmaceutical research
 - etc.
- Most of these data are accessible from the Web (though not necessarily public yet)

> And the problem is real...



nces, but also data and parameter values from published experimental region, layer, neuron type and cellular compartment), as well as the



> Example: Social Networks



- Social sites are everywhere these days (LinkedIn, Facebook, Dopplr, Digg, Plexo, Zyb, ...)
- Data is not interchangeable: how many times did you have to add your contacts? 😊
- Applications should be able to get to those data via standard means
 - there are, of course, privacy issues...

> What is needed?



- (Some) data should be available for machines for further processing
- Data should be possibly combined, merged on a Web scale
- Sometimes, data may describe other data (like the library example, using metadata)...
- ... but sometimes the data is to be exchanged by itself, like my calendar or my travel preferences
- Machines may also need to reason about that data

> In what follows...



- We will use a simplistic example to introduce the main Semantic Web concepts
- We take, as an example area, data integration

> The rough structure of data integration



1. Map the various data onto an abstract data representation
 - make the data independent of its internal representation...
2. Merge the resulting representations
3. Start making queries on the whole!
 - queries that could not have been done on the individual data sets

> A simplified bookstore data (dataset “A”)

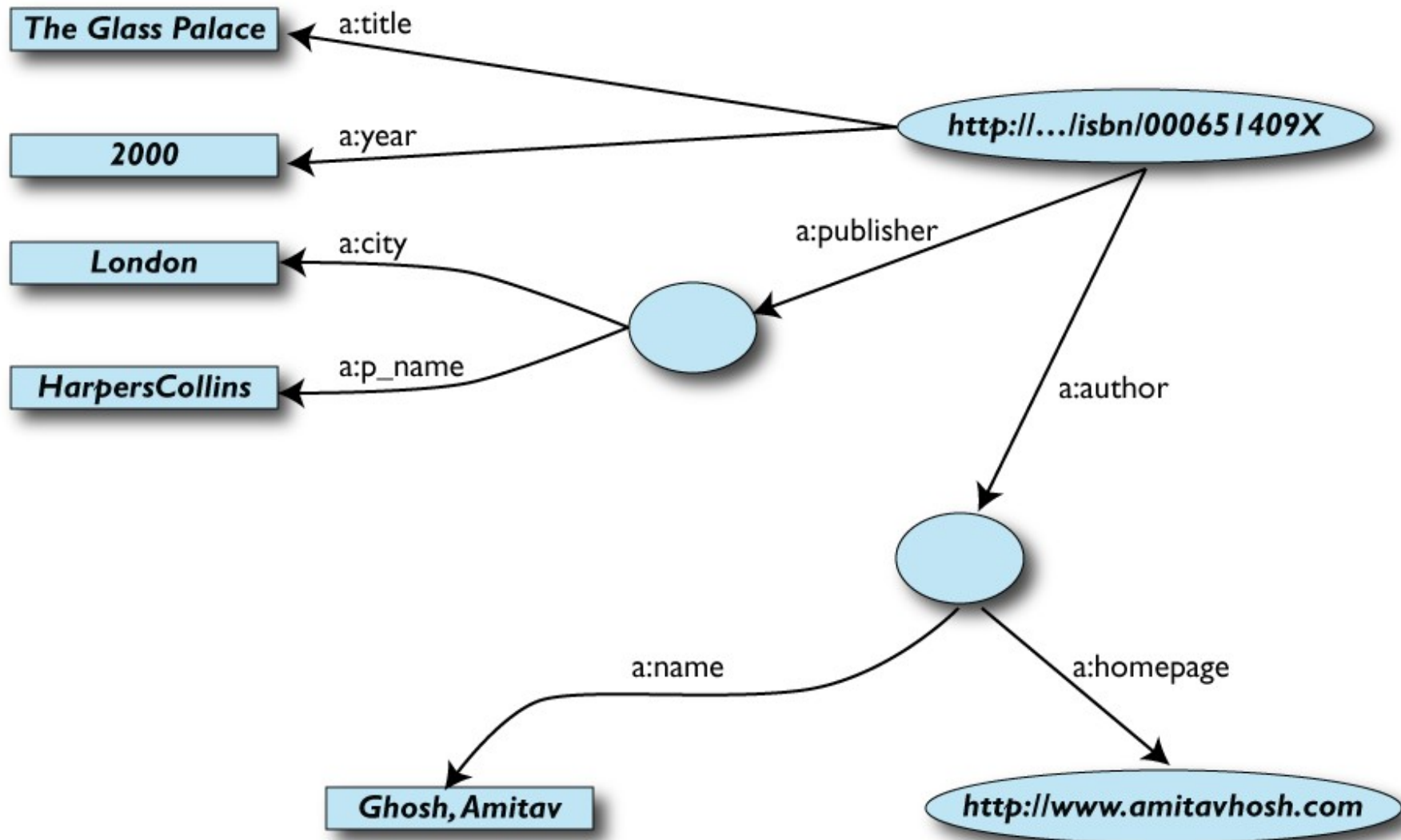


ID	Author	Title	Publisher	Year
ISBN0-00-651409-X	id_xyz	The Glass Palace	id_qpr	2000

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

ID	Publ. Name	City
id_qpr	Harpers Collins	London

> 1st: export your data as a set of relations



> Some notes on the exporting the data



- Relations form a graph
 - the nodes refer to the “real” data or contain some literal
 - how the graph is represented in machine is immaterial for now
- Data export does not necessarily mean physical conversion of the data
 - relations can be generated on-the-fly at query time
 - via SQL “bridges”
 - scraping HTML pages
 - extracting data from Excel sheets
 - etc.
- One can export part of the data

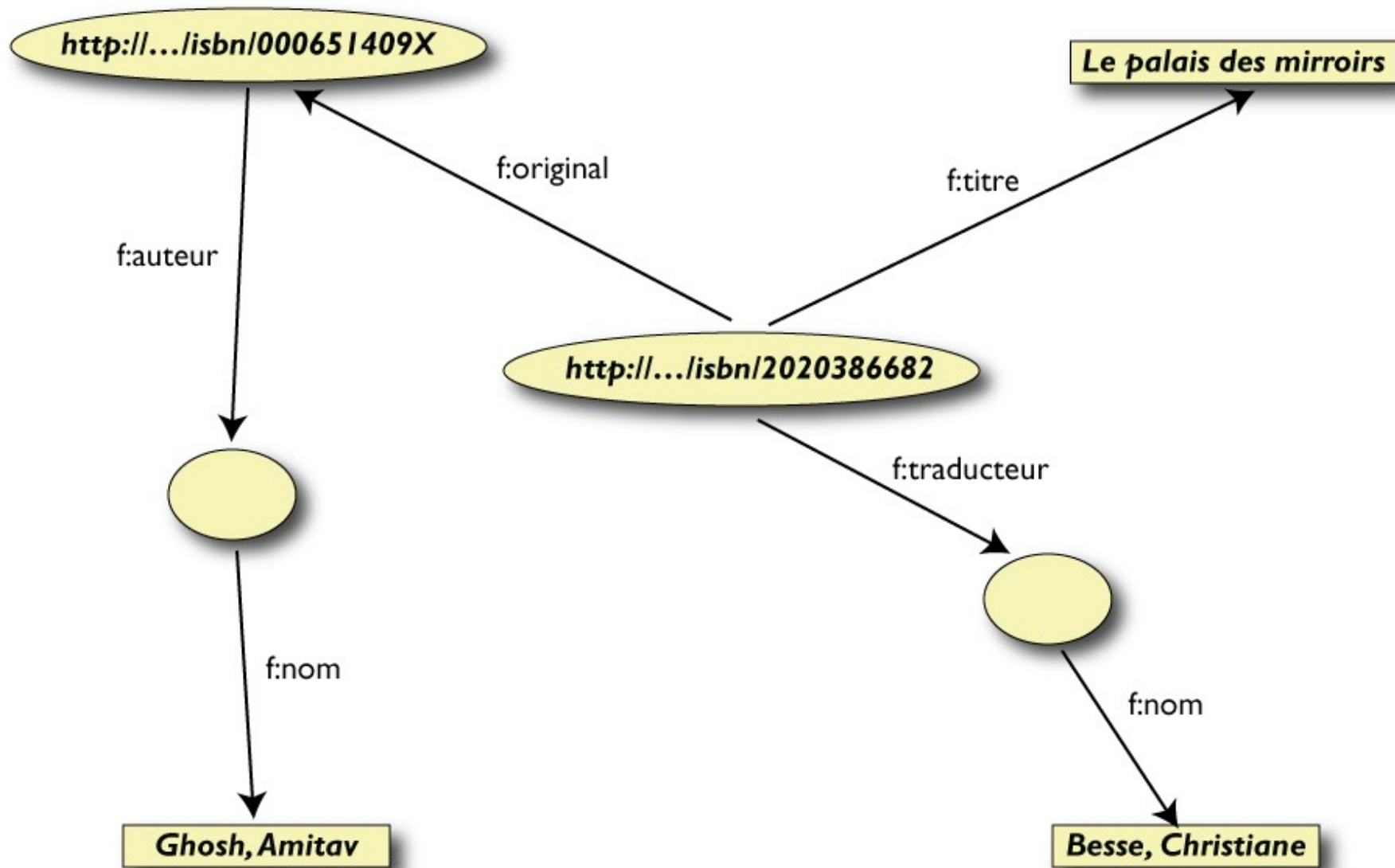
> Another bookstore data (dataset “F”)



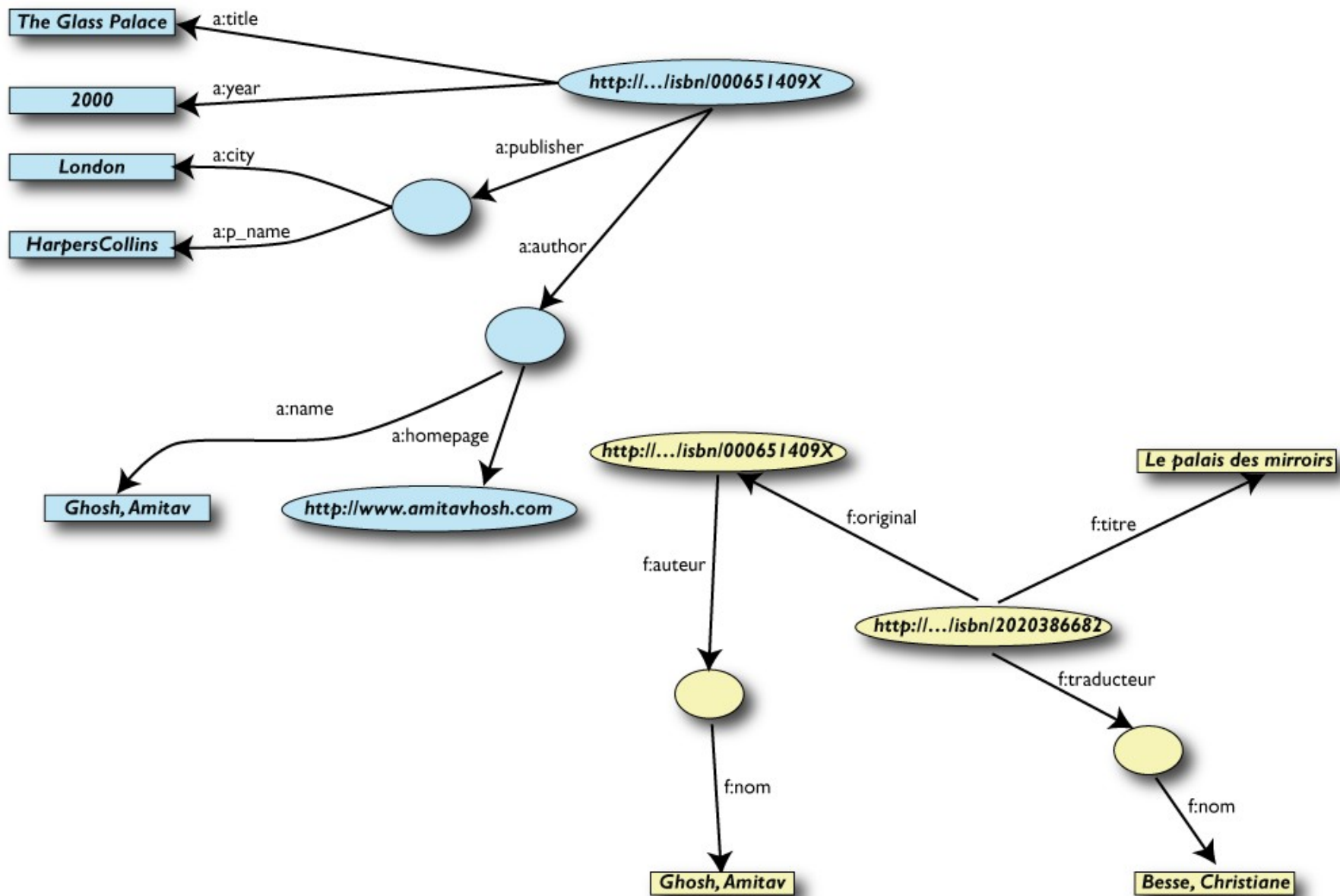
ID	Titre	Auteur	Traducteur	Original
ISBN0 2020386682	Le Palais des miroirs	i_abc	id_qrs	ISBN-0-00-651409-X

ID	Nom
id_abc	Ghosh, Amitav
id_qrs	Besse, Christiane

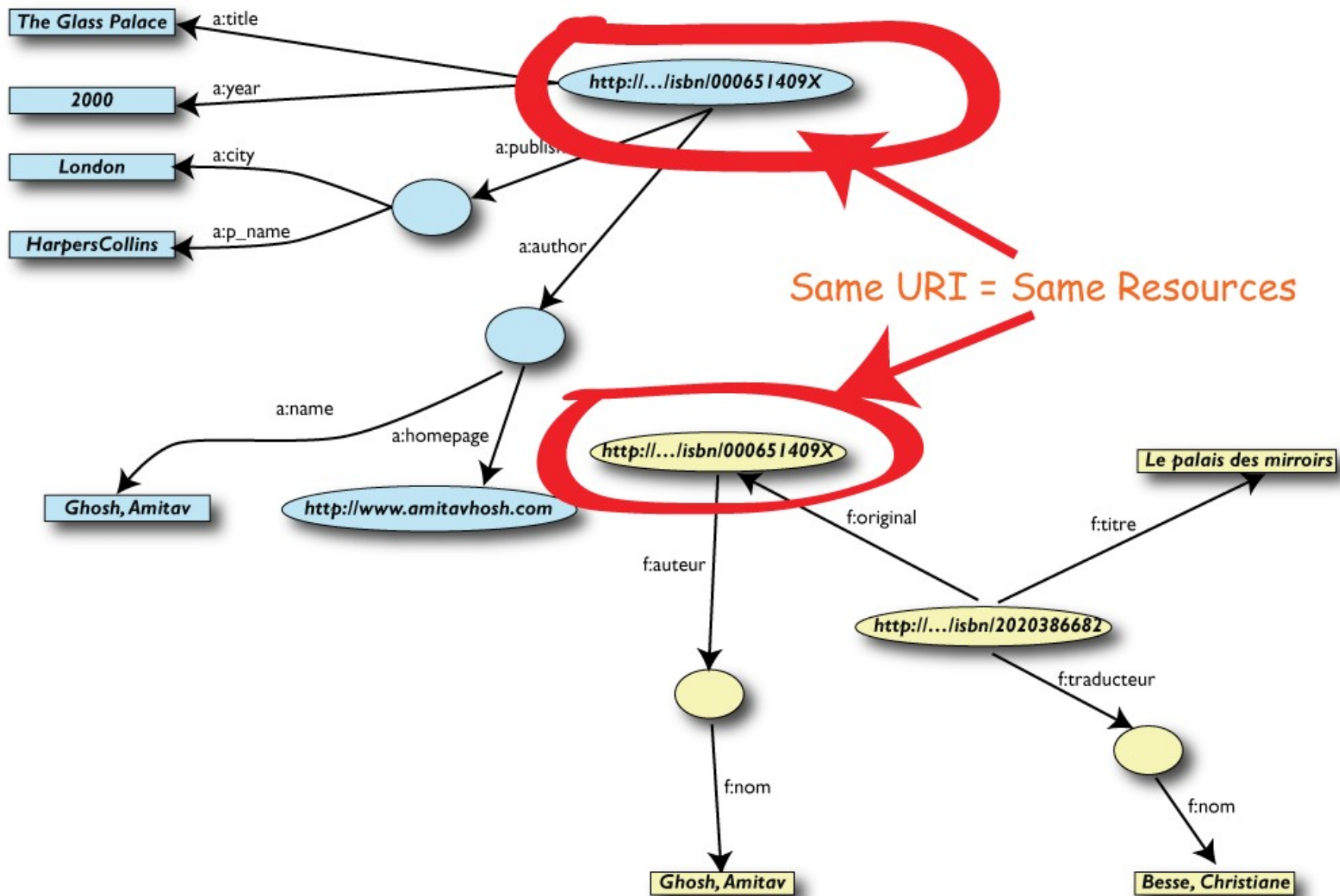
> 2nd: export your second set of data



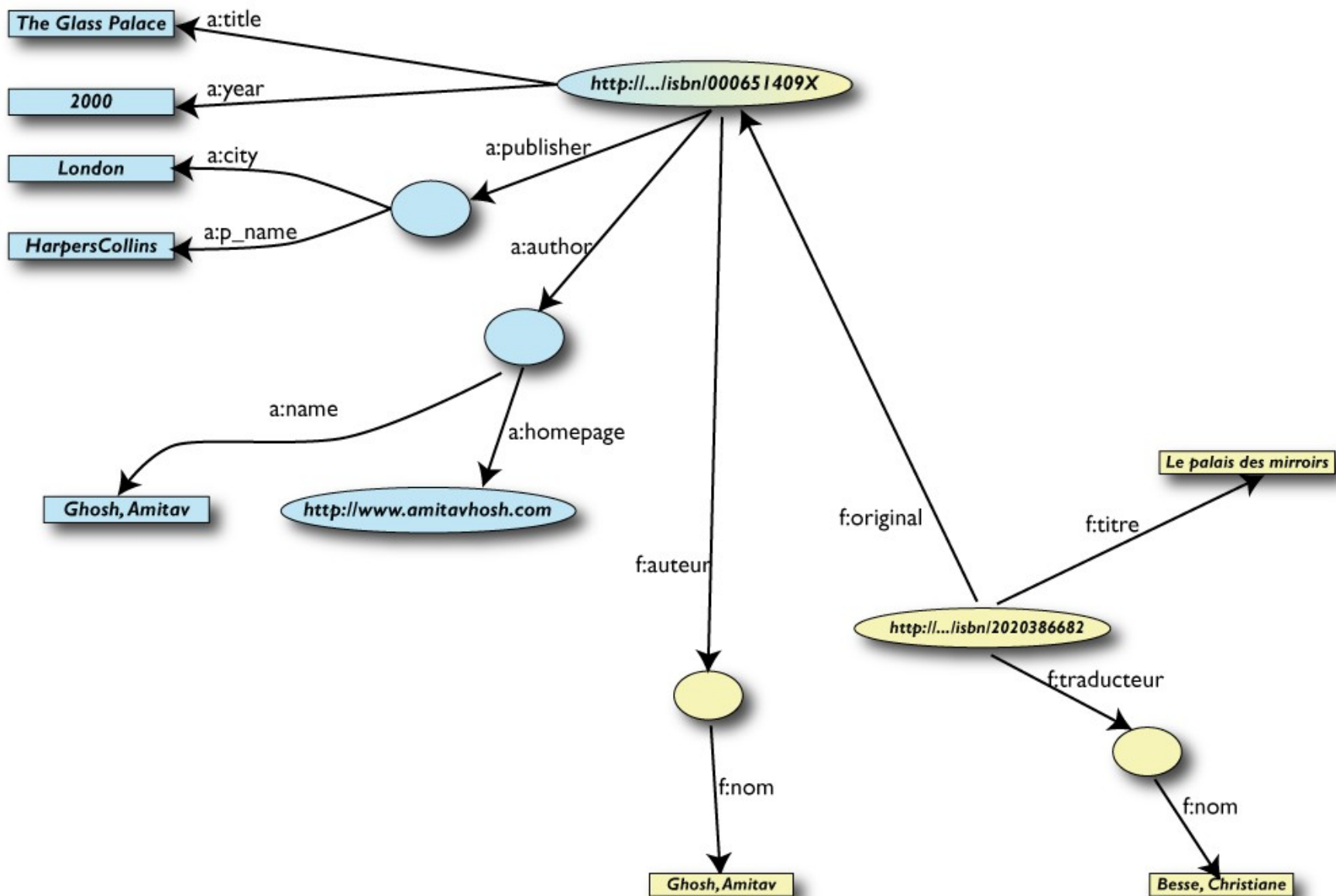
> 3rd: start merging your data



> 3rd: start merging your data (cont.)



