Semantics for Biodiversity Primer Session

Steve Baskauf and Mark Schildhauer

W3C Resource Description Framework http://www.w3.org/RDF/

W3C Semantic Web Activity http://www.w3.org/2001/sw/ logos used according to usage guidelines

Preliminaries:

This talk assumes you have knowledge on the level of the Primer video.

There is a link to that video at the end of this slide set.

This slide set is also linked form the Beginner's Guide to RDF webpage on the RDF Task Group website.

QR codes and hyperlinks lead to "try it yourself" resources.

What is the Semantic Web?

"The Semantic Web is about two things. It is about **common formats for integration** and **combination of data** drawn from diverse sources …

It is also about **language** for recording how the data relates to real world objects."

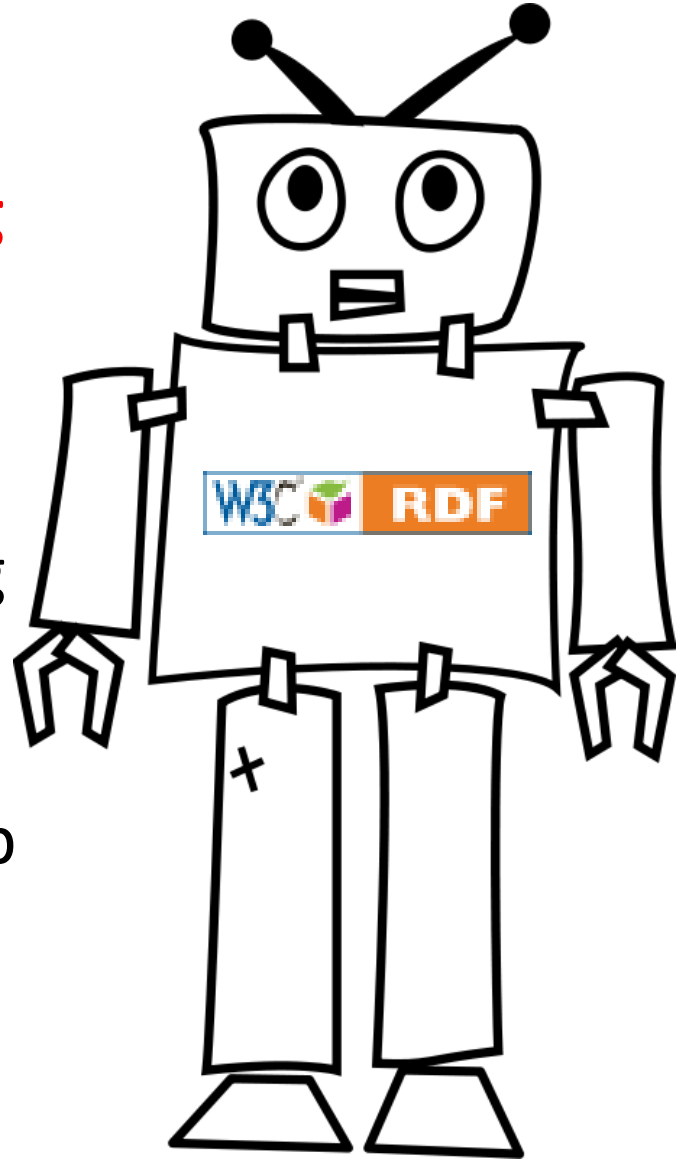
W3C Semantic Web Activity http://www.w3.org/2001/sw/

Semantics

"Semantic" in Semantic Web implies **machine reasoning** about the meaning of the exchanged data.

- Semantic reasoning can occur outside of the Web
- Semantic reasoning can occur without RDF, e.g. using PROLOG.

But often semantic reasoning **is** associated with the Web and RDF so this session is focused primarily on **RDF** and the **Semantic Web**.



Magic "machine" who can reason using RDF.

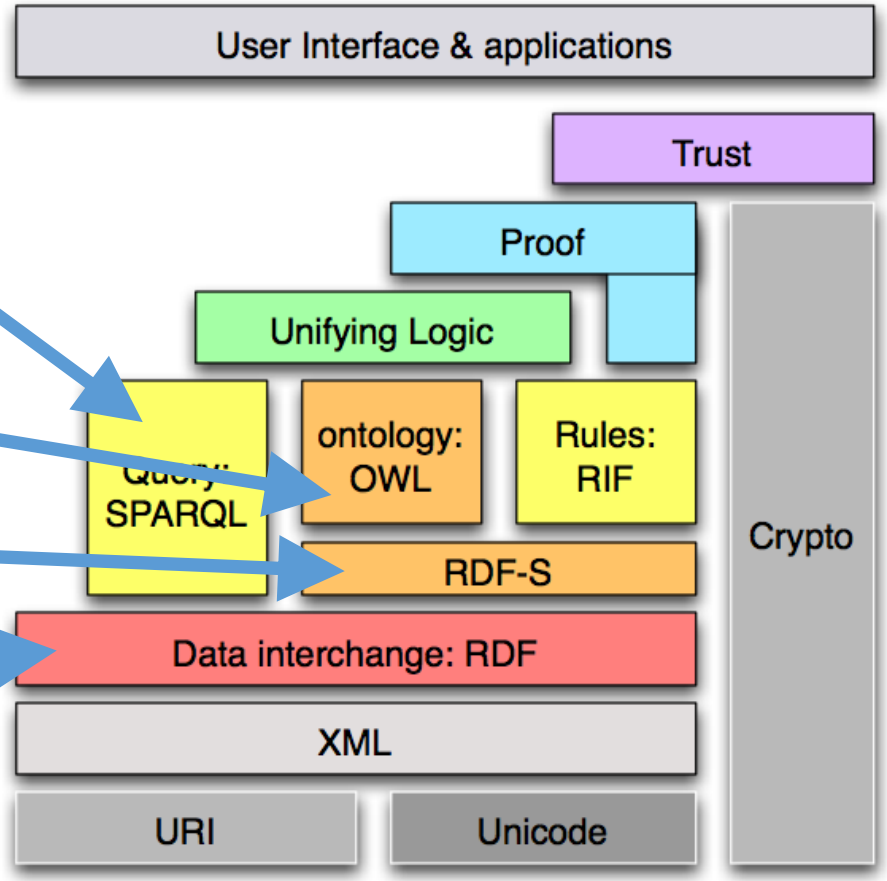
The Semantic "Stack"

Querying (SPARQL)

Property entailments (inverse, reflexive, transitive, etc.)
Class restrictions (cardinality, disjointness, etc.)
Equivalence (*sameAs*, *equivalentProperty*, etc.)

Notions of classes (domain and range) **and datatypes**

Triple and graphs



Tim Berners-Lee via Steve Bratt, W3C http://www.w3.org/2006/Talks/1023-sb-W3CTechSemWeb/W3CTechSemWeb.pdf

Part I: What is RDF good for ?