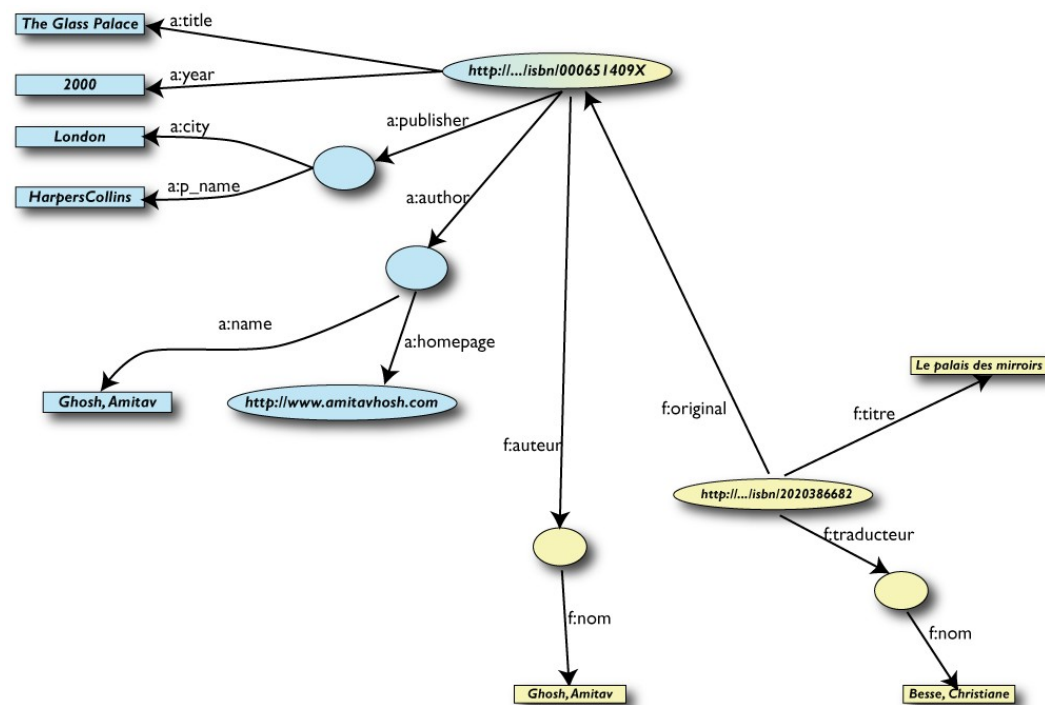
> 3rd: merge identical resources



> Start making queries...



- User of data “F” can now ask queries like:
 - « donnez-moi le titre de l’original »
 - (ie: “give me the title of the original”)
- This information is not in the dataset “F”...
- ...but can be retrieved by merging with dataset “A”!

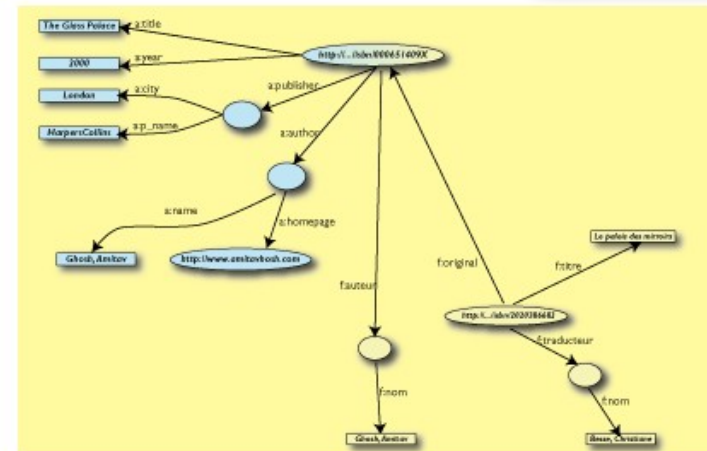
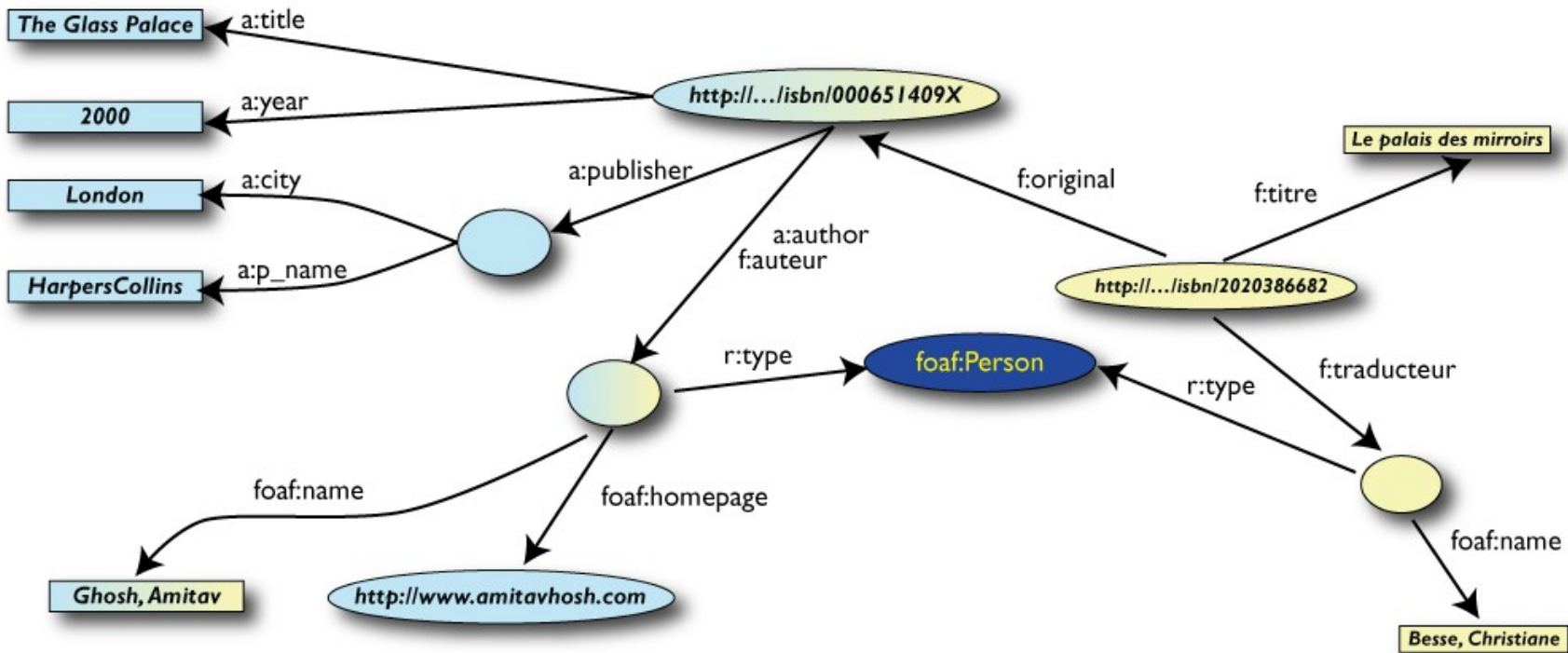


> However, more can be achieved...



- We “feel” that **a:author** and **f:auteur** should be the same
- But an automatic merge does 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

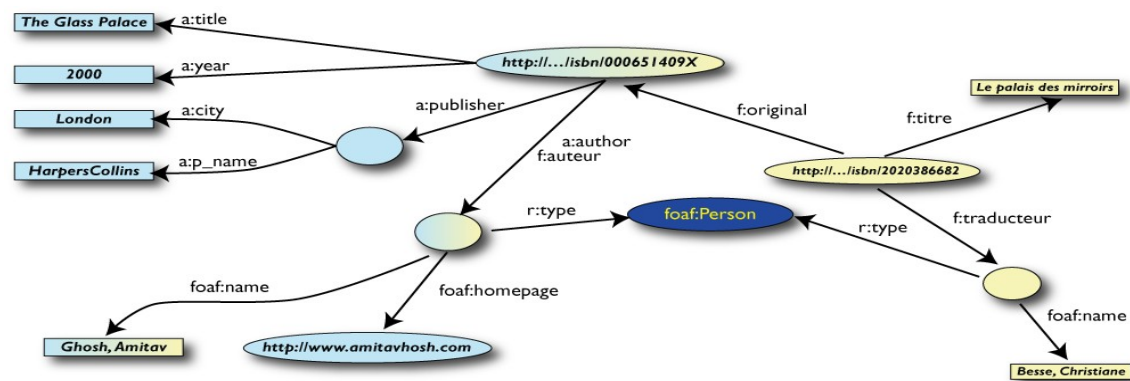
> 3rd revisited: use the extra knowledge



> Start making richer queries!



- User of dataset “F” can now query:
 - « donnes-moi la page d'accueil de l'auteur de l'original »
 - (ie, “give me the home page of the original's author”)
- 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”

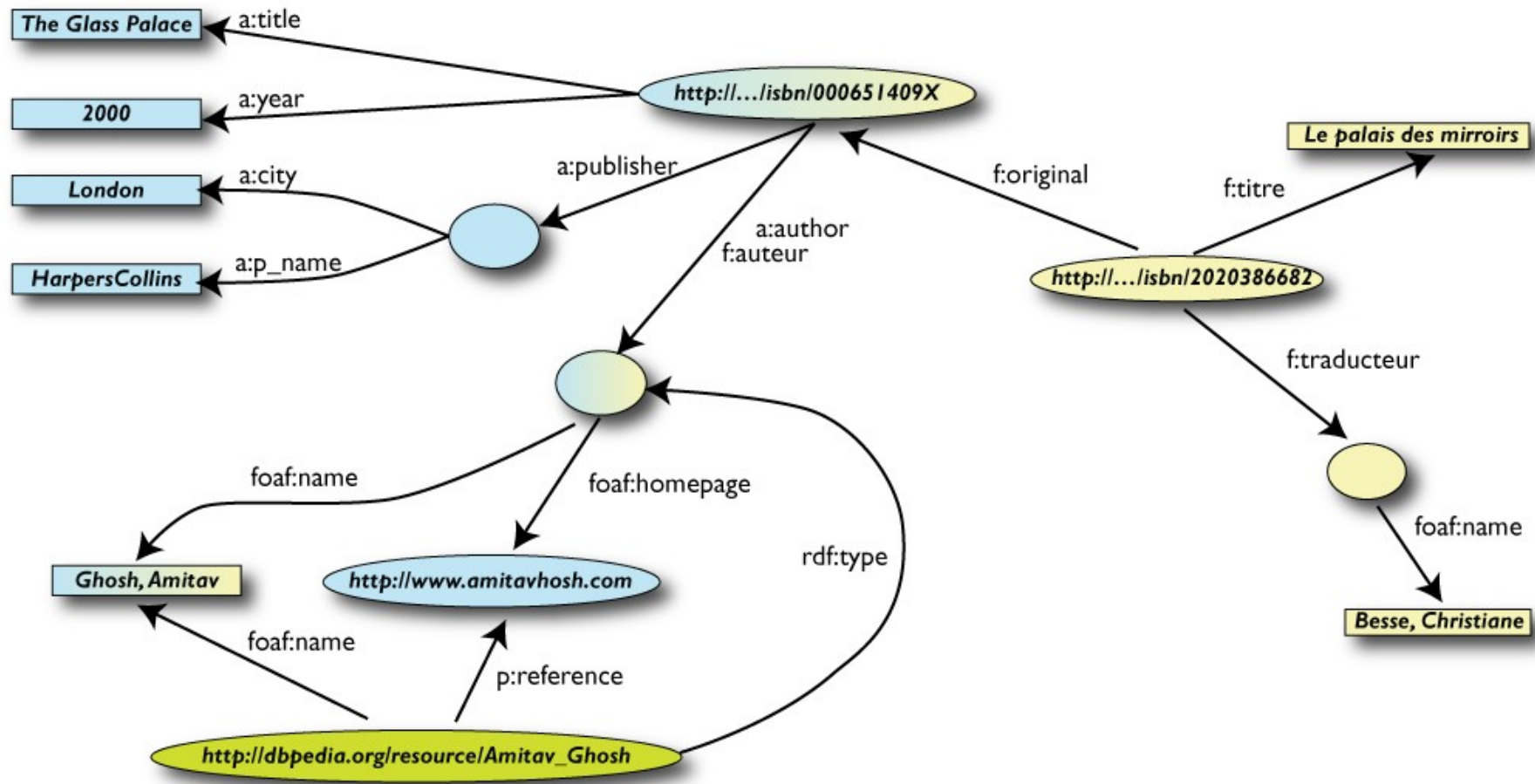


> Combine with different datasets

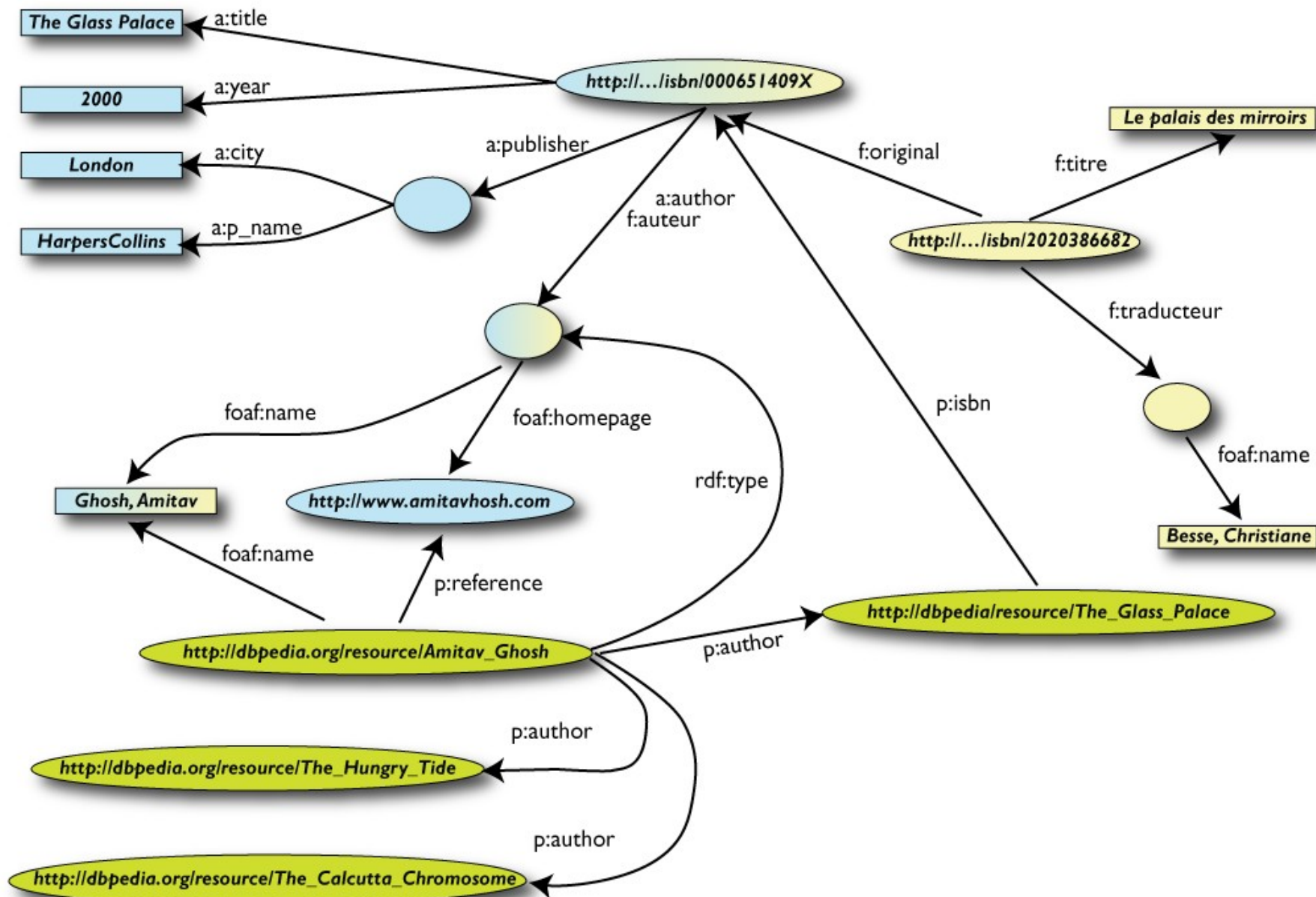


- Using, e.g., the “Person”, the dataset can be combined with other sources
- For example, data in Wikipedia can be extracted using dedicated tools
 - there is an active development to add some simple semantic “tag” to wikipedia entries (so called “Semantic Wiki”-s)
 - the “[DBpedia](#)” project can extract the “infobox” information from Wikipedia already... (see later)

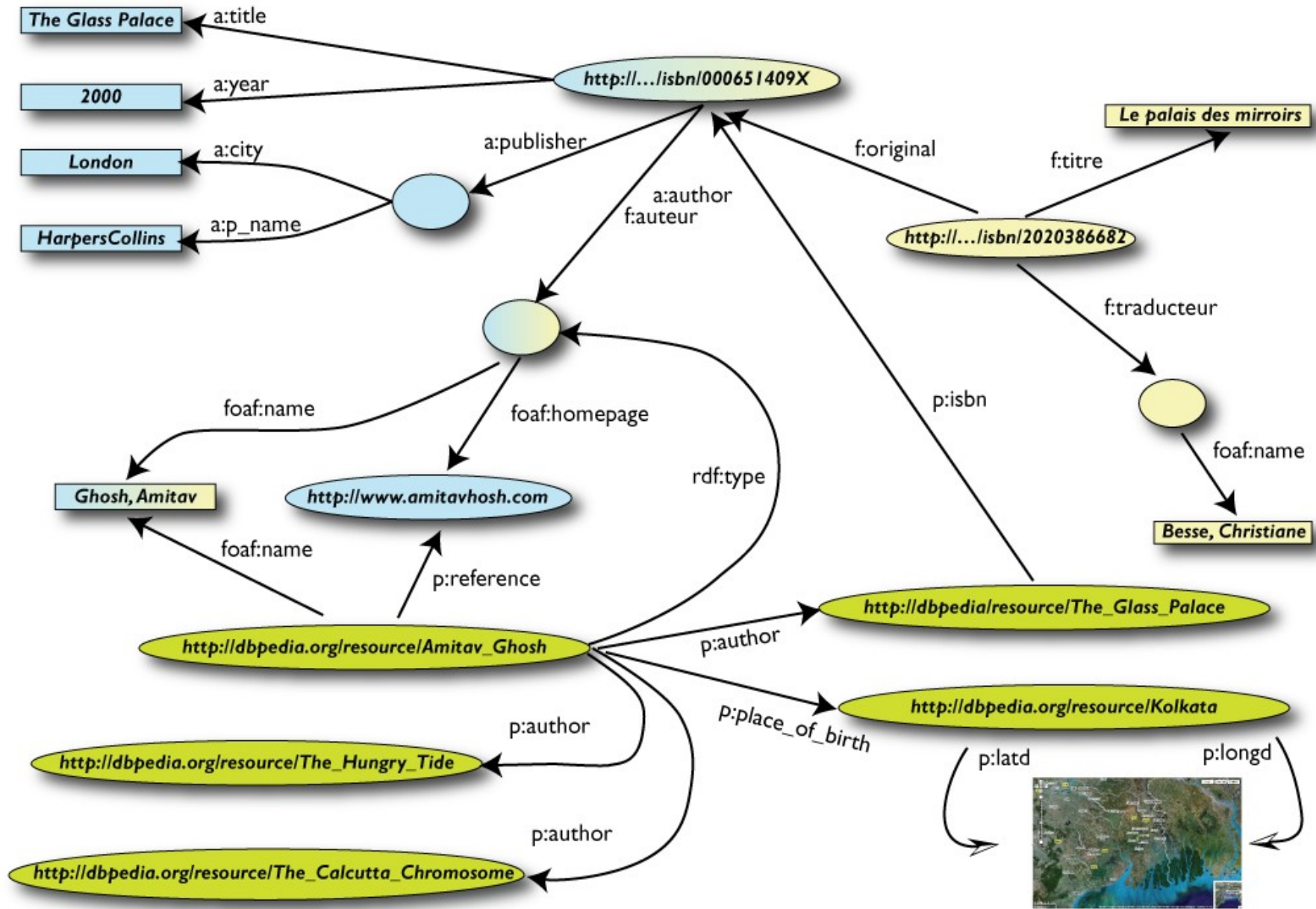
> Merge with Wikipedia data



> Merge with Wikipedia data



> Merge with Wikipedia data



> Is that surprising?