A. RDF provides a formal model and syntaxes to describe abstract ideas and relationships (formal semantics)

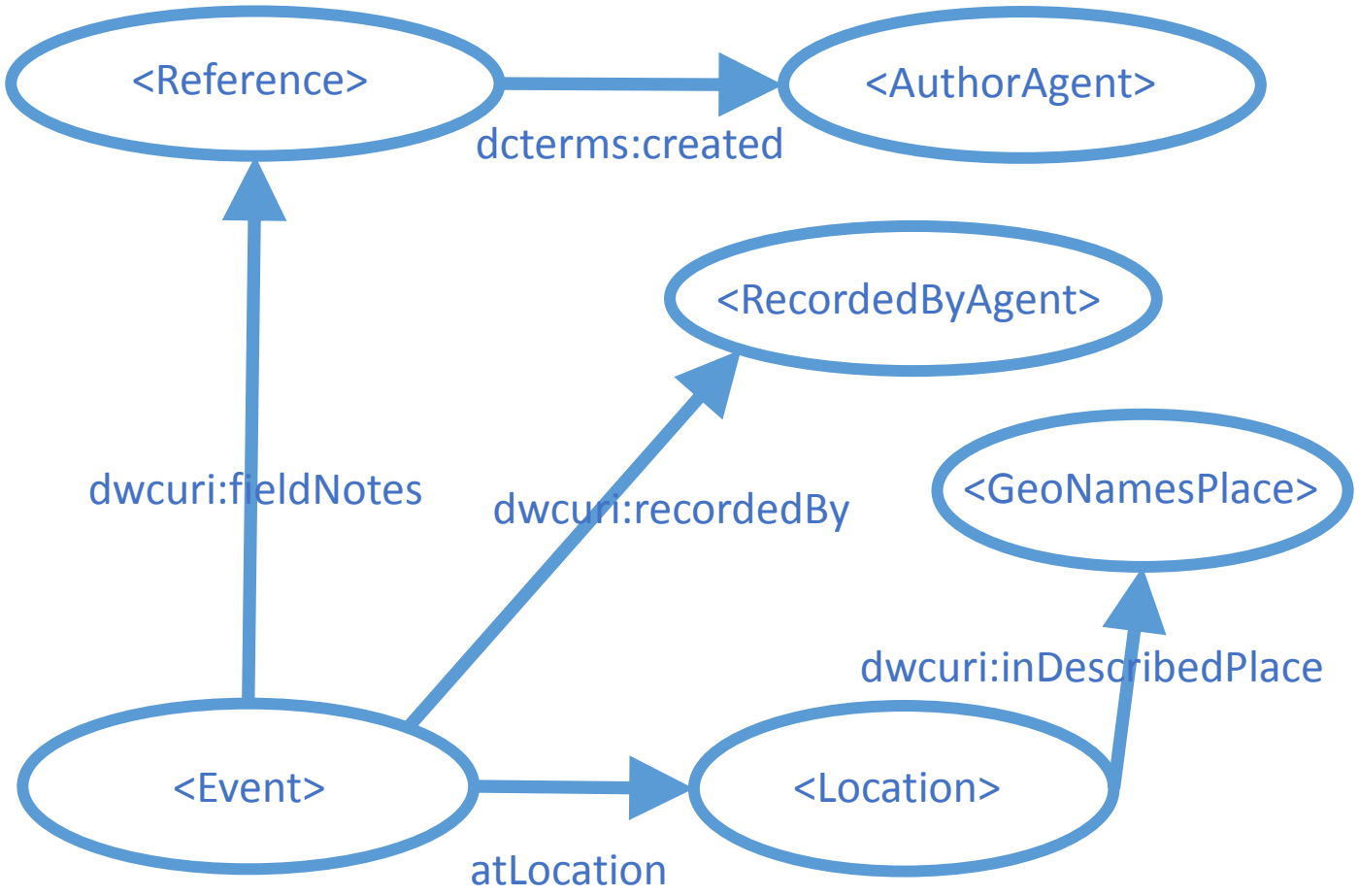
B. There is the potential to discover additional information about existing resources (Linked Data)

C. One can infer previously unstated facts based on logic (entailment)

D. One can evaluate the state of affairs in a set of asserted triples (consistency checking)

The Rod Page Challenge

Image from Rod Page Twitter feed



A. RDF provides a formal model and syntaxes to describe abstract ideas and relationships .

Formal Semantics of RDF

- RDF is described using **model theory** to specify its semantics.
- Its graph-based abstract syntax uses "names " (URIs and literals) to denote "things" (resources). URIs correspond to unique **identifiers**.
- It uses simple expressions of **relationships** (triples) to show how things are connected.

Triple = **subject** | **predicate** | **object**
(or **described resource** | **property** | **value**)
Graph = a set of triples

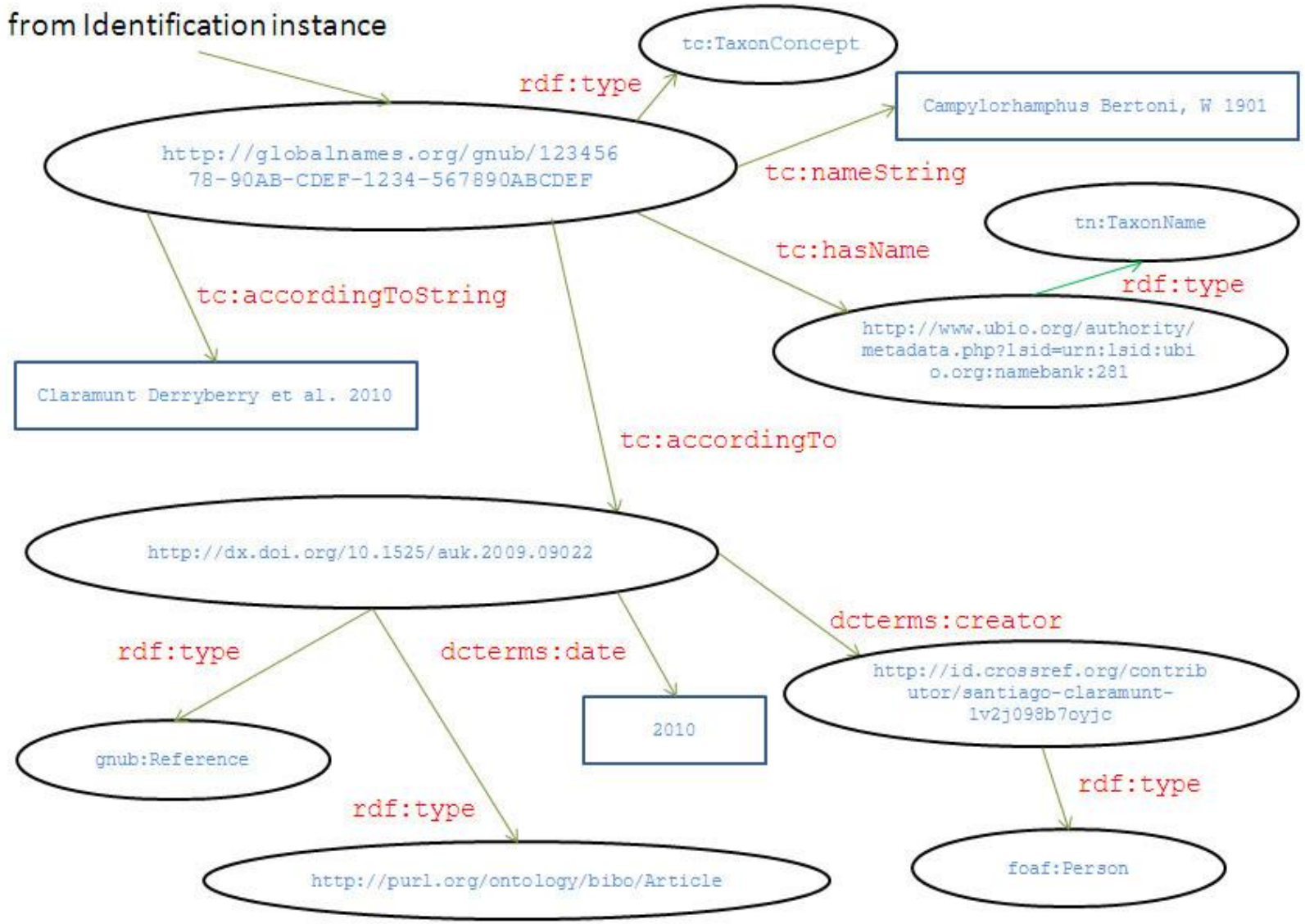
Describing abstract ideas: Taxon Concept vs. TNU ?

Example:

- 38 emails discussing: "What is a taxon concept"?
- Is it the same thing as, or similar to a "taxon name usage" (TNU)?

Resolution:

- expressing as a graphical RDF diagram showed them to be nearly the same.

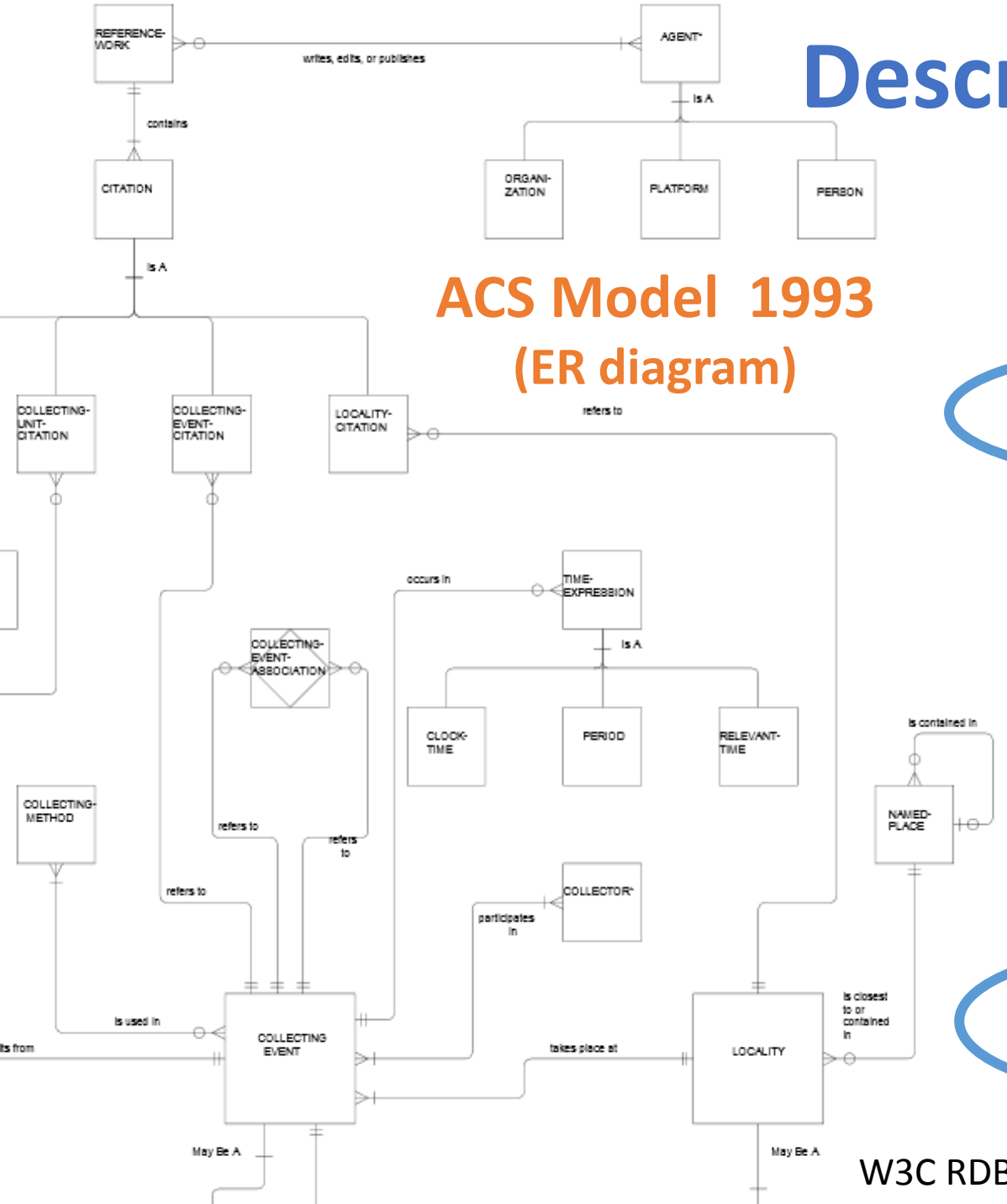


http://code.google.com/p/tdwg-rdf/wiki/TCSthread

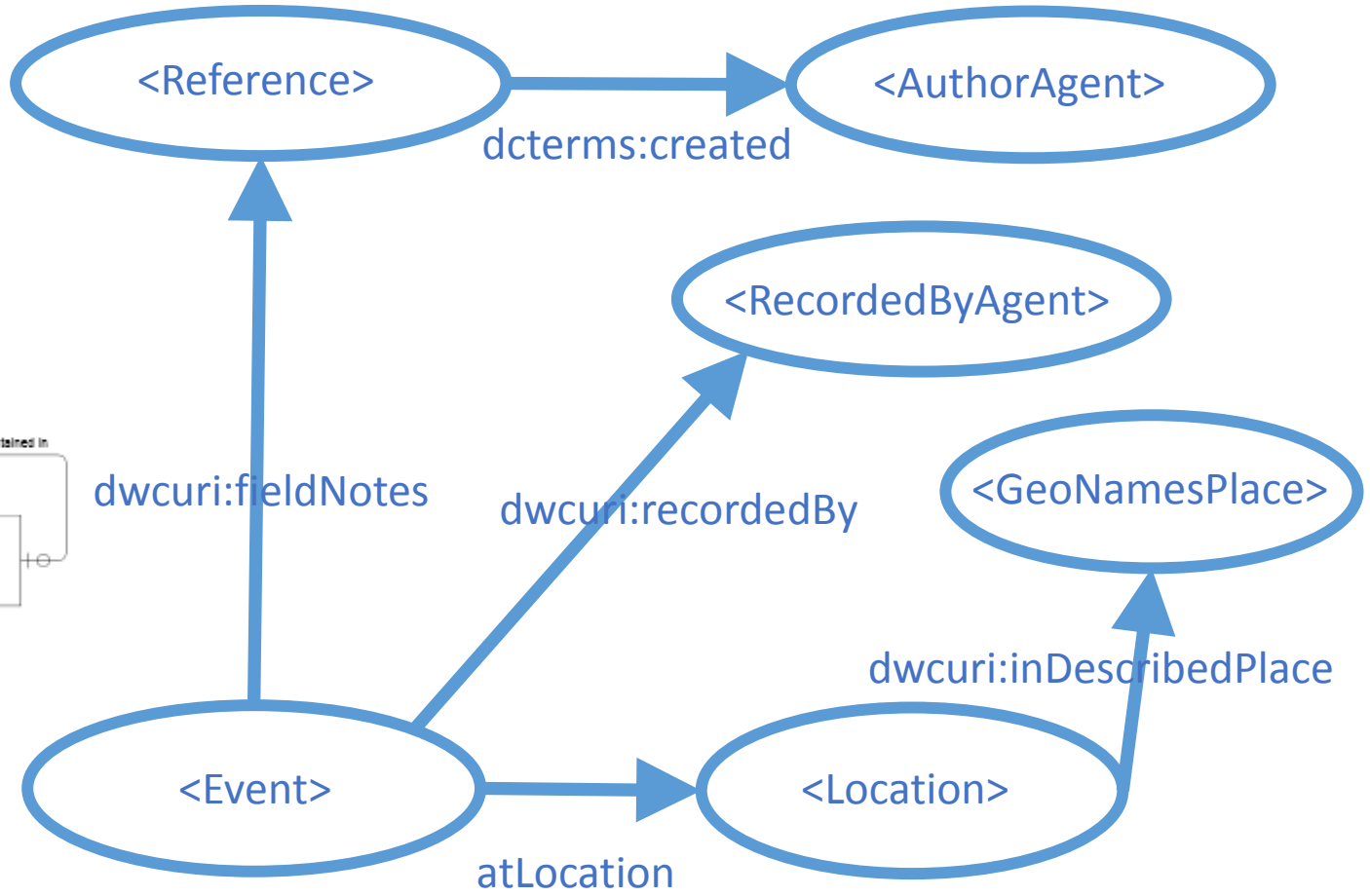
image from http://code.google.com/p/tdwg-rdf/wiki/TCSBasedTaxon