- Maybe but, in fact, no...
- What happened via automatic means is done all the time, every day by the users of the Web!
- The difference: a bit of extra rigor (e.g., naming the relationships) is necessary so that machines could do this, too

> What did we do?



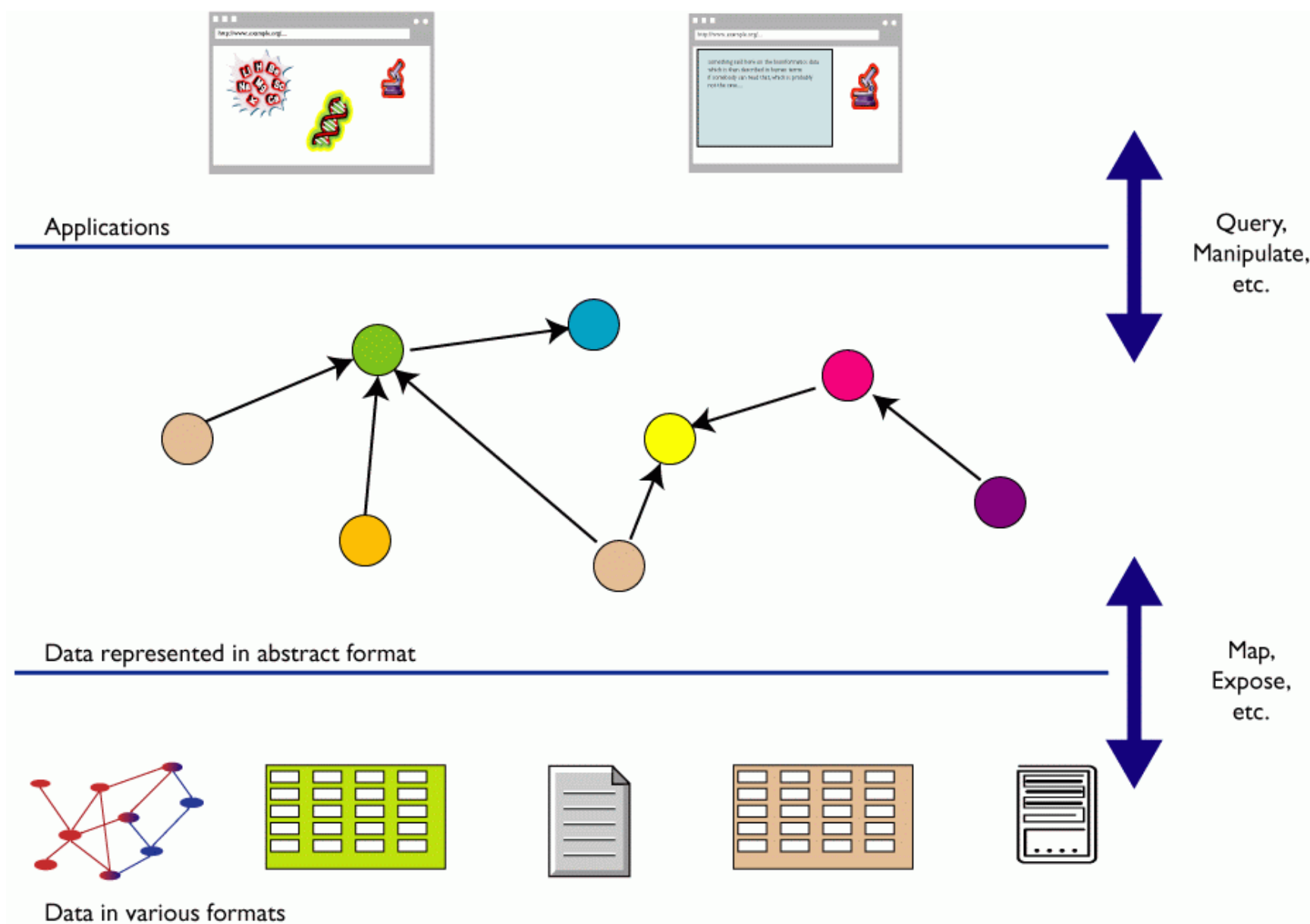
- We combined different datasets
 - all may be of different origin somewhere on the web
 - all may have different formats (mysql, excel sheet, XHTML, etc)
 - all may have different names for relations (e.g., multilingual)
- We could combine the data because some URI-s were identical (the ISBN-s in this case)
- We could add some simple additional information (the “glue”), also using common terminologies that a community has produced
- As a result, new relations could be found and retrieved

> It could become even more powerful



- We could add extra knowledge to the merged datasets
 - e.g., a full classification of various type of library data
 - geographical information
 - etc.
- This is where ontologies, thesauri, extra rules, etc, come in
 - ontologies/rule sets can be relatively simple and small, or...
 - huge ...
 - or anything in between... 😊
- Even more powerful queries can be asked as a result
 - e.g., in case of large ontologies the emphasis of queries may be drawn with the help of specialized engines

> What did we do? (cont)



> The abstraction pays off because...



- ... the graph representation is independent on the exact structures in, say, a relational database
- ... a change in local database schema's, XHTML structures, etc, do not affect the whole, only the “export” step
 - “schema independence”
- ... new data, new connections can be added seamlessly, regardless of the structure of other data sources

> So where is the Semantic Web?



- The Semantic Web provides technologies to make such integration possible!
- Hopefully you get a full picture at the end of the tutorial...



> RDF triples



- Let us begin to formalize what we did!
 - we “connected” the data...
 - but a simple connection is not enough... it should be named somehow
 - hence the RDF Triples: *a labeled connection between two resources*

> RDF triples (cont.)



- An RDF Triple (s, p, o) is such that:
 - “ s ”, “ p ” are URI-s, ie, resources on the Web; “ o ” is a URI or a literal
 - “ s ”, “ p ”, and “ o ” stand for “subject”, “predicate”, and “object”, respectively
 - conceptually: “ p ” connects, or relates the “ s ” and “ o ”
 - note that we use URI-s for naming: i.e., we can use `http://www.example.org/original`
 - here is the complete triple:

(`<http://...isbn...6682>`, `<http://.../original>`, `<http://...isbn...409X>`)

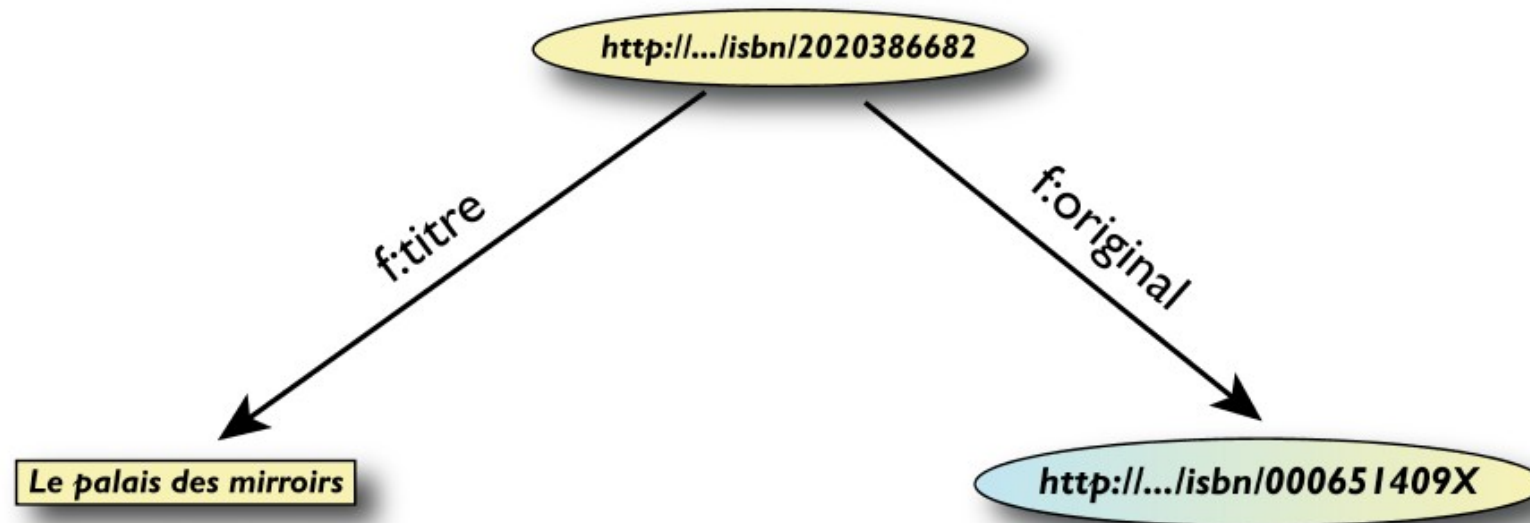
- RDF is a general model for such triples (with machine readable formats like RDF/XML, Turtle, N3, RXR, ...)
- ... and that's it!

> RDF triples (cont.)



- RDF triples are also referred to as “triplets”, or “statements”
- The “p” is also referred to as “property” in some cases
- Resources can use any URI; it can denote an element within an XML file on the Web, not only a “full” resource, e.g.:
 - `http://www.example.org/file.xml#element(home)`
 - `http://www.example.org/file.html#home`
 - `http://www.example.org/file2.xml#xpath1(//q[@a=b])`
- RDF triples form a directed, labeled graph (best way to think about them!)

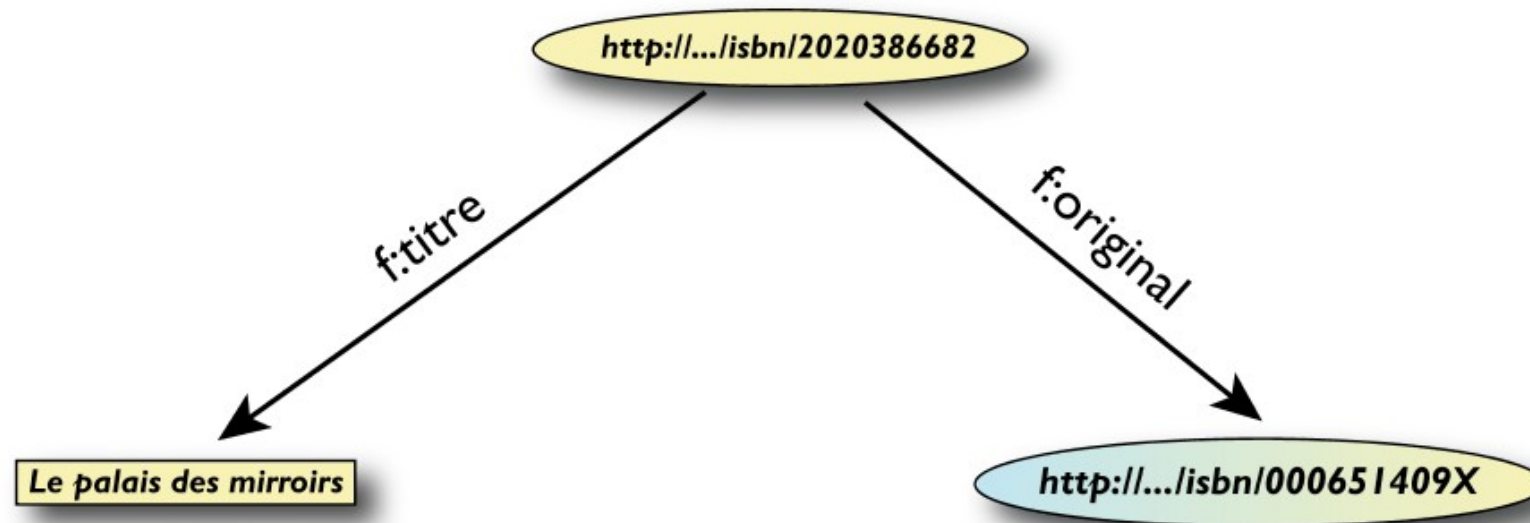
> A simple RDF example (in RDF/XML)



```
<rdf:Description rdf:about="http://.../isbn/2020386682">
  <f:titre xml:lang="fr">Le palais des miroirs</f:titre>
  <f:original rdf:resource="http://.../isbn/000651409X" />
</rdf:Description>
```

(Note: namespaces are used to simplify the URI-s)

> A simple RDF example (in Turtle)



```
<http://.../isbn/2020386682>  
  f:titre "Le palais des miroirs"@fr;  
  f:original <http://.../isbn/000651409X>.
```

> URI-s play a fundamental role

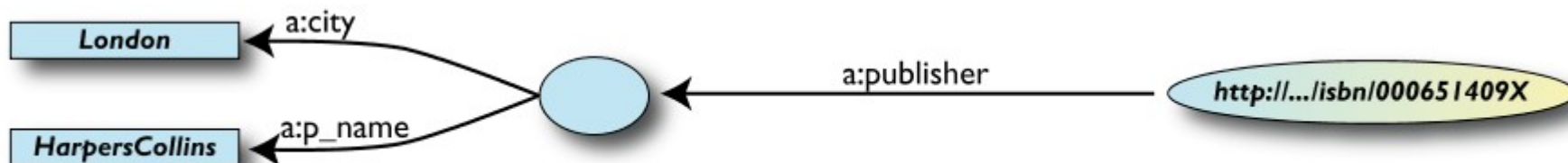


- URI-s made the merge possible
- Anybody can create (meta)data on any resource on the Web
 - e.g., the same XHTML file could be annotated through other terms
 - semantics is added to existing Web resources via URI-s
 - URI-s make it possible to link (via properties) data with one another
- URI-s ground RDF into the Web
 - information can be retrieved using existing tools
 - this makes the “Semantic Web”, well... “Semantic Web”

> “Internal” nodes



- Consider the following statement:
 - “the publisher is a «thing» that has a name and an address”
- Until now, nodes were identified with a URI. But...
- ...what is the URI of «thing»?



> One solution: create an extra URI



```
<rdf:Description rdf:about="http://.../isbn/000651409X">
  <a:publisher rdf:resource="urn:uuid:f60ffb40-307d-..." />
</rdf:Description>
<rdf:Description rdf:about="urn:uuid:f60ffb40-307d-...">
  <a:p_name>HarpersCollins</a:p_name>
  <a:city>HarpersCollins</a:city>
</rdf:Description>
```

- The resource will be “visible” on the Web as all other resources
 - care should be taken to define unique URI-s (hence the UUID in the example)
- Serializations may give syntactic help to define local URI-s (much like the id-s in HTML)

> Internal identifier (blank nodes)



```
<rdf:Description rdf:about="http://.../isbn/000651409X">
  <a:publisher rdf:nodeID="A234"/>
</rdf:Description>
<rdf:Description rdf:nodeID="A234">
  <a:p_name>HarpersCollins</a:p_name>
  <a:city>HarpersCollins</a:city>
</rdf:Description>
```

```
<http://.../isbn/2020386682> a:publisher _:A234.
_:A234 a:p_name "HarpersCollins".
```

- The exact syntax depends on the serialization format
- A234 is invisible from outside (it is not a “real” URI!); it is an internal identifier for a resource

> Blank nodes: the system can also do it



- Let the system create a “nodeID” internally (you do not really care about the name...)
- The example below is in Turtle (RDF/XML has something similar):

```
<http://.../isbn/000651409X> a:publisher [  
  a:p_name "HarpersCollins";  
  ...  
].
```



> Blank nodes: some more remarks



- Blank nodes require attention when merging
 - blank nodes with identical nodeID-s in different graphs are different
 - implementations must be careful with their naming schemes when merging
- Many applications prefer not to use blank nodes and define new URI-s “on-the-fly”
 - eg, when triples are in a database
- From a logic point of view, blank nodes represent an “existential” statement (“there is a resource such that...”)

> RDF in programming practice



- For example, using Java+Jena (HP's Bristol Lab):
 - a “Model” object is created
 - the RDF file is parsed and results stored in the Model
 - the Model offers methods to retrieve:
 - triples
 - (property,object) pairs for a specific subject
 - (subject,property) pairs for specific object
 - etc.
 - the rest is conventional programming...
- Similar tools exist in Python, PHP, etc.

> Jena example



```
// create a model
Model model=new ModelMem();
Resource subject=model.createResource("URI_of_Subject")
// 'in' refers to the input file
model.read(new InputStreamReader(in));
StmtIterator iter=model.listStatements(subject,null,null);
while(iter.hasNext()) {
    st = iter.next();
    p = st.getProperty();
    o = st.getObject();
    do_something(p,o);
}
```

> Merge in practice



- Environments merge graphs automatically
 - e.g., in Jena, the Model can load several files
 - the load merges the new statements automatically

> RDF schemas



> Need for RDF schemas



- This is the simple form of our “extra knowledge”:
 - define the terms we can use
 - what restrictions apply
 - what extra relationships are there?
- This is where RDF Schemas come in
 - officially: “RDF Vocabulary Description Language”; the term “Schema” is retained for historical reasons...

> Classes, resources, ...



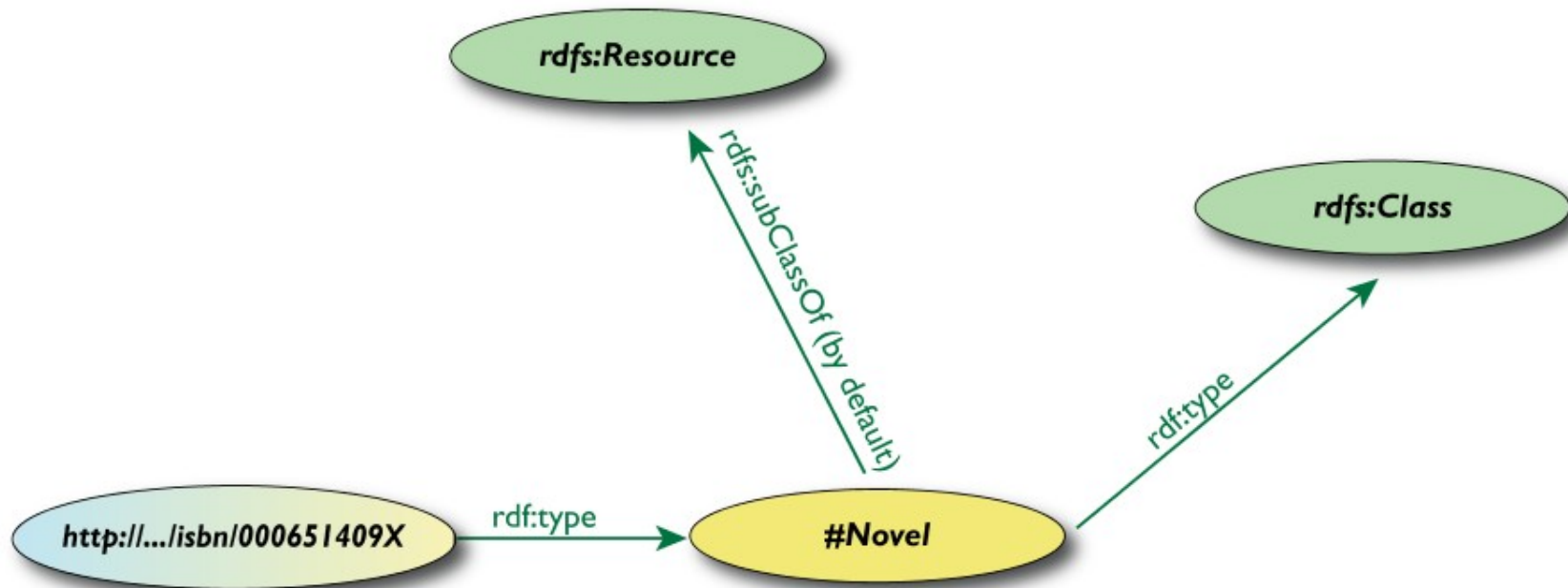
- Think of well known traditional ontologies or taxonomies:
 - use the term “novel”
 - “every novel is a fiction”
 - “«The Glass Palace» is a novel”
 - etc.
- RDFS defines resources and classes:
 - everything in RDF is a “resource”
 - “classes” are also resources, but...
 - ...they are also a collection of possible resources (i.e., “individuals”)
 - “fiction”, “novel”, ...