Describing Relationships

ACS Model 1993
(ER diagram)



Darwin/Dublin Core 2013
(RDF graphical representation)



W3C RDB2RDF Working Group http://www.w3.org/2001/sw/wiki/RDB2RDF

Database

Triplifying records from a view of an RDBMS

dwc:occurrenceID	dwc:recordedBy	dwc:occurrenceRemarks	dwc:eventID
LangdonTree_0134	K. R. Langdon	observed, specimen not collected	gsmp0739
LangdonTree_0133	K. R. Langdon	juvenile on north-facing slope	gsmp0739

primary key/ subject

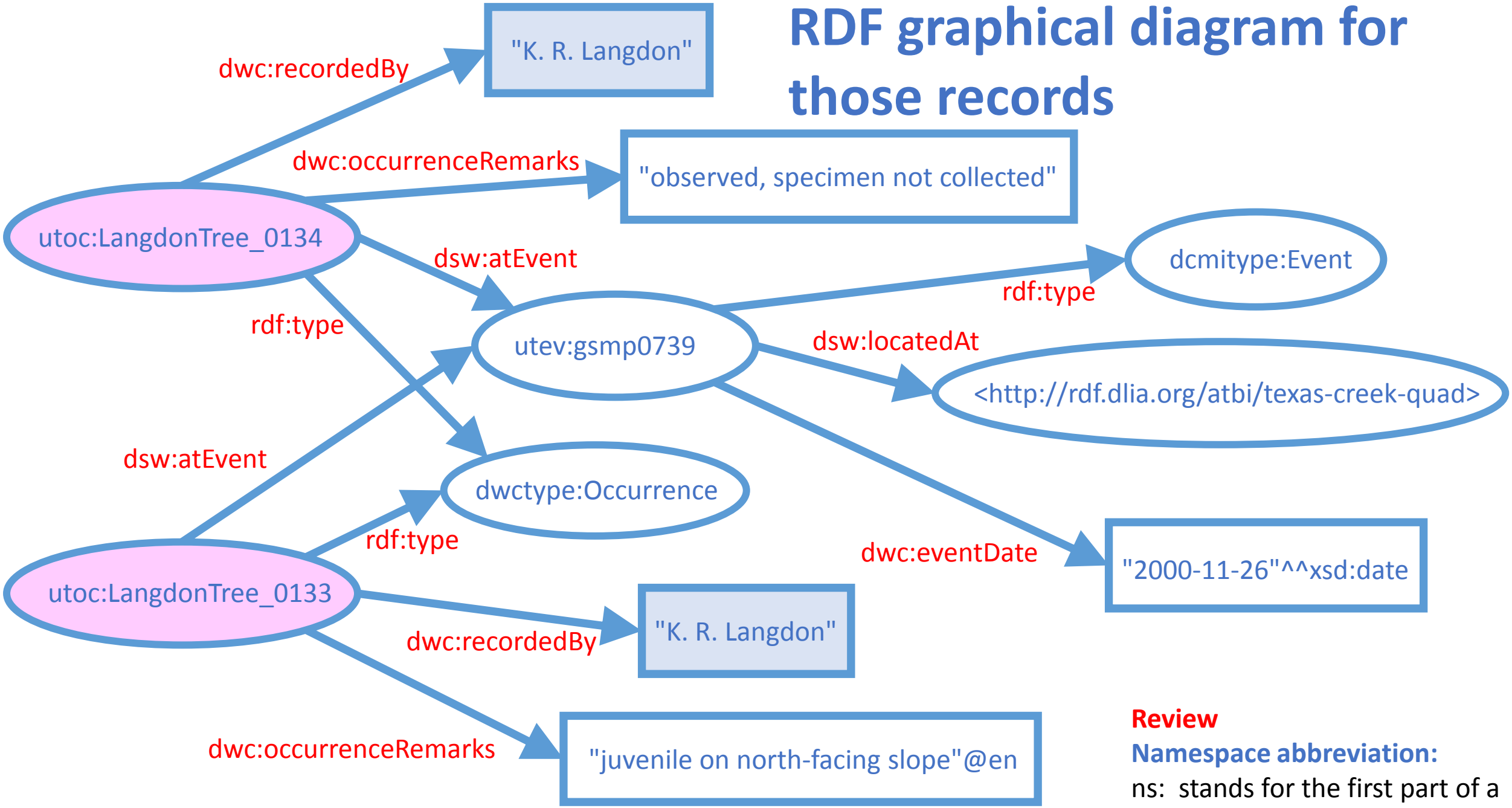
dwc:eventID	dwc:eventDate	dwc:locationID
gsmp0739	2000-11-26	http://rdf.dlia.org/atbi/texas-creek-quad

foreign key/ object

RDF/ Turtle

```
<http://tremont22.utk.edu/occurrence/LangdonTree_0134>
      a dwctype:Occurrence;
      dwc:recordedBy "K. R. Langdon";
      dwc:occurrenceRemarks "observed, specimen not collected"@en;
      dsw:atEvent <http://tremont22.utk.edu/event/gsmp0739>.
<http://tremont22.utk.edu/occurrence/LangdonTree_0133>
      a dwctype:Occurrence;
      dwc:recordedBy "K. R. Langdon";
      dwc:occurrenceRemarks "juvenile on north-facing slope"@en;
      dsw:atEvent <http://tremont22.utk.edu/event/gsmp0739>.
<http://tremont22.utk.edu/event/gsmp0739>
      a dcmitype:Event;
      dwc:eventDate "2000-11-26"^^xsd:date;
      dsw:locatedAt <http://rdf.dlia.org/atbi/texas-creek-quad>.
```

RDF graphical diagram for those records

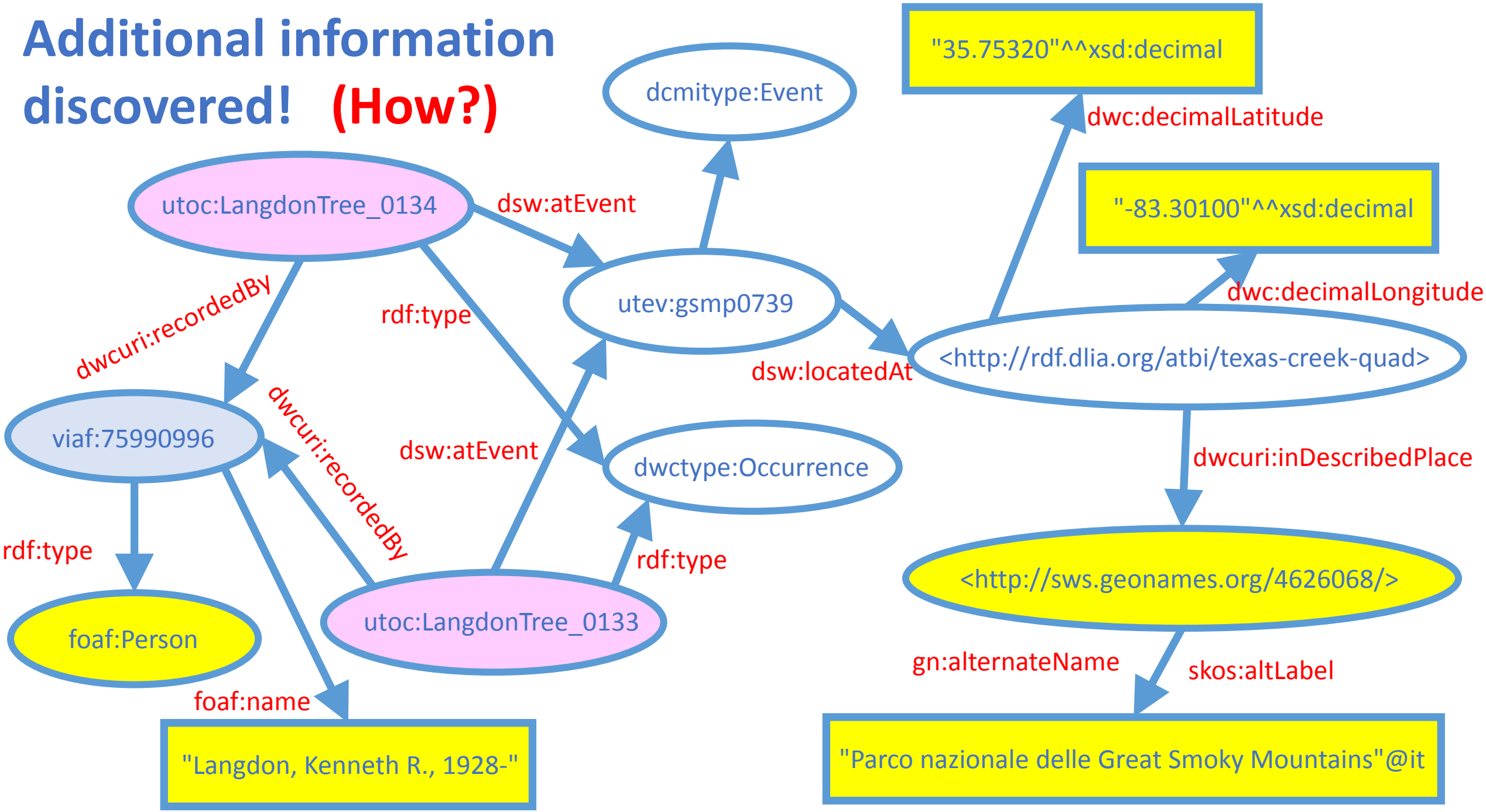


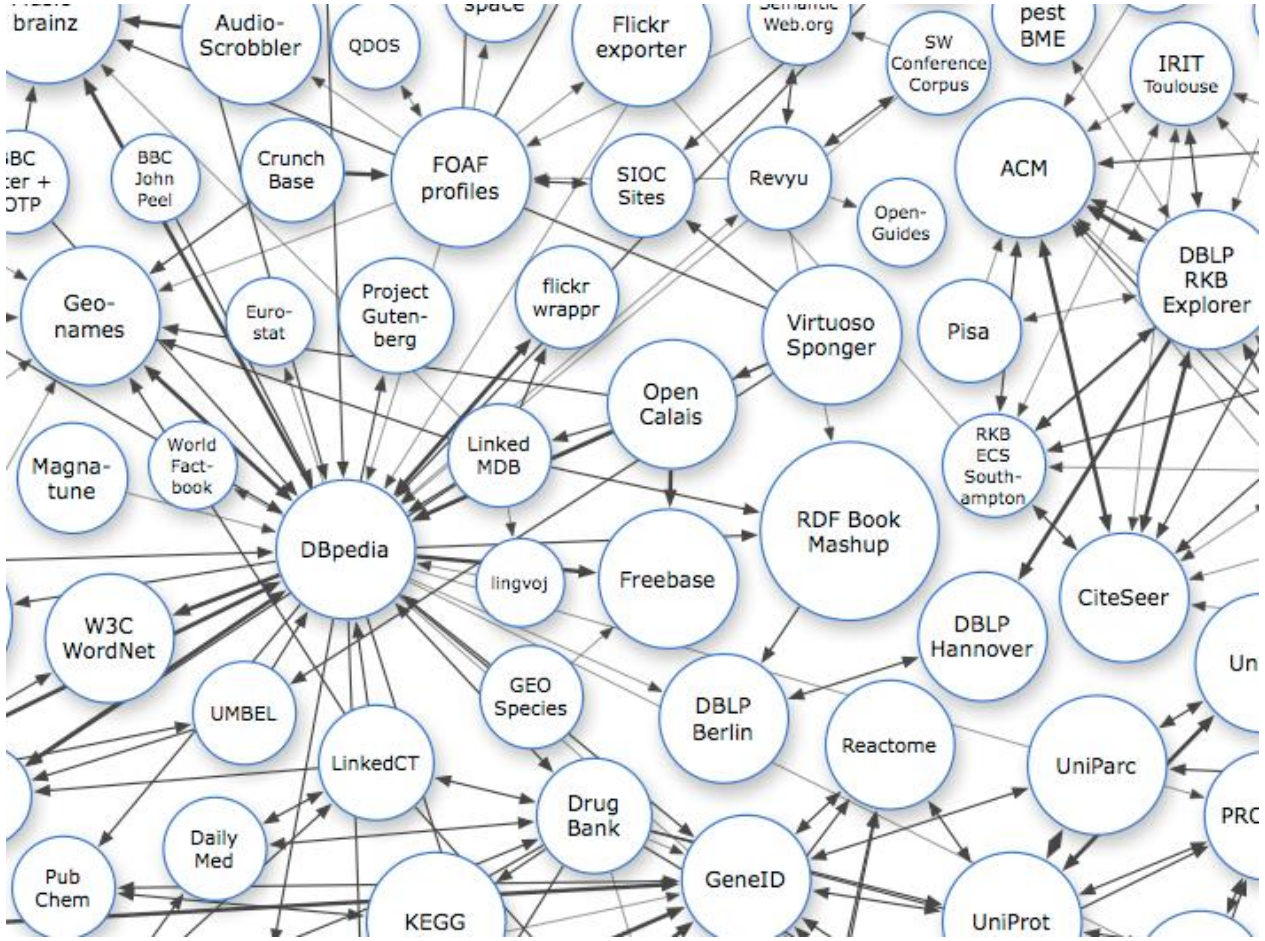
Review

Namespace abbreviation:

ns: stands for the first part of a URI (see Primer video)

Additional information discovered! (How?)





B. There is the potential to discover additional information about existing resources

Linking Open (LOD) Data Project Cloud Diagram from http://linkeddata.org/ CC BY-SA

Linked Data

Tim Berners-Lee expressed the "**Linked Data Principles**" in 2006:

1. Use **URIs** as names for things.
2. Use **HTTP URIs**, so that people can look up those names.
3. When someone **looks up a URI**, provide useful information, using the standards (RDF, SPARQL).
4. Include links to other URIs, so that they can **discover** more things.

"Linked Data" is a similar idea to "the Semantic Web" but focused on HTTP URIs as identifiers and more on data discovery than reasoning.

"Linked Open Data" (LOD) is Linked Data with an open license that does not impede reuse.

http://www.w3.org/DesignIssues/LinkedData.html

HTTP URIs as identifiers

HTTP URIs combine an **identifier** function (URI) with an **exchange protocol** (HTTP).

In theory, a client dereferencing the identifier can retrieve RDF about the identified resource.