> Classes, resources, ... (cont.)



- Relationships are defined among classes/resources:
 - “typing”: an individual belongs to a specific class (“«The Glass Palace» is a novel”)
 - to be more precise: “«<http://.../000651409x>» is a novel”
 - “subclassing”: *all* instances of one are also the instances of the other (“every novel is a fiction”)
- *RDFS formalizes these notions in RDF*

> Classes, resources in RDF(S)



- RDFS defines `rdfs:Resource`, `rdfs:Class` as nodes; `rdf:type`, `rdfs:subClassOf` as properties
 - (these are all special URI-s, we just use the namespace abbreviation)

> Schema example in RDF/XML



- The schema part (“application’s data types”):

```
<rdf:Description rdf:ID="Novel">  
  <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>  
</rdf:Description>
```

- The RDF data on a specific novel (“using the type”):

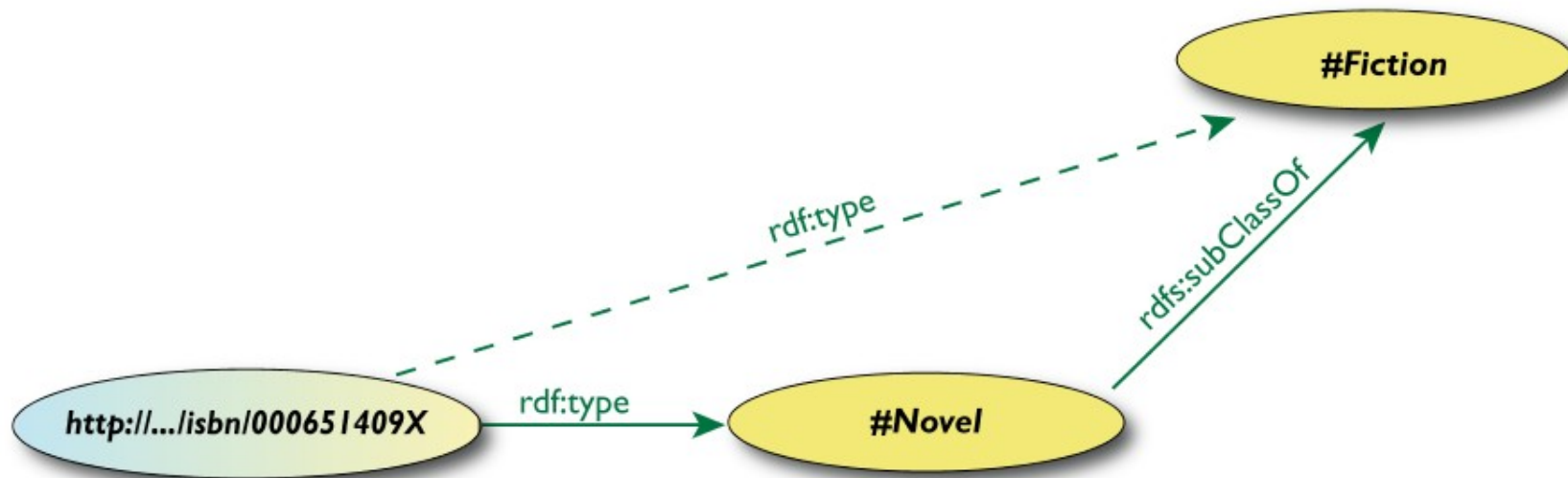
```
<rdf:Description rdf:about="http://.../isbn/000651409X">  
  <rdf:type rdf:resource="http://.../bookSchema.rdf#Novel"/>  
</rdf:Description>
```

> On types



- The `rdf:type` information may be very important for applications
 - e.g., it may be used for a categorization of possible nodes
 - probably the most frequently used RDF predicate...
- (remember the “Person” in our example?)

> Inferred properties



`(<http://.../isbn/000651409X> rdf:type #Fiction)`

- is not in the original RDF data...
- ...but can be inferred from the RDFS rules
- Better (“RDFS aware”) RDF environments return that triple, too

> Inference: let us be formal...



- The **RDF Semantics** document has a list of (44) entailment rules:
 - “if such and such triples are in the graph, add this and this triple”
 - do that recursively until the graph does not change
- The relevant rule for our example:

```
If:  
  uuu rdfs:subClassOf xxx .  
  vvv rdf:type uuu .  
Then add:  
  vvv rdf:type xxx .
```

- Whether those extra triplets are physically added to the graph or deduced when needed is an implementation issue

> Properties



- Property is a special class (**`rdf:Property`**)
 - properties are also resources identified by URI-s
- Properties' range and domain can be specified
 - i.e., what type of resources can serve as object and subject
- There is also a possibility for a “sub-property”
 - all resources bound by the “sub” are also bound by the other

> Properties (cont.)



- Properties are also resources (named via URI-s)...
- So properties of properties can be expressed as... RDF properties
 - this twists your mind a bit, but you can get used to it
- For example, (**P** **rdfs:range** **C**) means:
 - **P** is a property
 - **C** is a class instance
 - when using **P**, the “object” must be an individual in **C**
- This is an RDF statement with subject **P**, object **C**, and property **rdfs:range**

> Property specification serialized



- In RDF/XML:

```
<rdf:Property rdf:ID="title">  
  <rdfs:domain rdf:resource="#Fiction"/>  
  <rdfs:range rdf:resource="http://...#Literal"/>  
</rdf:Property>
```

- In Turtle:

```
:title  
  rdf:type      rdf:Property;  
  rdfs:domain   :Fiction;  
  rdfs:range    rdfs:Literal.
```

> What does this mean?



- Again, new relations can be deduced. Indeed, if

```
:title
  rdf:type      rdf:Property;
  rdfs:domain   :Fiction;
  rdfs:range    rdfs:Literal.

<http://.../isbn/000651409X> :title "The Glass Palace" .
```

- then the system can *infer* that:

```
<http://.../isbn/000651409X> rdf:type :Fiction .
```

> A bit of RDFS can take you far...



- Remember the power of merge?
- We could have used, in our example:
 - **f:auteur** is a subproperty of **a:author** and vice versa (although we will see other ways to do that...)
- Of course, in some cases, more complex knowledge is necessary (see later...)

> Get to RDF(S) data



> Simple approach



- Write RDF/XML or Turtle “manually”
- In some cases that is necessary, but it really does not scale...

> RDF can also be extracted/generated



- Use intelligent “scrapers” or “wrappers” to extract a structure (hence RDF) from a Web pages or XML files...
- ... and then generate RDF automatically (e.g., via an XSLT script)

> Formalizing the scraper approach: GRDDL



- **GRDDL** formalizes the scraper approach. For example:

```
<html xmlns="http://www.w3.org/1999/">
  <head profile="http://www.w3.org/2003/g/data-view">
    <title>Some Document</title>
    <link rel="transformation" href="http://.../dc-extract.xsl"/>
    <meta name="DC.Subject" content="Some subject"/>
    ...
  </head>
  ...
  <span class="date">2006-01-02</span>
  ...
</html>
```

- yields, by running the file through **dc-extract.xsl**:

```
<rdf:Description rdf:about="...">
  <dc:subject>Some subject</dc:subject>
  <dc:date>2006-01-02</dc:date>
</rdf:Description>
```



- The transformation itself has to be provided for each set of conventions (making use of meta-s, class id-s, etc...)
- A “bridge” to “microformats”
- A more general syntax is defined for XML formats in general (e.g., via the namespace document)
 - a method to get data in other formats to RDF (e.g., XBRL)

> Another upcoming solution: RDFa



- For example:

```
<div about="http://uri.to.newsitem">
  <span property="dc:date">March 23, 2004</span>
  <span property="dc:title">Rollers hit casino for £1.3m</span>
  By <span property="dc:creator">Steve Bird</span>. See
  <a href="http://www.a.b.c/d.avi" rel="dc:type:MovingImage">
    also video footage</a>...
</div>
```

- yields, by running the file through an RDFa processor:

```
<http://uri.to.newsitem>
  dc:date          "March 23, 2004";
  dc:title         "Rollers hit casino for £1.3m;
  dc:creator       "Steve Bird";
  dc:type:MovingImage <http://www.a.b.c/d.avi>.
```

> RDFa (cont.)



- RDFa extends (X)HTML a bit by:
 - defining general attributes to add metadata to any elements
 - provides an almost complete “serialization” of RDF in XHTML
- It is a bit like the microformats/GRDDL approach but with more rigor and fully generic
- See the separate tutorial this afternoon!

> Bridge to relational databases

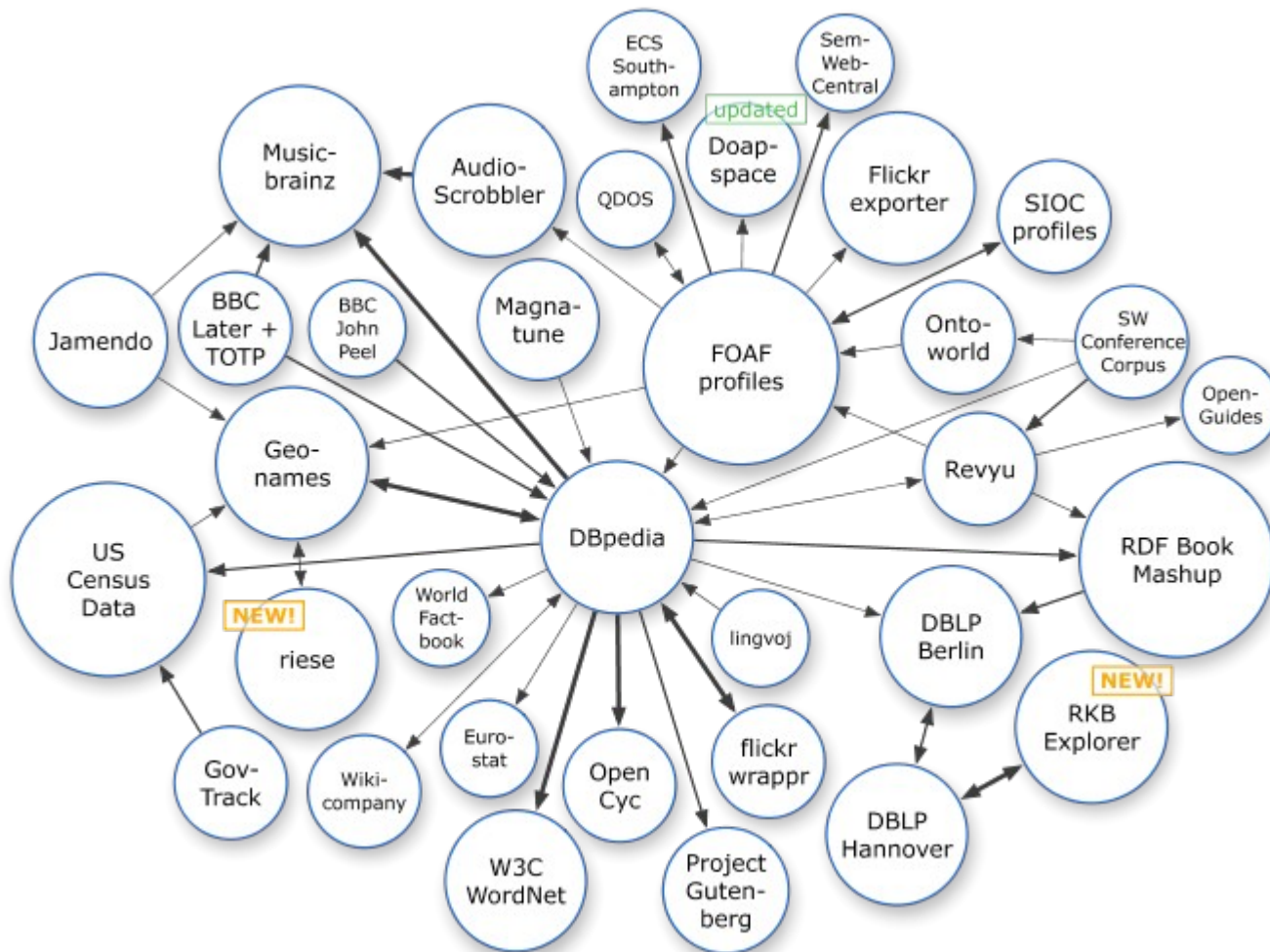


- Most of the data on the Web are stored in relational databases
 - “RDFying” them is not possible
 - relational databases are here to stay...
- “Bridges” are being defined:
 - a layer between RDF and the relational data
 - RDB tables are “mapped” to RDF graphs, possibly on the fly
 - different mapping approaches are being used
 - this is what our examples did...
 - a number RDB systems offer this facility already (eg, Oracle, OpenLink, ...)
- Work for a survey of mapping techniques has started at W3C

> A good example: Linking Open Data Project



- Goal: “expose” open datasets in RDF
- Set *RDF links among the data items* from different datasets
- Billions triples, millions of “links”



> Example data source: DBpedia



- Extracting structured data (“infobox”) from Wikipedia:

<http://en.wikipedia.org/wiki/Kolkata>

```
<http://dbpedia.org/resource/Kolkata>
  dbpedia:native_name "Kolkata (Calcutta)"@en;
  dbpedia:altitude "9";
  dbpedia:populationTotal "4580544";
  dbpedia:population_metro "14681589";
  geo:lat "22.56970024108887";
  ...
```

Kolkata (Chutiyon ka Shahar)

West Bengal • India



Victoria Memorial

Coordinates:  22.5697, 88.3697

Time zone	IST (UTC+5:30)
Area	1,480 km² (571 sq mi)
• Elevation	• 9 m (30 ft)
District(s)	Calcutta †
Population	14,681,589 (2001)
• Density	• 9,920/km² (25,693/sq mi)
• Metro	• 4,580,544
Mayor	Bikash Ranjan Bhattacharya
Codes	
• Pincode	• 700 xxx
• Telephone	• +91 (33)

Website: www.kolkatamycity.com 

† The Kolkata urban agglomeration also includes portions of North 24 Parganas and South 24 Parganas districts.

> Automatic links among open datasets



```
<http://dbpedia.org/resource/Kolkata>
```

```
owl:sameAs <http://sws.geonames.org/1275004/>;
```

```
...
```

DBpedia

```
<http://sws.geonames.org/1275004/>
```

```
owl:sameAs <http://DBpedia.org/resource/Kolkata>
```

```
wgs84_pos:lat "22.5697222";
```

```
wgs84_pos:long "88.3697222";
```

```
sws:population "4631392"
```

```
...
```

Geonames

Processors can switch automatically from one to the other...

> RDF data access, a.k.a. query (SPARQL)



> RDF data access



- How do I query the RDF data?
 - e.g., how do I get to the DBpedia data?

> Querying RDF graphs



- Remember the Jena idiom:

```
StmtIterator iter=model.listStatements(subject,null,null);  
while(iter.hasNext()) {  
    st = iter.next();  
    p = st.getProperty(); o = st.getObject();  
    do_something(p,o);  
}
```

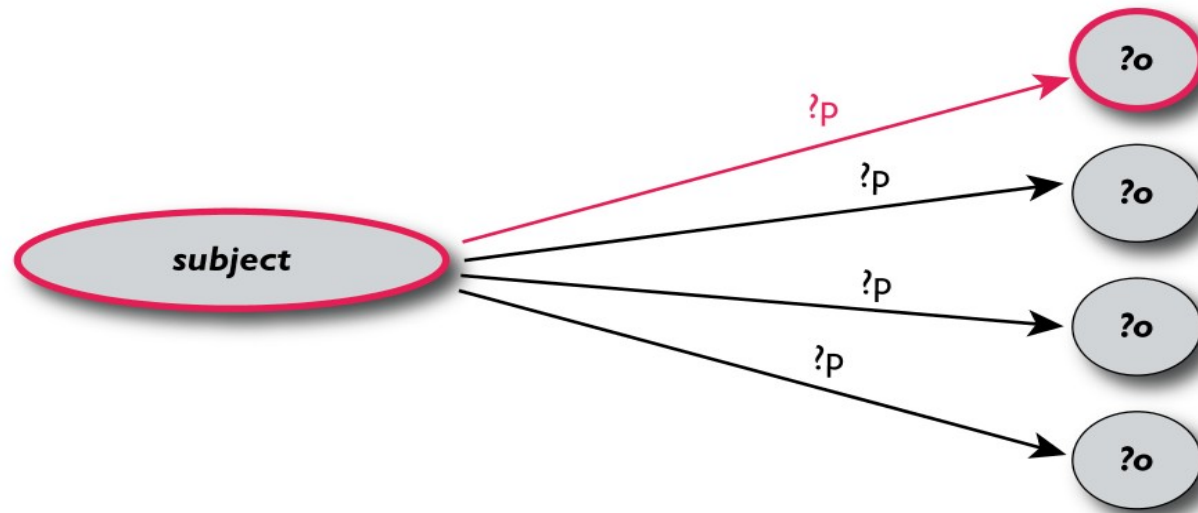
- In practice, more complex queries into the RDF data are necessary
 - something like: “give me the (a,b) pair of resources, for which there is an x such that (x parent a) and (b brother x) holds” (ie, return the uncles)
 - these rules may become quite complex
- This is the goal of **SPARQL** (Query Language for RDF)

> Analyze the Jena example



```
StmtIterator iter=model.listStatements(subject,null,null);  
while(iter.hasNext()) {  
    st = iter.next();  
    p = st.getProperty(); o = st.getObject();  
    do_something(p,o);  
}
```

- The $(\text{subject}, ?p, ?o)$ is a *pattern* for what we are looking for (with $?p$ and $?o$ as “unknowns”)



> General: graph patterns



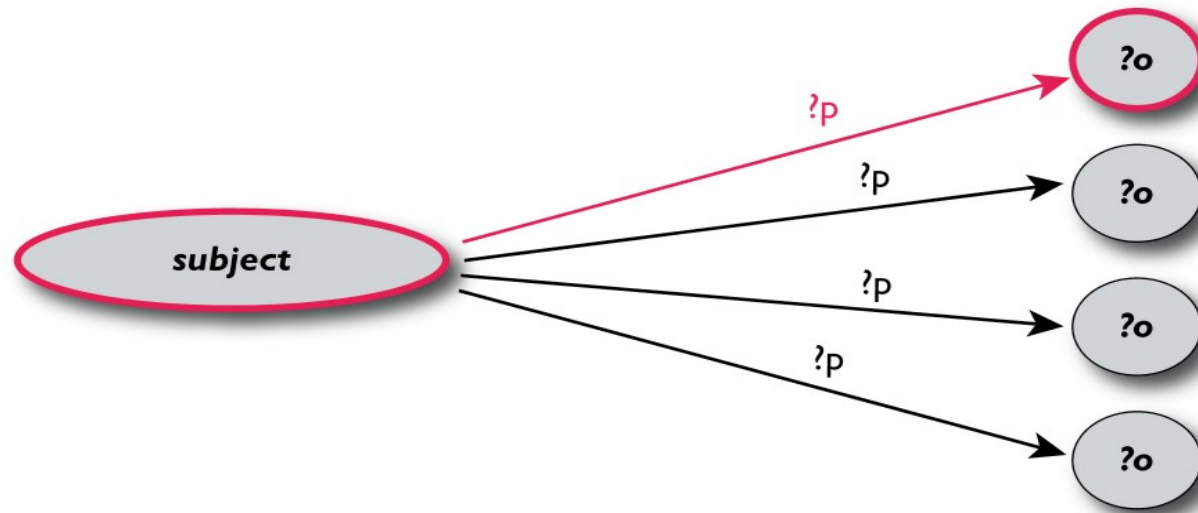
- The fundamental idea: generalize the approach to graph patterns:
 - the pattern contains unbound symbols
 - by binding the symbols (if possible), subgraphs of the RDF graph are selected
 - if there is such a selection, the query returns the bound resources
- SPARQL
 - is based on similar systems that already existed in some environments
 - is a programming language-independent query language