In our community, it is a best practice that URIs used to identify resources should be **persistent**. This requirement means that providers should think carefully before minting and exposing HTTP URIs.

http://www.w3.org/DesignIssues/LinkedData.html

```
<http://bioimages.vanderbilt.edu/baskauf/79649#loc>
    dwcuri:inDescribedPlace <http://sws.geonames.org/4617305/>.
```

HTTP Request Header:

```
GET //sws.geonames.org/4617305/ HTTP/1.1
Accept: application/rdf+xml
```

Response Header:

```
HTTP/1.1 303 See Other
Location: http://sws.geonames.org/4617305/about.rdf
…
```

HTTP Request Header:

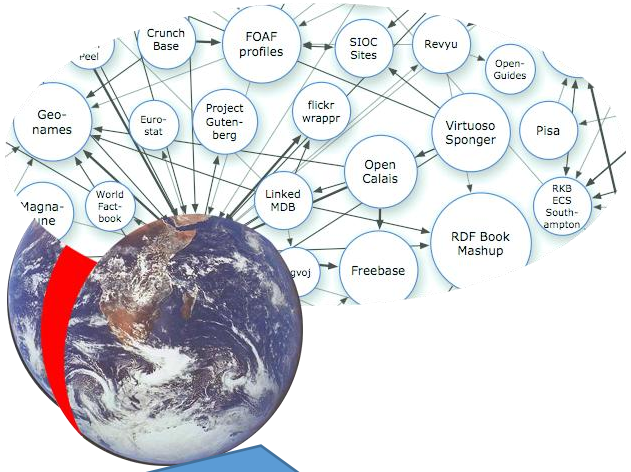
```
GET //sws.geonames.org/4617305/about.rdf HTTP/1.1
Accept: application/rdf+xml
```

Response Header:

```
HTTP/1.1 200 OK
Content-Type: application/rdf+xml;charset=UTF-8
…
```