> Our Jena example in SPARQL



```
SELECT ?p ?o  
WHERE {subject ?p ?o}
```

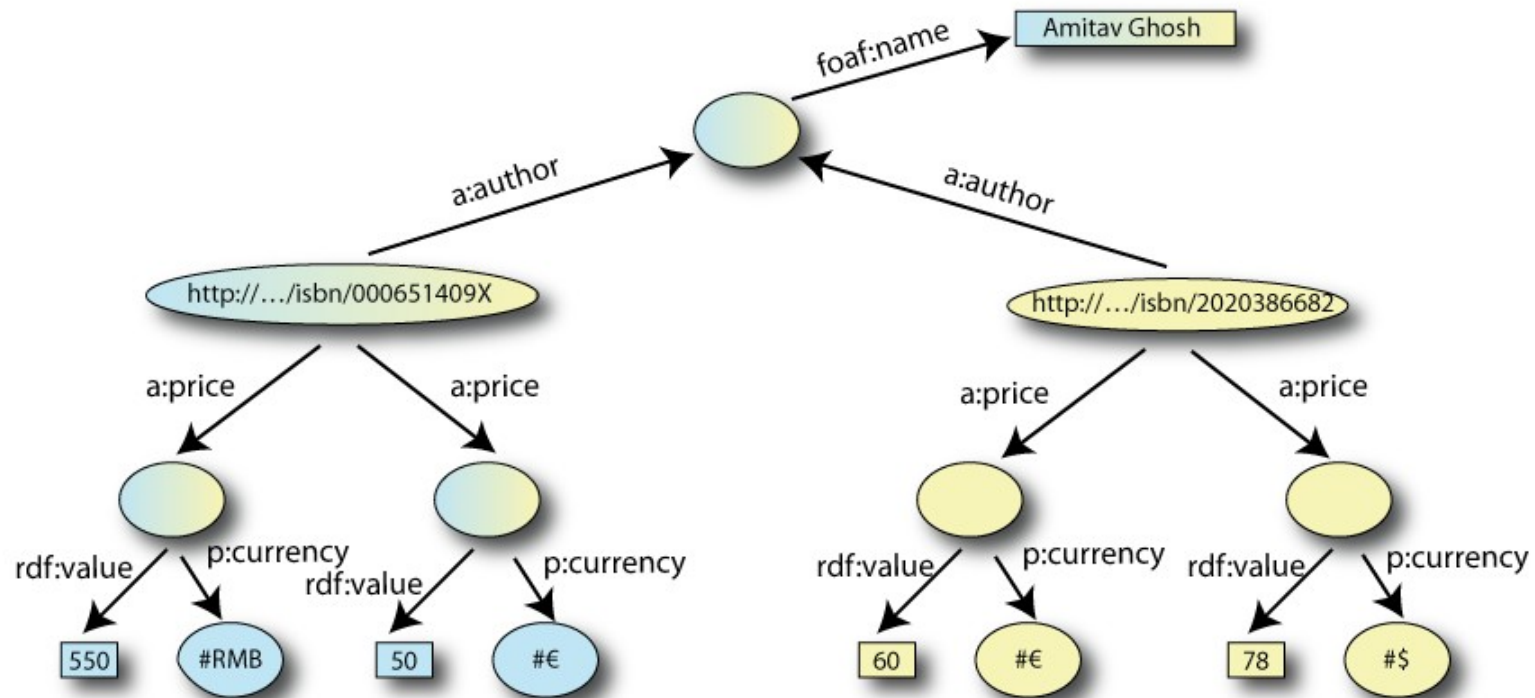
- The triples in **WHERE** define the graph pattern, with **?p** and **?o** “unbound” symbols
- The query returns all **p,o** pairs



> Simple SPARQL example



```
SELECT ?isbn ?price ?currency # note: not ?x!  
WHERE { ?isbn a:price ?x. ?x rdf:value ?price. ?x p:currency ?currency. }
```

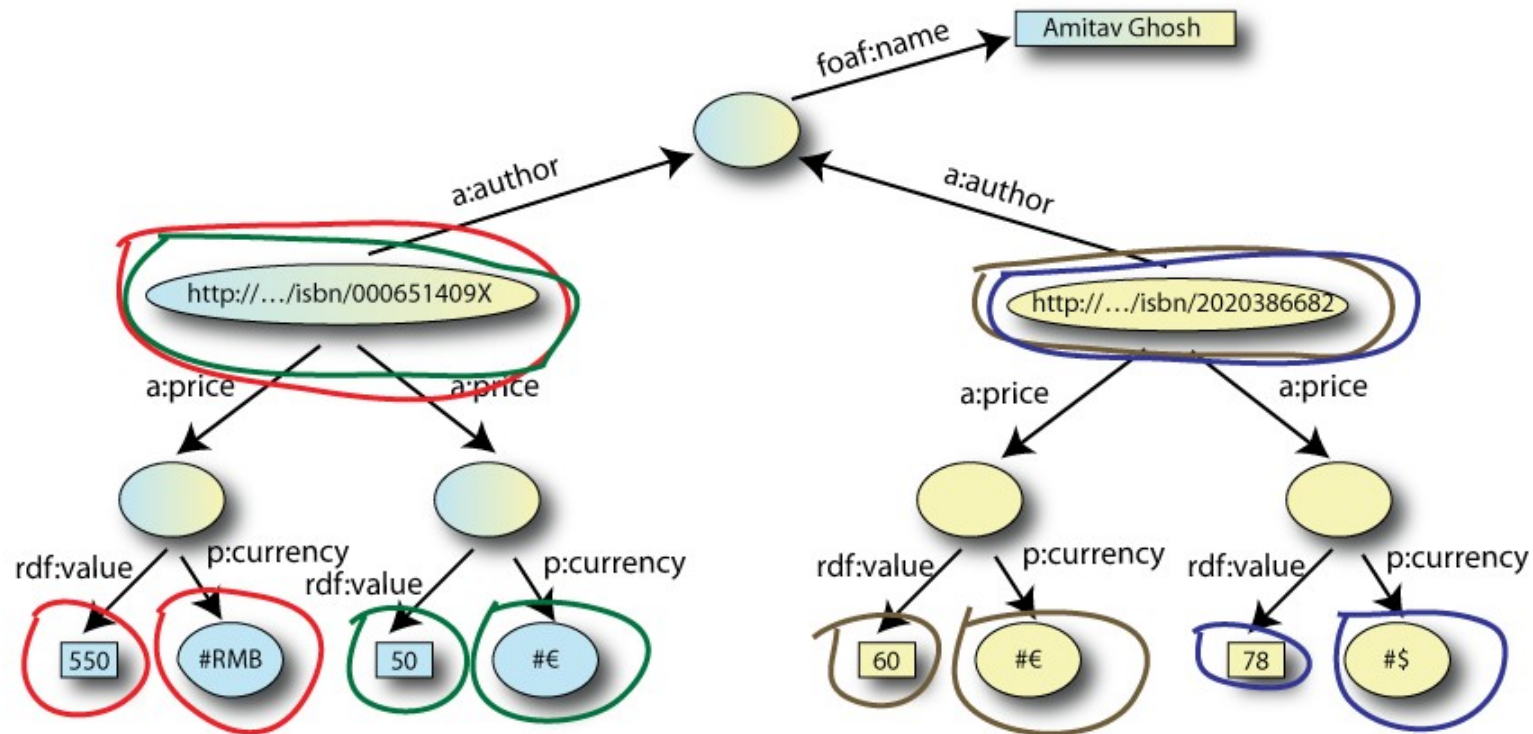


> Simple SPARQL example



```
SELECT ?isbn ?price ?currency # note: not ?x!  
WHERE { ?isbn a:price ?x. ?x rdf:value ?price. ?x p:currency ?currency. }
```

- Returns:
[<..49X>,550,RMB], [<..49X>,50,€], [<..6682>,60,€],
[<..6682>,78,\$]]

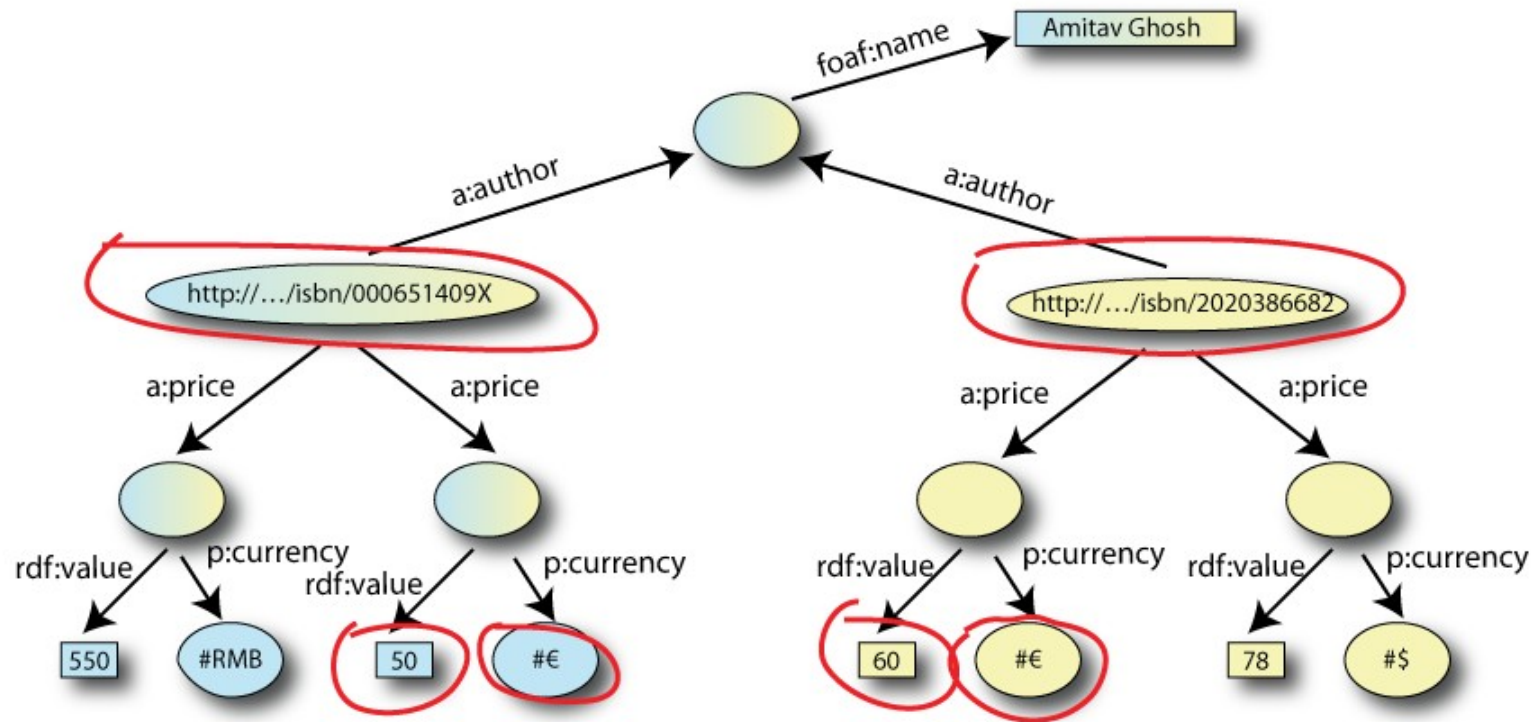


> Pattern constraints



```
SELECT ?isbn ?price ?currency # note: not ?x!  
WHERE { ?isbn a:price ?x. ?x rdf:value ?price. ?x p:currency ?currency.  
        FILTER(?currency == € )
```

- Returns: [[<..409X>,50,€], [<..6682>,60,€]]



> Other SPARQL features



- Some of the patterns may be optional
- Limit the number of returned results; remove duplicates
- Specify several data sources (via URI-s) within the query (essentially, a merge!)
- Construct a graph combining a separate pattern and the query results
- Use datatypes and/or language tags when matching a pattern

> SPARQL usage in practice



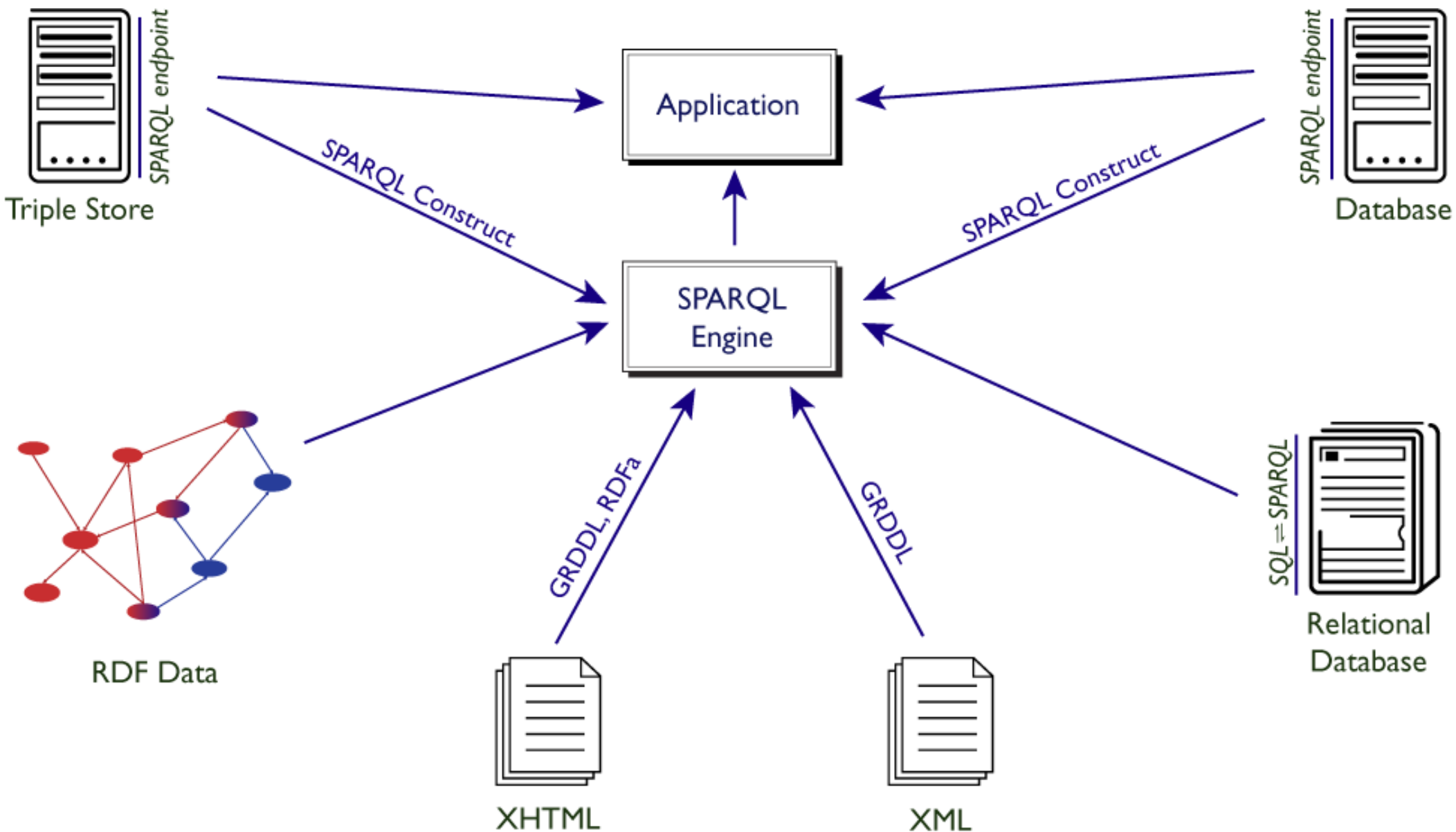
- SPARQL is usually used over the network
 - separate documents define the protocol and the result format
 - [SPARQL Protocol for RDF](#) with HTTP and SOAP bindings
 - SPARQL results in [XML](#) or [JSON](#) formats
- Big datasets usually offer “SPARQL endpoints” using this protocol
 - typical example: [SPARQL endpoint to DBpedia](#)

> A word of warning on SPARQL...



- Some features are missing
 - control and/or description on the entailment regimes of the triple store (RDFS? OWL-DL? OWL-Lite? ...)
 - modify the triple store
 - querying collections or containers may be complicated
 - no functions for sum, average, min, max, ...
 - ways of aggregating queries
 - ...
- Delayed for a next version...

> SPARQL as a unifying point



> Ontologies (OWL)





- RDFS is useful, but does not solve all possible requirements
- Complex applications may want more possibilities:
 - characterization of properties (not only listing their range and domain)
 - identification of objects with different URI-s
 - disjointness or equivalence of classes
 - construct classes, not only name them
 - more elaborate class hierarchies
 - 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”
 - restrict a property range when used for a specific class
 - etc.

> Ontologies (cont.)



- The term ontologies is used in this respect:

“defines the concepts and relationships used to describe and represent an area of knowledge”

- I.e., there is a need for Web Ontology Language(s)
 - RDFS can be considered as a simple ontology language
- Languages should be a compromise between
 - rich semantics for meaningful applications
 - feasibility, implementability

> Web Ontology Language = OWL



- OWL is an extra layer, a bit like RDF Schemas
 - own namespace, own terms
 - it relies on RDF Schemas
- It is a separate recommendation
- There is an active W3C Working Group working on extensions of the current standard
 - labeled as “OWL 2”
 - in what follows, some features will be referred to as “may come in future”, i.e., under consideration by that group

> OWL is complex...



- OWL is a large set of additional terms
- We will not cover the whole thing here...

> First some simple features



> Term equivalence



- For classes:
 - `owl:equivalentClass`: two classes have the same individuals
 - `owl:disjointWith`: no individuals in common
- For properties:
 - `owl:equivalentProperty`
 - remember the `a:author` vs. `f:auteur`?
- For individuals:
 - `owl:sameAs`: two URIs refer to the same concept (a.k.a. “individual”)
 - `owl:differentFrom`: negation of `owl:sameAs`

> Typical usage of owl:sameAs



- Linking our example of Kolkata from one data set (DBpedia) to the other (Geonames):

```
<http://dbpedia.org/resource/Kolkata>  
  owl:sameAs <http://sws.geonames.org/1275004/>;
```

- This is the main mechanism of “Linking” in the Linking Open Data project

> Other example: connecting to French



> Property characterization

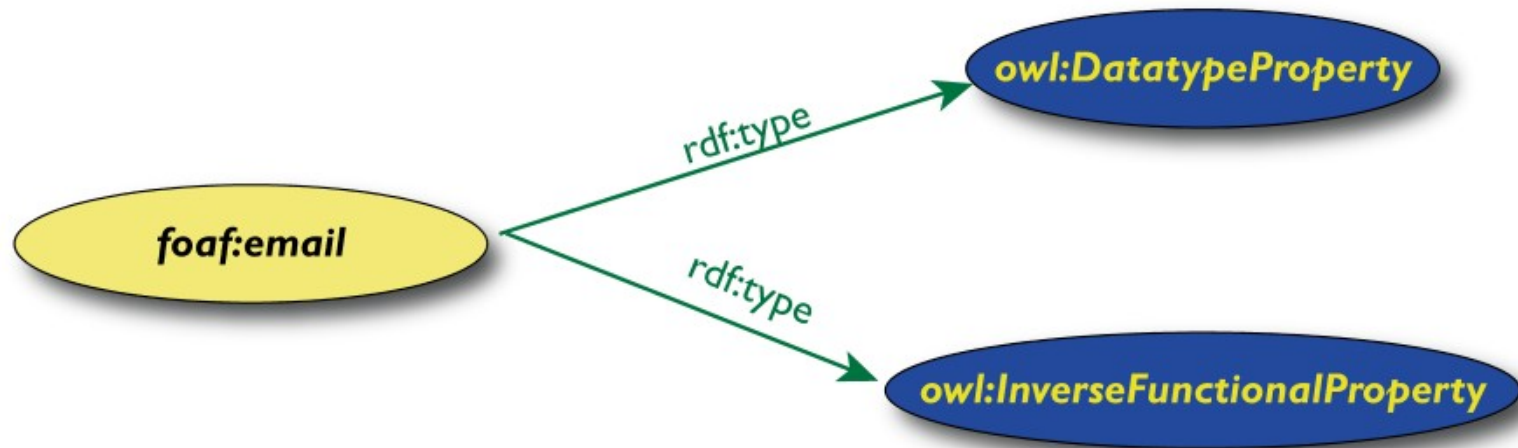


- In OWL, one can characterize the behavior of properties (symmetric, transitive, functional, inverse functional...)
- OWL also separates *data* and *object* properties
 - “datatype property” means that its range are literals

> Characterization example



- “**foaf:email**” is inverse functional (i.e., two different subjects cannot have identical objects)



> What this means is...



- If the following holds in our triples:

```
:email rdf:type owl:InverseFunctionalProperty.  
<A> :email "mailto:a@b.c".  
<B> :email "mailto:a@b.c".
```

- then the following holds, too:

```
<A> owl:sameAs <B>.
```

- I.e., new relationships were discovered again (beyond what RDFS could do)

> Other property characterizations



- Functional property (`owl:FunctionalProperty`)
- Transitive property (`owl:TransitiveProperty`)
- Symmetric property (`owl:SymmetricProperty`)
- Inverse of another property (`owl:inverseOf`)
- May come in future:
 - reflexive and irreflexive object properties
 - specify that properties are “disjoint”
 - combining (“chaining”) properties (a generalization of transitivity)

> Classes in OWL

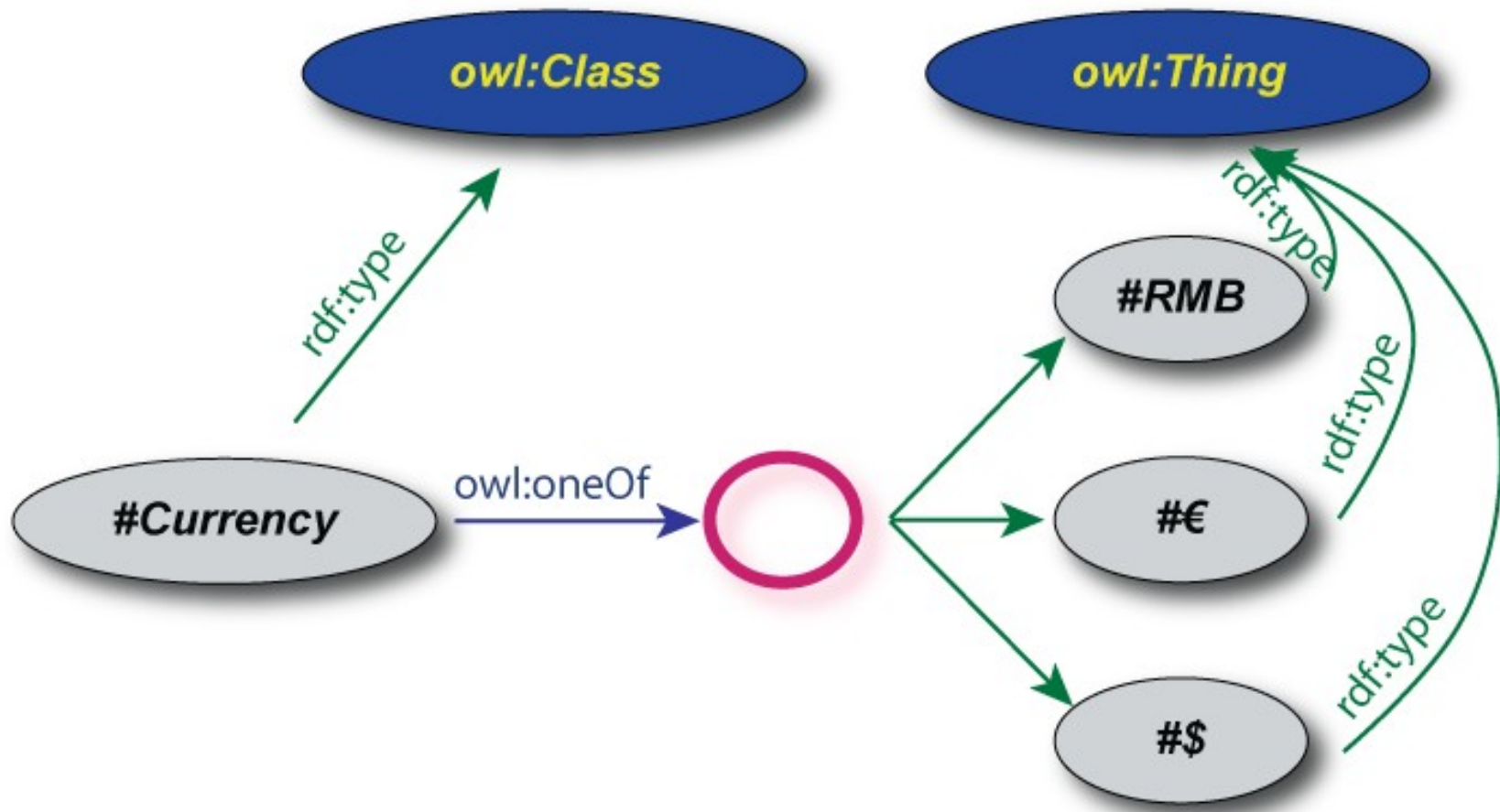


- In RDFS, you can subclass existing classes... that's all
- In OWL, you can construct classes from existing ones:
 - enumerate its content
 - through intersection, union, complement
 - etc
- OWL makes a stronger distinction between classes and individuals
 - referring to its own **Class** and to “**Thing**”, respectively
 - of course, `owl:Class` is a subclass of `rdfs:Class`, i.e., it is a refinement

> OWL classes can be “enumerated”



- The OWL solution, where possible content is explicitly listed:

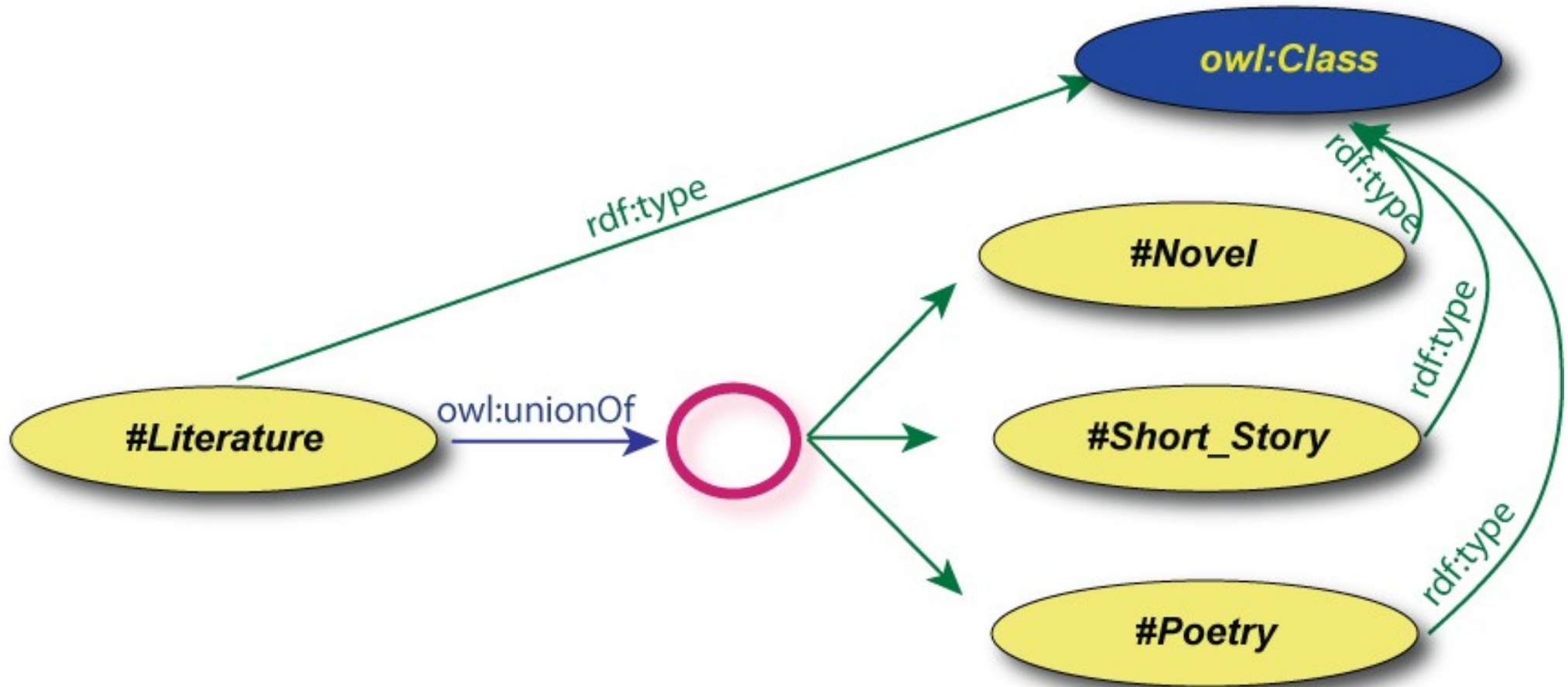


(don't worry about the syntax mapping...)

> Union of classes



- Essentially, like a set-theoretical union:



- Other possibilities include: intersection, complement of

> What we have so far...



- The OWL features listed so far are already fairly powerful
- E.g., various databases can be linked via **owl:sameAs**, functional or inverse functional properties, etc.
- It is still possible to find all inferred relationship using a traditional rule engine
 - there are some restrictions on subclassing that one has to follow, though

> However... that may not be enough



- Very large vocabularies might require even more complex features
 - typical example: definition of all concepts in a health care environment
- One major issue is the way classes (i.e., “concepts”) are defined
- OWL includes those extra features but... the inference engines become (much) more complex 😞

> Property value restrictions



- Classes are created by restricting the property values
- For example: how would I characterize a “listed price”?
 - it is a price (which may be a general term), but one that is given in one of the “allowed” currencies (say, €, RMB, or \$)
 - more formally:
 - the value of “**p:currency**”, when applied to a resource on listed price, must take one of those values...
 - ...thereby defining the class of “listed price”

> Restrictions expressed in RDF



```
:Listed_Price rdf:type owl:Class;  
  rdfs:subClassOf [  
    rdf:type          owl:Restriction;  
    owl:onProperty  <http://...#currency>;  
    owl:allValuesFrom :Currency.  
  ].
```

- “**allValuesFrom**” could be replaced by “**someValuesFrom**” to express another type of restriction
 - e.g., I could have said: there should be a price given in at least one of those currencies
- or “**hasValue**”, when restricted to one specific value

> Similar concept: cardinality restriction



- In a property restriction, the issue was to restrict the possible values of a property
- In a cardinality restriction, the number of relations with that property is restricted
- Eg: “a book being on offer” could be characterized as having at least one price property (i.e., the price of the book has been established)

> Cardinality restriction



```
:Book_on_sale rdf:type owl:Class;  
  rdfs:subClassOf [  
    rdf:type          owl:Restriction;  
    owl:onProperty   <http://...#price>;  
    owl:minCardinality "1"^^xsd:integer.  
  ].
```

- could also be “`owl:cardinality`” or “`owl:maxCardinality`”

> But: OWL is hard!



- The combination of class constructions with various restrictions is extremely powerful
- What we have so far is following the same logic as before
 - extend the basic RDF and RDFS possibilities with new features
 - expect to infer new relationships based on those
- However... a full inference procedure is hard
 - not implementable with simple rule engines, for example
 - in some cases, it may even be impossible 😞

> OWL profiles



- The term OWL “profiles” comes to the fore:
 - restricting which terms can be used and under what circumstances (restrictions)
 - if one abides to those restrictions, then simpler inference engines can be used

> OWL profiles (cont.)



- In the *current* OWL standard, three such “profiles” are defined:
 - OWL Full: no restrictions whatsoever
 - OWL DL (and its “sub profile” OWL Lite): major restrictions to ensure implementability
- The current OWL 2 work will add new profiles
 - profiles that are simple enough to be implementable with simple rule engines (like the first few examples we had)
 - profiles that are optimized to a small number of class and property definition but a large amount of data
 - etc.



- No constraints on the various constructs
 - this means that:
 - Class can also be an individual, a URI can denote a property as well as a Class
 - e.g., it is possible to talk about class of classes, etc.
 - one could make statements on RDFS constructs (e.g., declare `rdf:type` to be functional...)
 - etc.
- But: *an OWL Full ontology may be undecidable!*



- A number of restrictions are defined
 - Classes, individuals, properties are strictly separated: a class *cannot* be an individual of another class
 - strict separation of the user's and the reserved (RDFS, OWL) terms
 - no statements on RDFS and OWL resources, for example
 - the values of user's object must be individuals
 - i.e., they are used to create relationships between individuals
 - no characterization of *datatype* properties (functional, etc)
 - ...
- But: well known inference algorithms exist!

> Note on OWL profiles



- OWL profiles are defined to reflect compromises:
 - expressibility vs. implementability
- Some application just need to express and interchange terms (with possible scruffiness): OWL Full is fine
 - they may build application-specific reasoning instead of using a general one
- Some applications need rigor, but only a simple set of statements: a rule engine based profile from OWL 2 might be o.k.
- Some applications need rigor and complex term classification; then OWL DL might be the good choice

> Ontology development



- The hard work is to create the ontologies
 - requires a good knowledge of the area to be described
 - some communities have good expertise already (e.g., librarians)
 - OWL is just a tool to formalize ontologies
- Large scale ontologies are often developed in a community process
- Ontologies should be shared and reused
 - can be via the simple namespace mechanisms...
 - ...or via explicit inclusions
- Applications can also be developed with very small ontologies, though

> Ontologies examples



- International Country List
 - example for an OWL Lite ontology
- Large ontologies are being developed (converted from other formats or defined in OWL)
 - [eClassOwl](#): eBusiness ontology for products and services, 75,000 classes and 5,500 properties
 - [National Cancer Institute's ontology](#): about 58,000 classes
 - [Open Biomedical Ontologies Foundry](#): a collection of ontologies, including the [Gene Ontology](#) to describe gene and gene product attributes in any organism or protein sequence and annotation terminology and data ([UniProt](#))
 - [BioPAX](#): for biological pathway data

> What have we achieved?

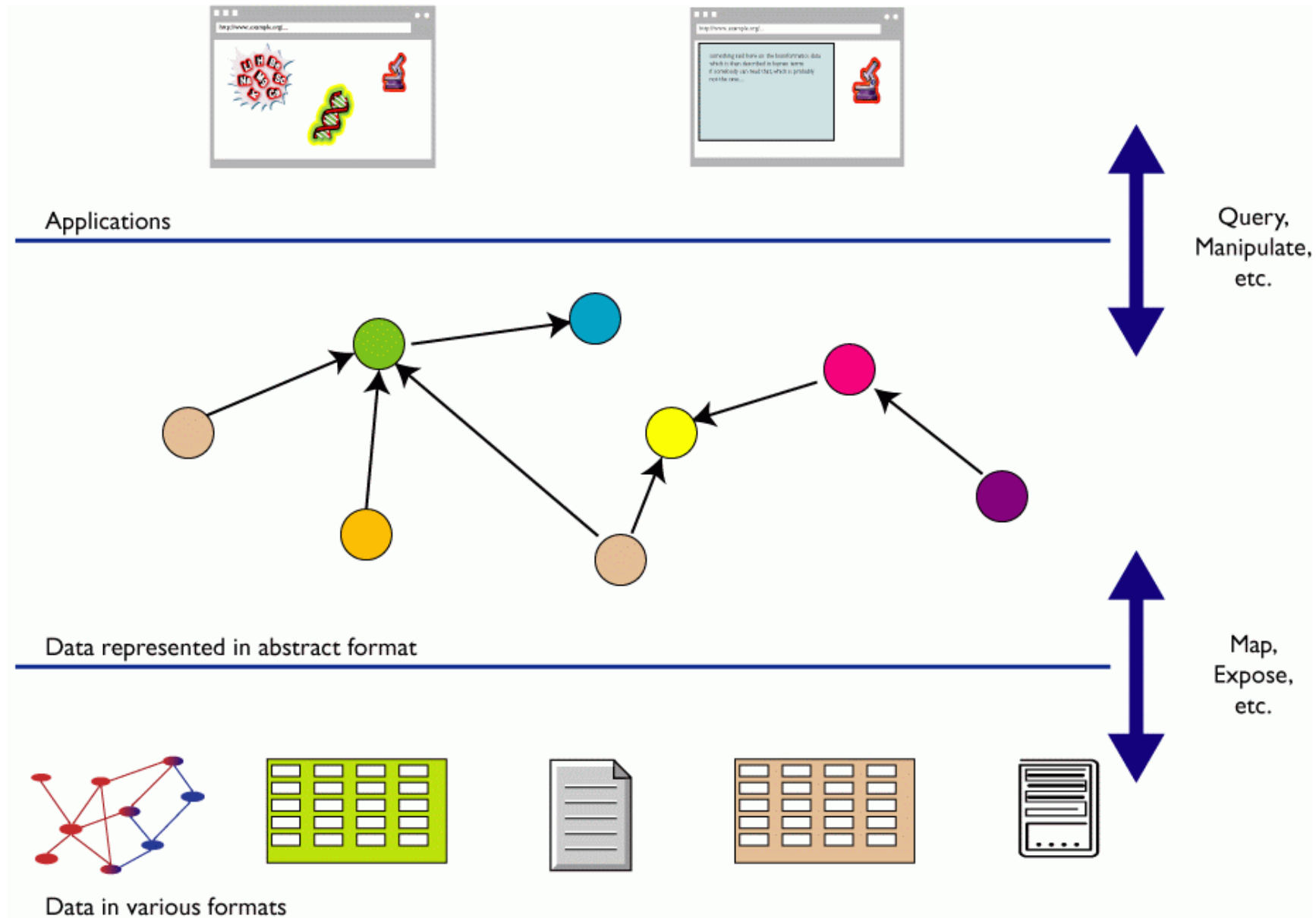


> We have not talked about everything...

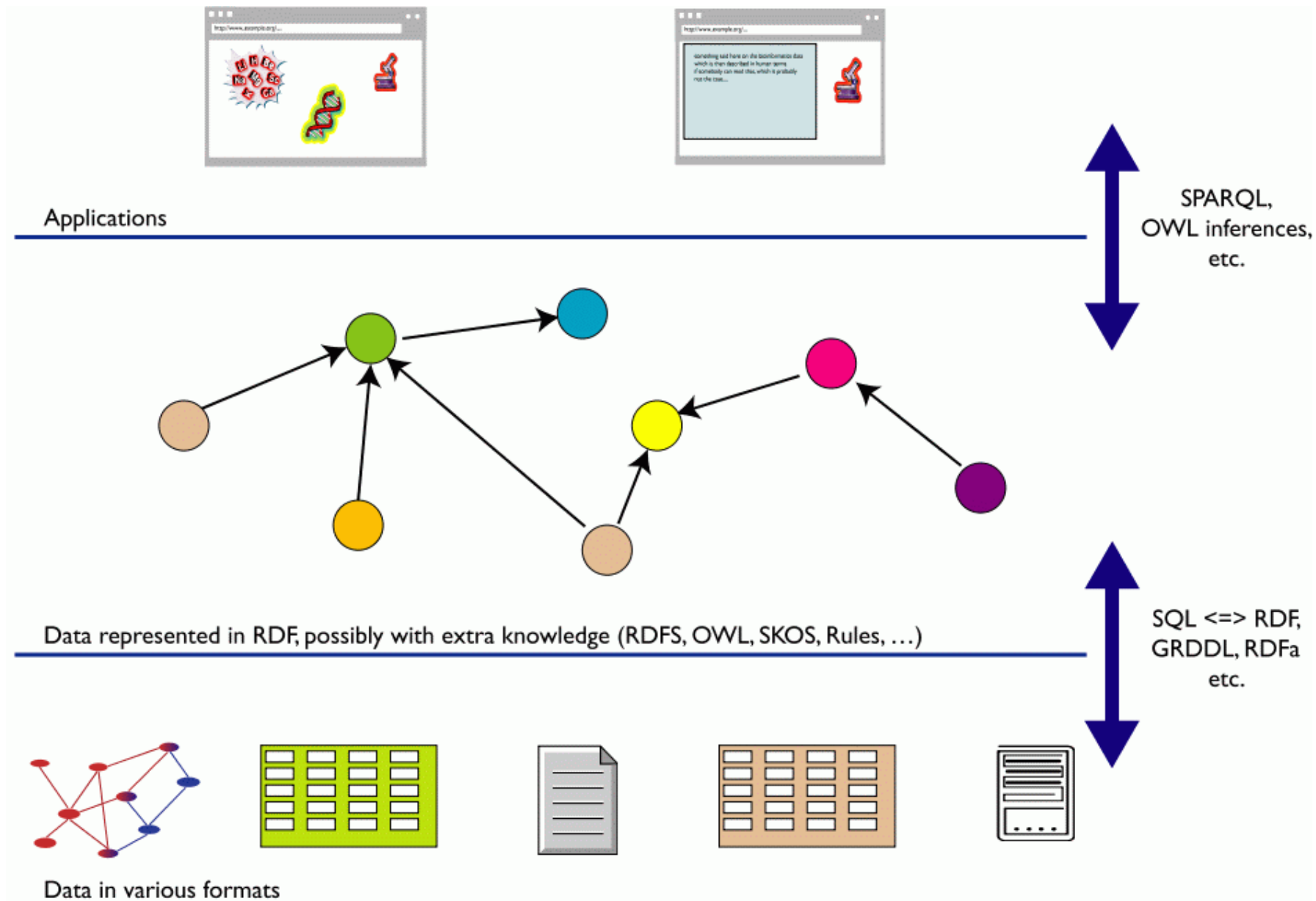


- Some other aspects of SW are being developed
 - some at W3C, others are still research
- For example:
 - RIF: using general rule engines with SW data; also *interchange* rule descriptions (just like data are interchanged)
 - SKOS: general framework to express term structures like vocabularies, taxonomies, glossaries
 - eg, to interface bibliographic records

> Remember the integration example?



> Same with what we learned



> Available documents, tools



> Available specifications: Primers, Guides



- The “[RDF Primer](#)” and the “[OWL Guide](#)” give a formal introduction to RDF(S) and OWL
- [GRDDL Primer](#) and [RDFa Primer](#) have been published
- The [W3C Semantic Web Activity Homepage](#) has links to all the specifications

> “Core” vocabularies



- There are also a number “core vocabularies” (not necessarily OWL based)
 - **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
 - **Music Ontology**: on the description of CDs, music tracks, ...
 - **SIOC**: Semantically-Interlinked Online Communities
 - **vCard in RDF**
 - ...
- One should never forget: ontologies/vocabularies must be shared and reused!

> Some books



- J. Davies, D. Fensel, F. van Harmelen: Towards the Semantic Web (2002)
- S. Powers: Practical RDF (2003)
- F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P. Patel-Schneider: The Description Logic Handbook (2003)
- G. Antoniu, F. van Harmelen: Semantic Web Primer (2004)
- A. Gómez-Pérez, M. Fernández-López, O. Corcho: Ontological Engineering (2004)
- ...

See the separate [Wiki page](#) collecting book references

> Further information



- [Dave Beckett's Resources](#) at Bristol University
 - *huge* list of documents, publications, tools, ...
- [Planet RDF](#) aggregates a number of SW blogs
- [Semantic Web Interest Group](#)
 - a forum developers with archived (and public) mailing list, and a constant IRC presence on `freenode.net#swig`
 - anybody can sign up on the list

> Some SW Tools (not an exhaustive list!)



• Triple Stores

- RDFStore, AllegroGraph, Tucana
- RDF Gateway, Mulgara, SPASQL
- Jena's SDB, D2R Server, SOR
- Virtuoso, Oracle11g
- Sesame, OWLIM, Tallis Platform
- ...

• Reasoners

- Pellet, RacerPro, KAON2, FaCT++
- Ontobroker, Ontotext
- SHER, Oracle 11g, AllegroGraph
- ...

• Converters

- flickurl, TopBraid Composer
- GRDDL, Triplr, jpeg2rdf
- ...

• Search Engines

- Falcon, Sindice, Swoogle
- ...

• Middleware

- IODT, Open Anzo, DartGrid
- Ontology Works, Ontoprise
- Profium' SIR, Software AG's EII
- Thetus Publisher, Asio, SDS
- ...

• Semantic Web Browsers

- Disco, Tabulator, Zitgist, OpenLink Viewer
- ...

• Development Tools

- SemanticWorks, Protégé
- Jena, Redland, RDFLib, RAP
- Sesame, SWI-Prolog
- TopBraid Composer, DOME
- ...

• Semantic Wiki and CMS systems

- Semantic Media Wiki, Platypus
- Visual knowledge, Drupal 7

Inspired by "Enterprise Semantic Web in Practice", Jeff Pollock, Oracle. See also [W3C's Wiki Site](#).

> So how do applications look like?



> Application patterns



- It is fairly difficult to “categorize” applications (there are always overlaps)
- With this caveat, some of the application patterns:
 - data integration (ie, integrating data from major databases)
 - intelligent (specialized) portals (with improved local search based on vocabularies and ontologies)
 - content and knowledge organization
 - knowledge representation, decision support
 - X2X integration (often combined with Web Services)
 - data registries, repositories
 - collaboration tools (eg, social network applications)

> Applications do not have to be complicated



- Goal: reuse of older experimental data
- Keep data in databases or XML, just export key “fact” as RDF
- Use a faceted browser to visualize and interact with the result

Internal Compound Repurposing Example

Welcome, Allergy & Respiratory Team Member

This tool allows you to identify opportunities for additional uses of compounds from other teams within your project. It combines internal data, public data and the results of data mining experiments to provide testable hypotheses.

Control Panel & Item Filtering

Area	5/2	Approach	3/2	Term+Reason	1/2	Max_Stage_Reached	1/2	Literature Links
29 Pain	<input checked="" type="checkbox"/>	7 Antibody	<input type="checkbox"/>	37 ACTIVE	<input type="checkbox"/>	51 Candidate	<input checked="" type="checkbox"/>	0 - 50
16 Metabolic Disease	<input checked="" type="checkbox"/>	1 Recombinant	<input type="checkbox"/>	12 BIOMARKER	<input type="checkbox"/>	10 Discovery	<input type="checkbox"/>	
8 Cancer	<input type="checkbox"/>	18 SM_Agonist	<input checked="" type="checkbox"/>	51 EFFICACY	<input checked="" type="checkbox"/>	41 Exploratory	<input type="checkbox"/>	

Internal Compound Repurposing Example

Welcome, Allergy & Respiratory Team Member

This tool allows you to identify opportunities for additional uses of compounds from other teams within your project. It combines internal data, public data and the results of data mining experiments to provide testable hypotheses.

Control Panel & Item Filtering

Area	5/2	Approach	3/2	Term+Reason	1/2	Max_Stage_Reached	1/2	Literature Links
29 Pain	<input checked="" type="checkbox"/>	7 Antibody	<input type="checkbox"/>	37 ACTIVE	<input type="checkbox"/>	51 Candidate	<input checked="" type="checkbox"/>	0 - 50
16 Metabolic Disease	<input checked="" type="checkbox"/>	1 Recombinant	<input type="checkbox"/>	12 BIOMARKER	<input type="checkbox"/>	10 Discovery	<input type="checkbox"/>	
8 Cancer	<input type="checkbox"/>	18 SM_Agonist	<input checked="" type="checkbox"/>	51 EFFICACY	<input checked="" type="checkbox"/>	41 Exploratory	<input type="checkbox"/>	
3 Sexual Health	<input checked="" type="checkbox"/>	12 SM_Antagonist	<input checked="" type="checkbox"/>	11 MARKET	<input type="checkbox"/>	19 HTS	<input type="checkbox"/>	
2 Infectives	<input checked="" type="checkbox"/>	21 SM_Inhibitor	<input checked="" type="checkbox"/>	11 REORG	<input type="checkbox"/>	11 Phase I	<input type="checkbox"/>	
1 Urogenitals	<input checked="" type="checkbox"/>			10 TOXIC	<input type="checkbox"/>	13 Phase III	<input type="checkbox"/>	
						41 Screening	<input type="checkbox"/>	

51 items filtered from 710 originally (Reset All Filters)

Area	Original+Indication	Target_Name	Approach	Start	Term+Reason	Max_Stage_Reached	Owner	OMIM	UT_All Lit	2007 UT_Neoch	IMA	GED	Pathway	Compounds
Metabolic Disease	Diabetes	Liver glycogen phosphorylase	SM_Inhibitor	2007-Q2	EFFICACY	Candidate	P. Person							SW-030072
Sexual Health	Erectile Dysfunction	Integrin alpha-3 (Galactoprotein B3)(VLA-3) (CD49C)	SM_Antagonist	2006-Q3	EFFICACY	Candidate	P. Person					1		SW-029782
Sexual Health	Erectile Dysfunction	Leukotriene C4 synthase	SM_Agonist	2006-Q3	EFFICACY	Candidate	M. Manager				1	1		SW-029638
Sexual Health	Erectile Dysfunction	transcription elongation factor A (SII)-like 4	SM_Inhibitor	2005-Q2	EFFICACY	Candidate	P. Person							SW-029926
Infectives	HIV	Putative four repeat ion channel (Jx)	SM_Inhibitor	2006-Q2	EFFICACY	Candidate	L. Leader							SW-029994
Infectives	HIV	Voltage-gated potassium channel protein Kv1.2 (Kv)	SM_Agonist	2007-Q1	EFFICACY	Candidate	A. Scientist						1	SW-029653
Urogenitals	Incontinence	Human Rfx binding motif (RBM) gene, partial cds.	SM_Agonist	2007-Q3	EFFICACY	Candidate	L. Leader						1	SW-029684
Pain	Migraine	Monocarboxylate transporter homologue 2 (SLC6A4) (SLC6A4)	SM_Inhibitor	2007-Q3	EFFICACY	Candidate	L. Leader		18					SW-030085

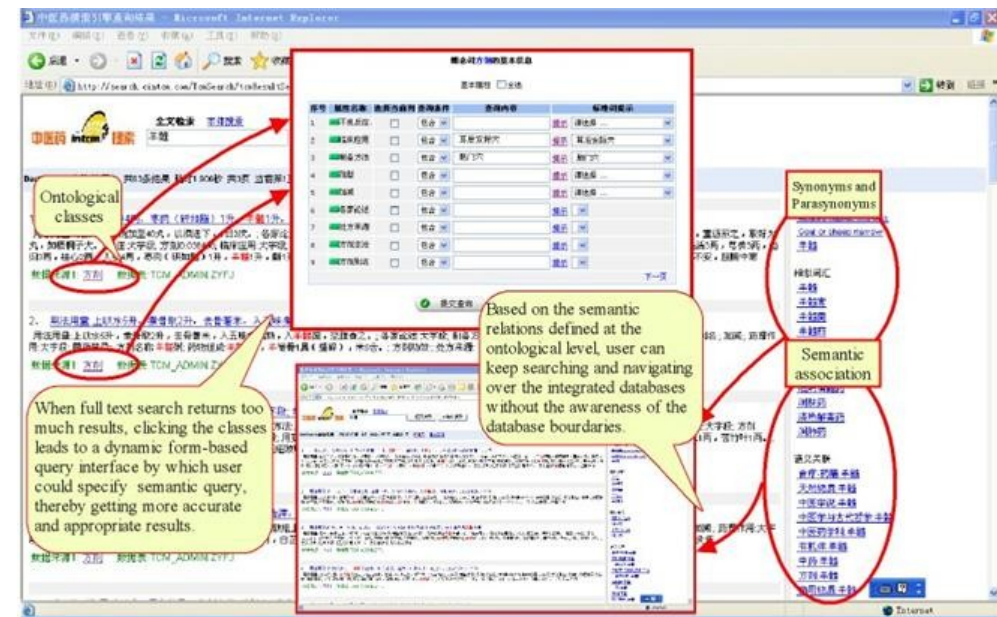
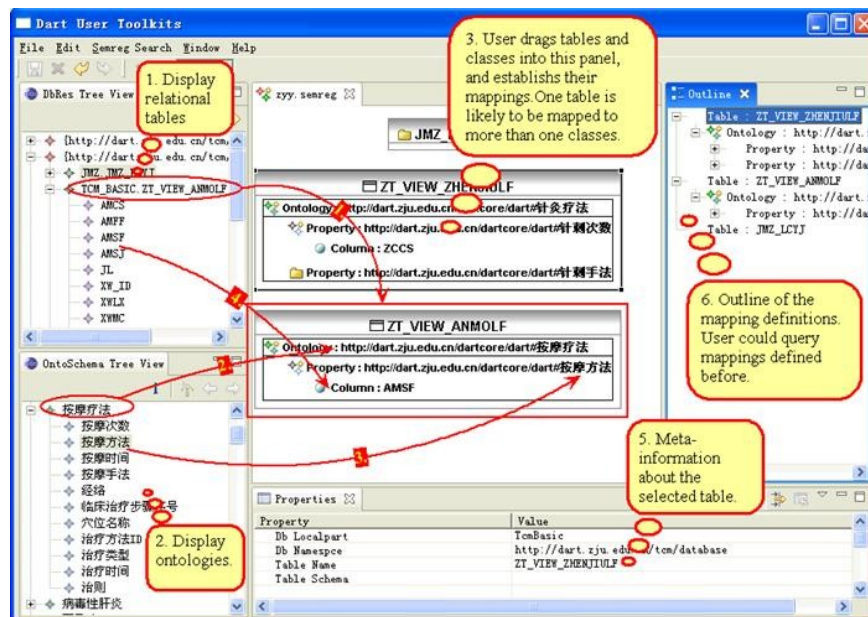
compounds
SW-030072
SW-029782
SW-029638
SW-029926
SW-029994
SW-029653
SW-029684
SW-030085

Courtesy of Nigel Wilkinson, Lee Harland, Pfizer Ltd, Melliya Annamalai, Oracle (SWEO Case Study)

> Integrate knowledge for Chinese Medicine



- Integration of a large number of relational databases (on traditional Chinese medicine) using a Semantic Layer
 - around 80 databases, around 200,000 records each
- A visual tool to map databases to the semantic layer using a specialized ontology
- Form based query interface for end users



Courtesy of Huajun Chen, Zhejiang University, (SWEQ Case Study)

> Find the right experts at NASA



- Expertise locator for nearly 70,000 NASA civil servants using RDF integration techniques over 6 or 7 geographically distributed databases, data sources, and web services...

POPS v.28.3 - Connected to 'POPS on FatDuck' - Using Model 'POPS on FatDuck Model' - Logged in as 'Michael Grove'

File Options Bookmarks Advanced Help

NASA Center (15) Project (176) Competency (21) People (1)

ARC
DFRC
GRC
GSFC
HQ
IVV
JPL
JSC
KSC
LARC
MAF
MSFC

Source: x500

Mars Global Surveyor
Mars Odyssey 2001
Mars R&A
Mars Reconnaissance Orbiter 2005 (...
Messenger
Minor Revital
Mission Operations
Mission Science Guest Investigator
Mission Success - Center Specific
Multi-Mission Operations
NMP Program Management and Futur...
NPOESS Preparatory Project (NPP)

Source: WIMS

Astrobiology
Astronomy and Astrophysics
Climate Change and Variability
Earth Atmosphere
Earth Science Applications Research
Earth System Modeling
Fluid Physics
Fundamental Physics
Geophysical/Geologic Science
Geospatial Science and Technologies
Icing Physics
Laser Technology

Source: CMS

Jeanne M.

Source:

Information Panel

View Different Social Network's Present in the Data

POPS

Jeanne M.
Skill: Earth Sciences Competency Suite
Project: Center Investment Accounts

Facility: HQ

Michael H Grove

Name: Michael Grove
Email: @nasa.gov
Phone: 301.
Employer: Clark and Parsia

Legend:

- Same Skill and Same Department
- Same Skill and Same Project
- Same Skill, Project, and Facility
- Am I Connected?

1 of 1

Social Net

Michael Grove, Clark & Parsia, LLC, and Andrew Schain, NASA, (SWEO Case Study)

> Help in choosing the right drug regimen



- Help in finding the best drug regimen for a specific case
 - find the best trade-off for a patient
- Integrate data from various sources (patients, physicians, Pharma, researchers, ontologies, etc)
- Data (eg, regulation, drugs) change often, but the tool is much more resistant against change

The screenshot shows the PharmaSURVEYOR interface for 'Survey 1 Version 2'. It displays a table comparing the 'Current Regimen' with two 'Safety Optimized Profiles' (1 and 2). The table lists Adverse Drug Effects (ADEs) with their severity and whether they are present in each regimen. The 'Current Regimen' has ADEs in all six rows, while the optimized profiles have ADEs in only three rows each.

Severity	Adverse Drug Effect	Current Regimen	Safety Optimized Profile 1	Safety Optimized Profile 2
ADE Moderate	Muscle Weakness (Myasthenia)	Yes	Yes	Yes
ADE Minor	Excessive Sweating (Diaphoresis)	Yes	Yes	Yes
ADE Moderate	Heart Throbbing or Pounding (Palpitations)	Yes	Yes	Yes
ADE Moderate	Hives (Urticaria)	Yes	Yes	Yes
ADE Major	Bladder Inflammation (Cystitis)	Yes	Yes	Yes
ADE Major	Urinary Tract Infection	Yes	Yes	Yes

row(s) 1 - 6 of 6

[Export to Excel](#)

Courtesy of Erick Von Schweber, PharmaSURVEYOR Inc., (SWEQ Use Case)

> Help for deep sea drilling operations



- Integration of experience and data in the planning and operation of deep sea drilling processes
- Discover relevant experiences that could affect current or planned drilling operations
 - uses an ontology backed search engine

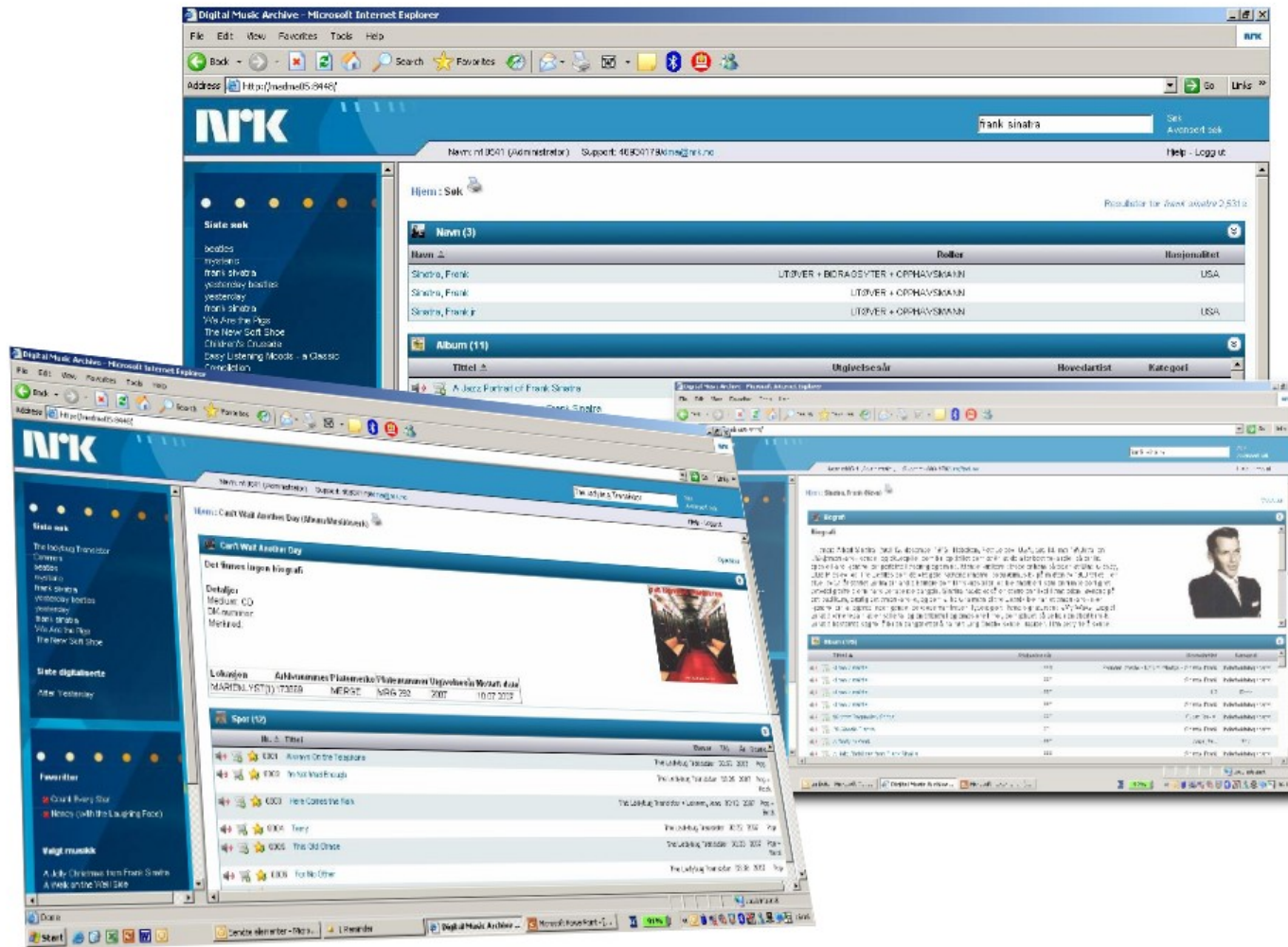
The screenshot displays the AKSIO search interface. At the top, there is a search bar with the text "leak in barrier elements" and a button labeled "AKSIO-search". Below the search bar, there are several search filters on the left side, including "discipline", "operation", "equipment", "state", "keywords_ref", "wellbore_id_ref", and "field_id". The "state" filter is expanded, showing a list of states with checkboxes: Corrosion (1), Erosion (2), Lack Of Maintenance (2), Leak in barrier elements (5), Scale Deposition (4), Too High Mud Density (1), and Well Integrity Problem (7). The "field_id" filter is also expanded, showing a list of field IDs with checkboxes: EXPLORATION (1), GULLFAKS (1), GULLFAKS SØR (1), HEIDRUN (1), HULDRA (1), MIDGARD (2), RIMFAKS (1), SNORRE (1), and VISUND (2). On the right side, there are search results labeled "Results 1 - 7 of 7". The first result is titled "Top plug/20\" EZSV" and has a 55% annotation score. The second result is titled "RIH with drill stem teststring" and has a 50% annotation score. The third result is titled "Flowing well" and has a 47% annotation score. The fourth result is titled "Fill drop sub assy prior to making up packer for barrier assy to avoid possible trapped pressure" and has a 39% annotation score. Each result includes a description, a date, and a link to "annotate" or "comment".

Courtesy of David Norheim and Roar Fjellheim, Computas AS (SWEO Use Case)

> Digital music asset portal at NRK



- Used by program production to find the right music in the archive for a specific show

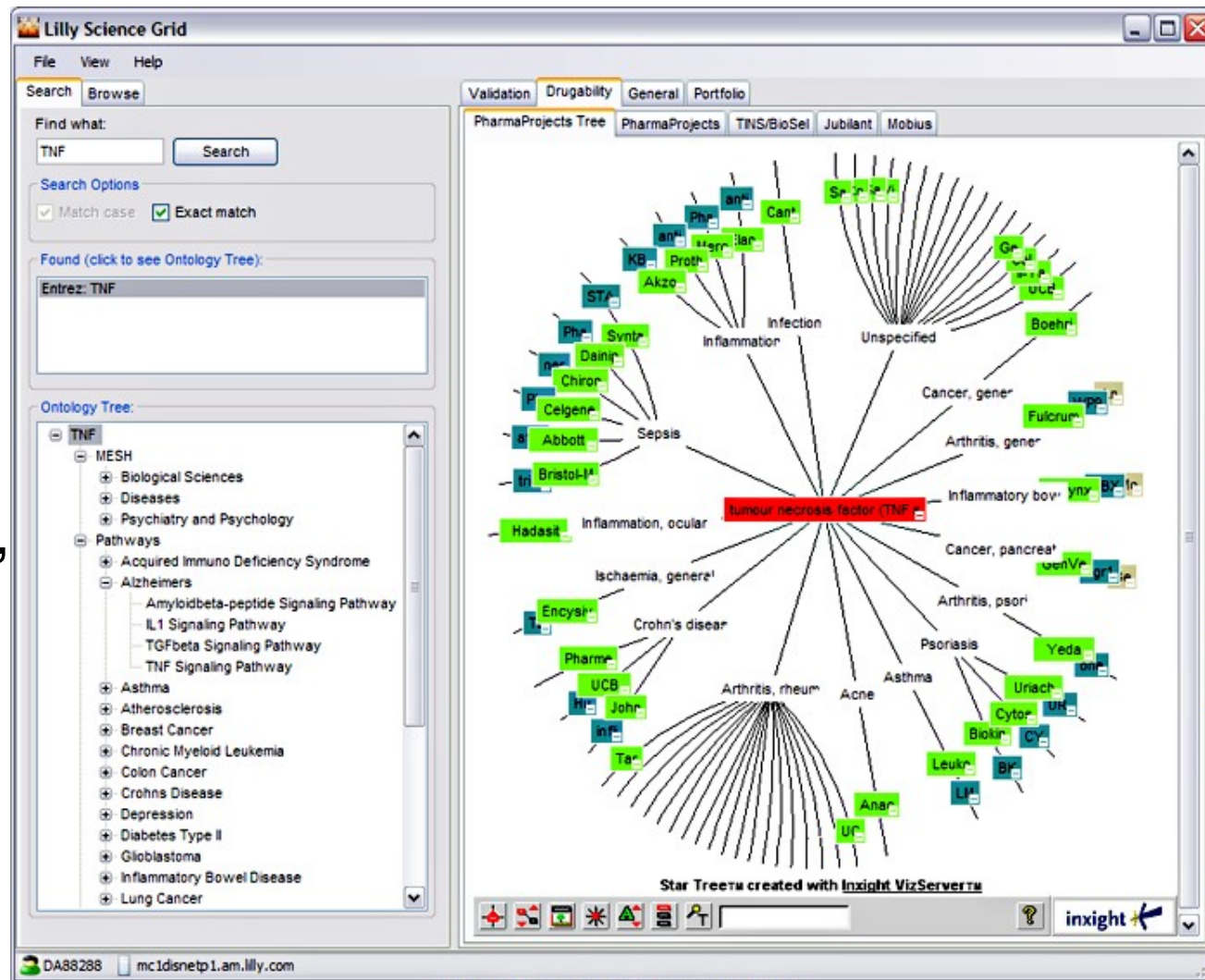


Courtesy of Robert Engels, ESIS, and Jon Roar Tønnesen, NRK (SWEO Case Study)

> Eli Lilly's Target Assessment Tool



- Better prioritization of possible drug target, integrating data from different sources and formats
- Integration, search, etc, via ontologies (proprietary and public)



Courtesy of Susie Stephens, Eli Lilly (SWEQ Case Study)



- Integrate various vendors' product descriptions via RDF
 - ring tones, games, wallpapers
 - manage complexity of handsets, binary formats
- A portal is created to offer appropriate content
- Significant increase in content download after the introduction



> Improved Search via Ontology (GoPubMed)



- Search results are re-ranked using ontologies
- Related terms are highlighted, usable for further search

The screenshot displays the GoPubMed web interface within a Mozilla Firefox browser window. The left sidebar, titled 'what', contains a hierarchical ontology of categories. The 'Diseases' category is expanded, and 'Tinnitus' is highlighted with a red circle. The main content area shows search results for the query 'tinnitus', which has returned 1,000 articles. The top result is '5: Pros and cons of tinnitus retraining therapy.' by Hatanaka A et al., with a PMID of 18368566. A blue arrow points to the word 'tinnitus' in the title, which is highlighted in yellow. Below this, other results are listed, including '1: Gabapentin effectiveness on the sensation of subjective idiopathic tinnitus : a pilot study.' and '3: Algorithm for evaluation of pulsatile tinnitus.' The interface also includes a 'find it!' button and a 'goPubMed' logo.

> Improved Search via Ontology (Go3R)



- Same dataset, different ontology
 - (ontology is on non-animal experimentation)

The screenshot displays the Go3R web application interface within a Mozilla Firefox browser window. The interface is divided into two main sections: a left sidebar for navigation and a main content area for search results.

Left Sidebar (Navigation):

- what**
- 3R Relevance Filters (Beta)**
- Top categories**
- Diseases & Symptoms [601]**
 - Tinnitus [547]
 - Hearing Loss [248]
 - Vertigo [98]
 - Disease [118]
 - Hearing Loss, Sensorineural [95]
 - more
- Methodology [408]**
- Life Sciences [503]**
- Body Systems & Structures [401]**
- Bioethics [102]**
- Reduction [90]**
- more
- Statistics [125]**
- Substances, Preparations & Products [277]**
- Biological Material & Organisms for Animal U**
- Method Specification [36]**
- Animal Species [40]**
- Product Properties & Effects [62]**
- Product Testing & Assessment [20]**
- 3Rs Methods in the Life Sciences [6]**
- Animal Experiment [6]**
- 3Rs Relevant [5]**
- In Vitro Experimental Design [20]**
- In Vivo Experimental Design [5]**
- Animal Condition, Physiological or Psycholog**
- Animal Care & Handling [3]**
- Toxic Actions of Substances [7]**
- Unclassified [390]**
- Find related categories ...**
- My last 5 queries**
- Clipboard [0]**

Main Content Area (Search Results):

- Search bar: **find it!**
- 1,000 articles**
- ifferences (P > 0.05).
- 2: Microvascular decompression of cochleovestibular nerve.** PMID: 18389269 Related Articles
Yap L et al., Eur Arch Otorhinolaryngol, 2008
This report provides a review of all the published studies on MVD of the eighth (8th) nerve in alleviating cochleovestibular symptoms and presents three additional patients who underwent MVD of the eighth nerve for tinnitus or vertigo.
- 3: Algorithm for evaluation of pulsatile tinnitus.** PMID: 18368578 Related Articles
Mattox DE et al., Acta Otolaryngol, 128 (4): 427-31, 2008
Among patients with arterial tinnitus, carotid atherosclerotic disease was the most common.
- 4: Functional imaging of unilateral tinnitus using fMRI.** PMID: 18368576 Related Articles
Lanting CP et al., Acta Otolaryngol, 128 (4): 415-21, 2008
This article shows that the inferior colliculus plays a key role in unilateral subjective tinnitus.
- 5: Pros and cons of tinnitus retraining therapy.** PMID: 18368566 Related Articles
Hatanaka A et al., Acta Otolaryngol, 128 (4): 365-8, 2008
A significant reduction in the Tinnitus Handicap Inventory (THI) was obtained as early as 1 month after implementation of tinnitus retraining therapy (TRT).
- 6: Mass casualty incident management, triage, injury distribution of casualties and**

> Radar Network's Twine



- “Social bookmarking on steroids”

- Item relationships are based on ontologies

- evolving over time
- possibly enriched by users

- Internals in RDF, will be available via APIs and SPARQL

The screenshot shows the Twine web interface in a Mozilla Firefox browser window. The browser title is "The Manhattan Project: The Birth of the Atomic Bomb in the Words of Its Creators, Eyewitnesses and Historians. © twine - Mozilla Firefox". The Twine interface has a navigation bar with links: Home, My Twines, My Connections, Explore, Start a Twine, and a search bar. The main content area is titled "Ivan's private twine" and is marked as "Private". It has tabs for Summary, Items, Members, and Manage, along with an "Add Item" button. Below the tabs are buttons for "Share and Collect", "Comment", and "Add Detail". The main content displays a book titled "The Manhattan Project: The Birth of the Atomic Bomb in the Words of Its Creators, Eyewitnesses and Historians." with a book cover image. Below the title, there is a table of metadata:

Original URL	http://www.amazon.com/Manhattan-Project-Creators-Eyewitnesses-Historians/d
Publisher	Black Dog & Leventhal Publishers
Release Date	Sep 17, 2007 (3 months ago)
Price	\$24.95

The right sidebar shows a list of tags categorized into:

- Places: Asia, Europe, United States
- People: Albert Einstein, Groves, Harry S. Truman, J. Robert Oppenheimer, Klaus Fuchs, Leo Szilard, Richard Feynman
- Organizations: Atomic Heritage Foundation, Manhattan Project
- Other tags: Atomic bomb

> Yahoo's microsearch



- Make use of RDF, RDFa, microformats, etc, in pages
 - E.g., geo location, or various spellings of a name are discovered:

The screenshot shows a Mozilla Firefox browser window with the address bar displaying `http://www.yr-bcn.es/demos/microsearch/search.do?p=ivan+herman&y=Search&fr=lo`. The page title is "ivan herman - Yahoo! Search Results - Mozilla Firefox". The search results are displayed under the "YAHOO! RESEARCH micro SEARCH" logo. The search term "ivan herman" is entered in the search box. The results show a list of links, a map of Amsterdam, and a timeline of events.

Search Results Persons: 10 vCards: 1020 Events: 448 Unfinished: 1 | 1 - 10 of about 284029 for ivan herman - 13 sec. ([About this page](#))

- Ivan's blog**
<http://www.ivan-herman.net/> - 45k Update metadata
- Ivan Herman**

name: Ivan Herman
personal mailbox: <mailto:ivan@w3.org>
homepage: <http://www.ivan-herman.net>
Ivan Herman gives a talk on behalf of the China Office entitled "What is the ... Ben Adida, Elias Torres, and **Ivan Herman** give a tutorial entitled "RDFa: ...
<http://www.w3.org/People/Ivan/> - 23k Update metadata
homepage
[Ivan's blog](#)
<http://www.ivan-herman.net/> - 45k Update metadata
- Ivan Herman's Photos**

name: Herman Iván, Ivan Herman, Herman Iván, Ivan Herman
homepage: <http://www.ivan-herman.net>, <http://www.ivan-herman.net>
My Photos. I like traveling and making photos. ... **Iván Herman**, 2008-02-28, ivan@ivan-herman.net The images are licensed under a ...
<http://www.ivan-herman.net/Photos/> - 4k Update metadata
homepage
[Ivan's blog](#)
<http://www.ivan-herman.net/> - 45k Update metadata
holds account
[Ivan Herman | Facebook](#)

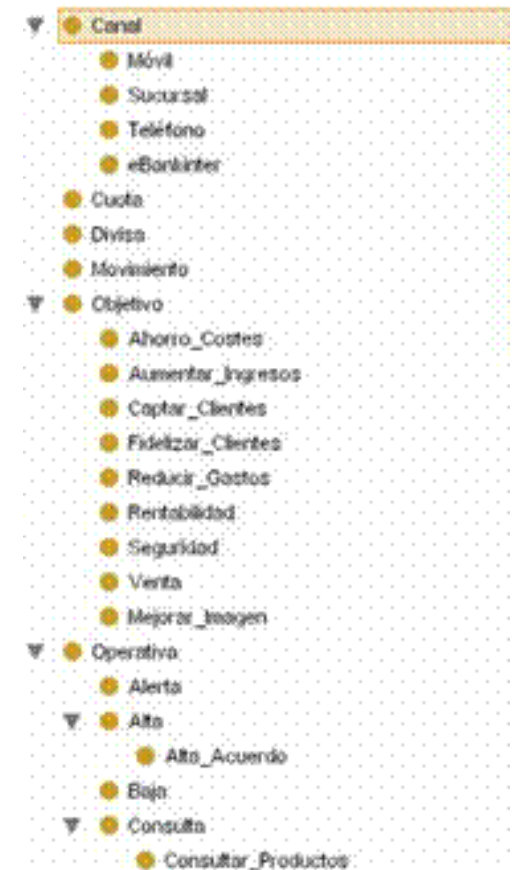
The map on the right shows a satellite view of Amsterdam with two location markers: "Ivan Herman" and "Herman Iván / Ivan Herman". The timeline below the map shows a bar chart of activity from April 2006 to August 2007, with a peak in May 2006.

> Suggestions' database...



bankinter.

- Employees of the bank can submit new ideas for innovation, improving the business process, reduce costs, etc
- The entry system analyses the entry, shows similar ideas already in the system based on the concepts (not words)
- User gets immediate feedback, system gets better search, analysis, etc

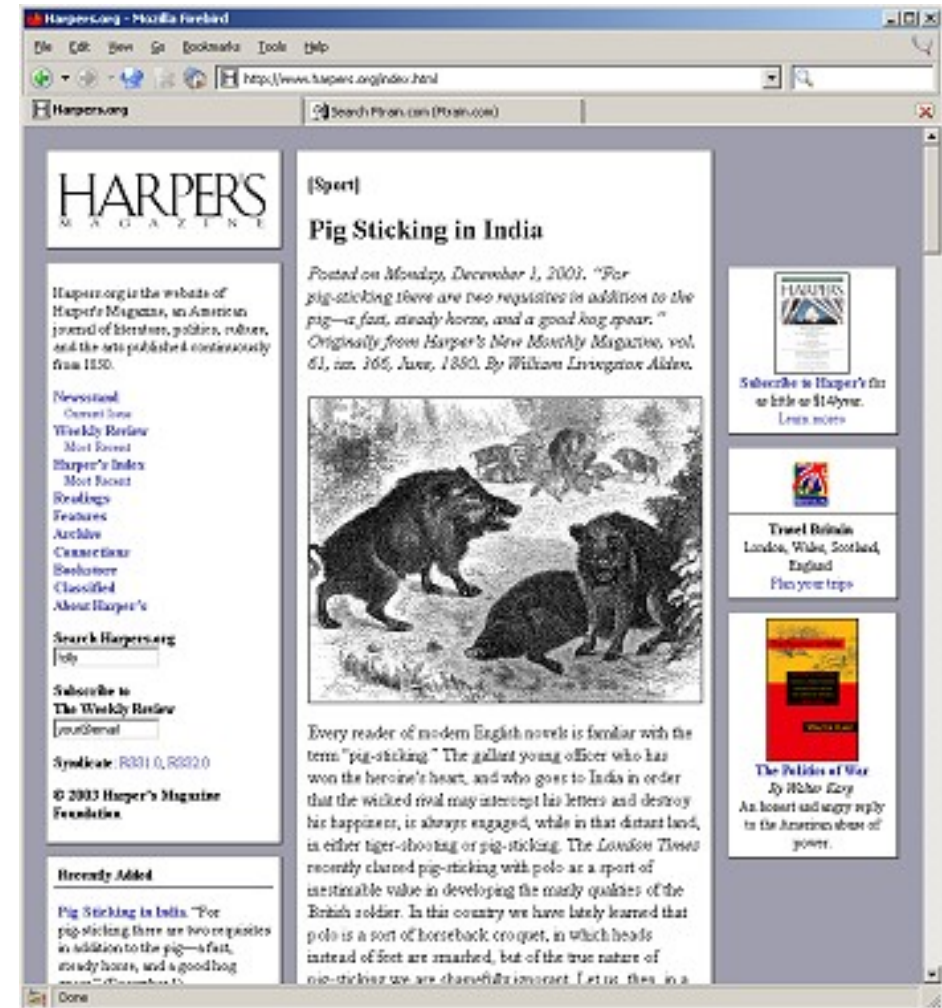


Courtesy of José Luís Bas Uribe, Bankinter, and Richard Benjamins, iSOCO, (SWEQ Case Study)

> Other examples...



- Sun's White Paper and System Handbook **collections**
- Nokia's **S60 support** portal
- Harper's Online Magazine
- Oracle's **virtual pressroom**
- Opera's **community site**
- Dow Jones' **Synaptica**



> Thank you for your attention!



- These slides are publicly available on:

<http://www.w3.org/2008/Talks/0421-Beijing-IH/>

(under Creative Commons Attributions' license)