Response Body:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<rdf:RDF xmlns:cc="http://creativecommons.org/ns#" …>
<gn:Feature rdf:about="http://sws.geonames.org/4617305/">
<gn:name>Davidson County</gn:name>
<gn:alternateName xml:lang="fr">Comté de Davidson</gn:alternateName>
<gn:alternateName xml:lang="bg">Дейвидсън</gn:alternateName>
<gn:alternateName xml:lang="ja">デイヴィッドソン郡</gn:alternateName>
<gn:countryCode>US</gn:countryCode>
<gn:population>626681</gn:population>
<gn:parentFeature rdf:resource="http://sws.geonames.org/4662168/"/>
<rdfs:seeAlso rdf:resource="http://dbpedia.org/resource/Davidson_County%2C_Tennessee"/>
…
```



Example (dereferencing a URI)

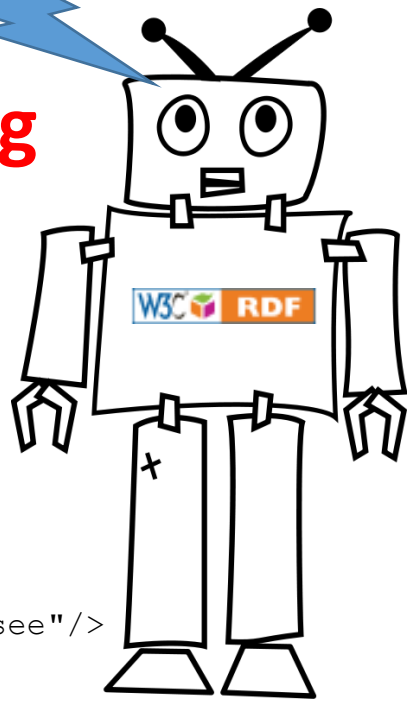


image of infamous "Bicentennial Oak"
http://bioimages.vanderbilt.edu/baskauf/79649

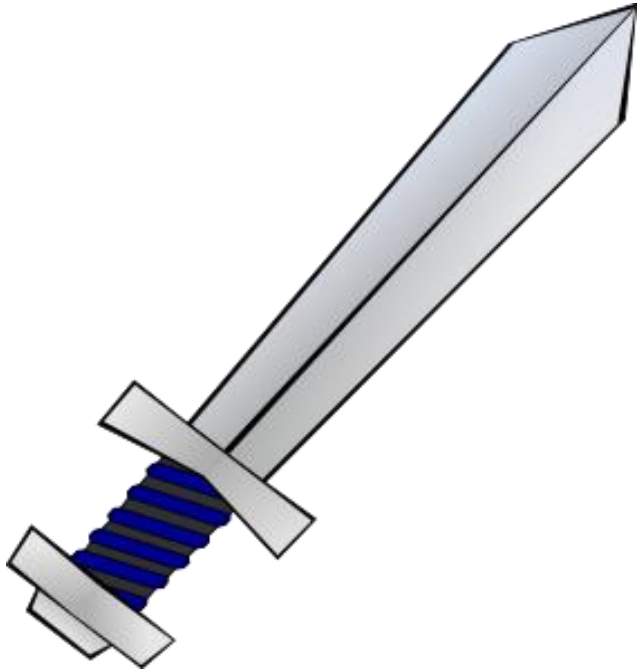
Linking Open (LOD) Data Project Cloud Diagram from
http://linkeddata.org/ CC BY-SA

Dereferencing HTTP URIs (cont.)

- The method illustrated is called a "303 redirect". Darwin Core and Dublin Core terms do this when their terms are dereferenced.
- There are other methods (hash URIs, RDFa embedded in Web pages).
- There is no guarantee that an HTTP URI will dereference to anything.
- There is no guarantee that an HTTP URI will dereference to RDF for machines (requesting **Content-type: application/rdf+xml**).
- Making HTTP URIs dereference to RDF for machines is generally considered a good thing (e.g. Recommendation 7 of the TDWG GUID Applicability Statement standard)

"In general, it is not assumed that complete information about any resource is available. **RDF does not prevent anyone from making assertions that are nonsensical or inconsistent with other statements, or the world as people see it.** Designers of applications that use RDF should be aware of this and may design their applications to tolerate incomplete or inconsistent sources of information."

RDF Concepts and Abstract Syntax http://www.w3.org/TR/rdf-concepts/



The double-edged sword

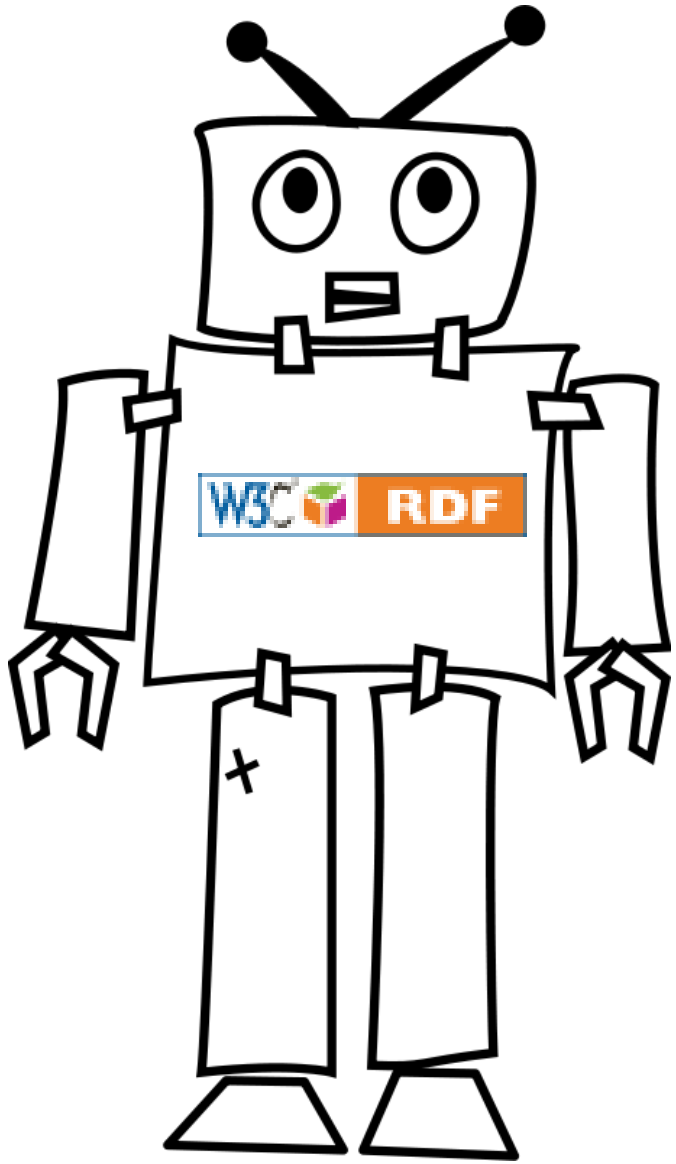
Linked Data Pros and Cons

We may discover new useful information, but:

- we could incorporate triples that are "bad" into our graph (incorrect info, spam, sabotage)
- we may create inconsistencies via triples we introduce into our graph (carelessness)

In a scientific context triples should come from verified sources (known **provenance**).

C. One can infer previously unstated facts based on logic

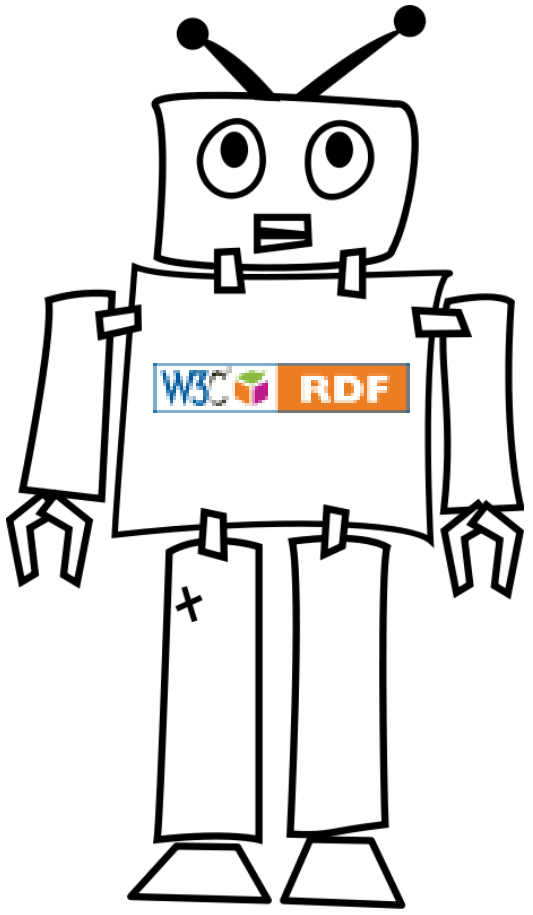


How can a “machine” can “infer unstated facts” based on RDF ?

A semantic client:

- is **software** which is constructed to work according to the rules laid out by standards
- **consumes** information in the form of RDF **triples**

A semantic client can also be called a “reasoner”. How does it “reason”?

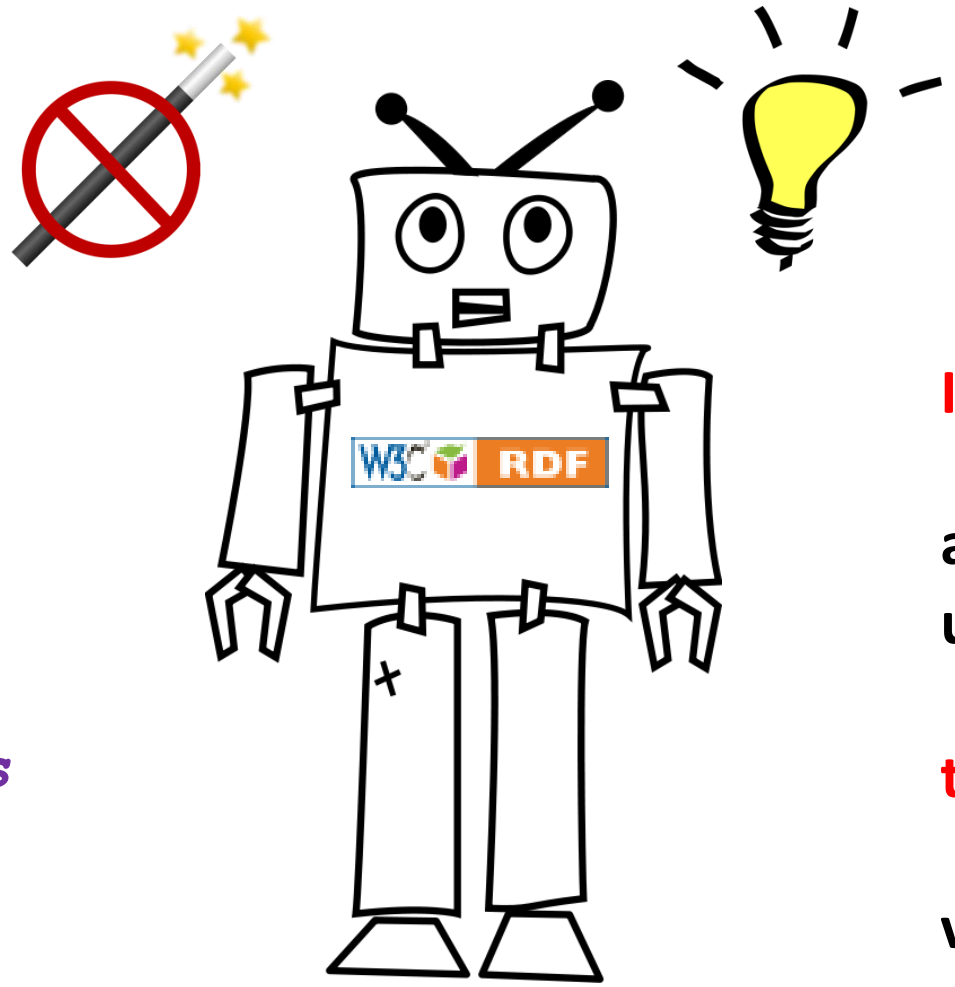


A semantic client does **not** "know" what URIs "mean":

dwc:decimalLatitude

vs:

xq:p2-glwsopgn_2q4as



But it can follow rules:

If

aaa *rdfs:range* XXX.
uuu aaa vvv.

then

vvv *rdf:type* XXX.

"The chief utility of a formal semantic theory is **not** to provide any deep analysis of the nature of the things being described by the language or to suggest any particular processing model, but rather to provide a technical way to determine when inference processes are valid, i.e. when they preserve truth."
RDF Semantics http://www.w3.org/TR/rdf-mt/

Entailment

An **interpretation** assigns special meaning to an RDF vocabulary.

For example, the **rdfs-interpretation** satisfies the a semantic condition which establishes an **entailment rule** about ranges and class membership:

Rule *rdfs3*:
If graph E contains {aaa `rdfs:range` XXX. uuu aaa vvv.} then add {vvv `rdf:type` XXX.}

What does that mean???

Application of an entailment rule

The FOAF (Friend Of A Friend) vocabulary asserts:

```
foaf:depiction rdfs:range foaf:Image.
```

This does **NOT** mean that the object of a triple containing *foaf:depiction* **must be** an image.

The RDF allows the predicate *foaf:depiction* to be used with **any kind of object**.

The entailment rule ***rdfs3*** means that that a semantic client can generate an **inferred triple** stating that the *rdf:type* of the object is *foaf:Image*.

FOAF = Friend of a Friend vocabulary http://xmlns.com/foaf/spec/

Entailment rule example

The RDF allows me to assert that:

```
<http://viaf.org/viaf/9854560> foaf:depiction
<http://commons.wikimedia.org/wiki/File:Van_Gogh_Age_19.jpg>.
```

In English we would say:

{The person Vincent van Gogh} has a depiction {a certain jpeg image}

From the range of *foaf:depiction*, a client can infer that:

```
<http://commons.wikimedia.org/wiki/File:Van_Gogh_Age_19.jpg> rdf:type foaf:Image.
```

Hooray!!! We have gained new knowledge!

RDF also allows me to assert that:

```
<urn:lsid:ubio.org:namebank:111731>
      foaf:depiction <http://dbpedia.org/resource/Moby-Dick>.
```

In English we would say:

{The name *Physeter macrocephalus* Linnaeus, 1758} has a depiction {the novel Moby Dick}

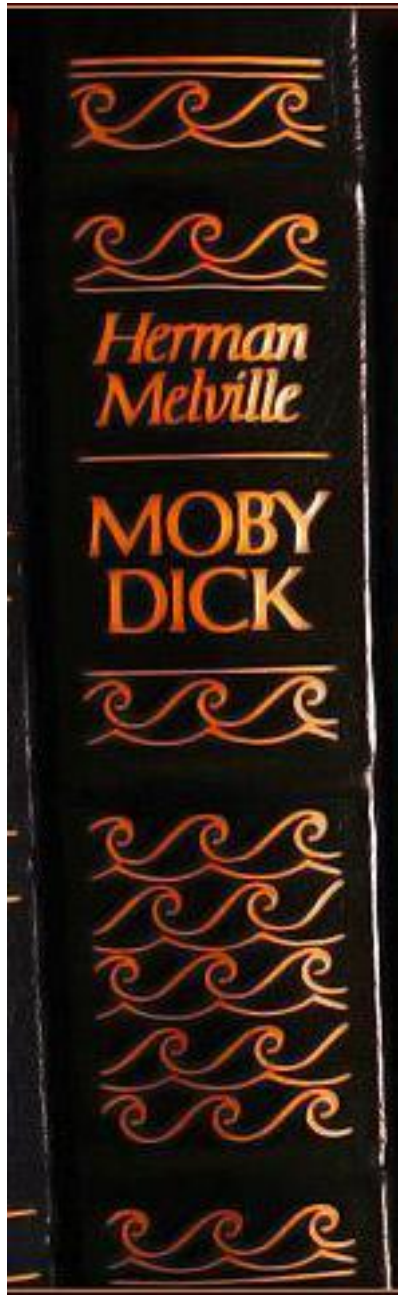
DBpedia declares

```
<http://dbpedia.org/resource/Moby-Dick>
   rdf:type bibo:Book
```

But a semantic client infers

```
<http://dbpedia.org/resource/Moby-Dick>
      rdf:type foaf:Image.
```

based on the range declaration of *foaf:depiction*



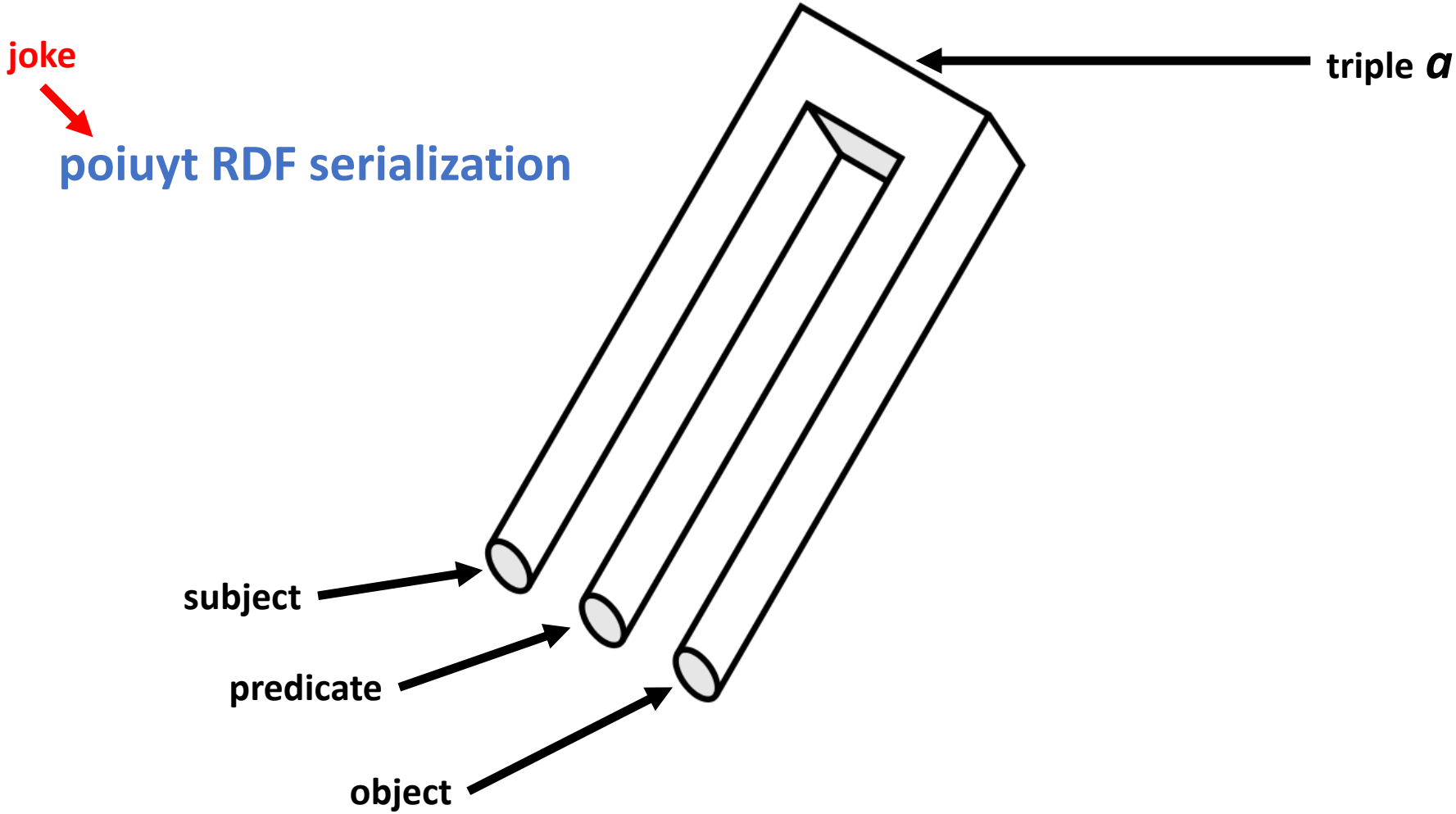
A novel is an image !!! Oops. We must be more careful with *foaf:depiction* because of its range declaration.

Entailment summary

- Entailment rules do **NOT enforce** conditions.
- Entailment rules **imply** that other unstated triples exist.
- Inferred triples are true to the extent that the statements which entail them are also true. This introduces a requirement for an element of **trust**.
- A client is **not required** to apply all possible entailment rules. So one can't assume that all unstated entailed triples will be inferred.

"a set of statements ***A*** entails a statement ***α*** if in any state of affairs wherein all statements from ***A*** are true, also ***α*** is true."

Quote from section 3 of OWL 2 Primer http://www.w3.org/TR/owl2-primer/#Modeling_Knowledge:_Basic_Notions

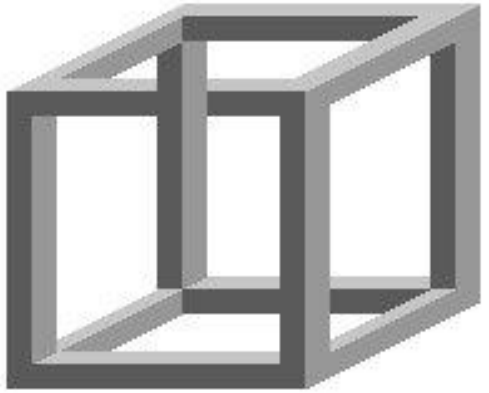


D. One can evaluate the state of affairs in a set of asserted triples

“Stronger” entailment

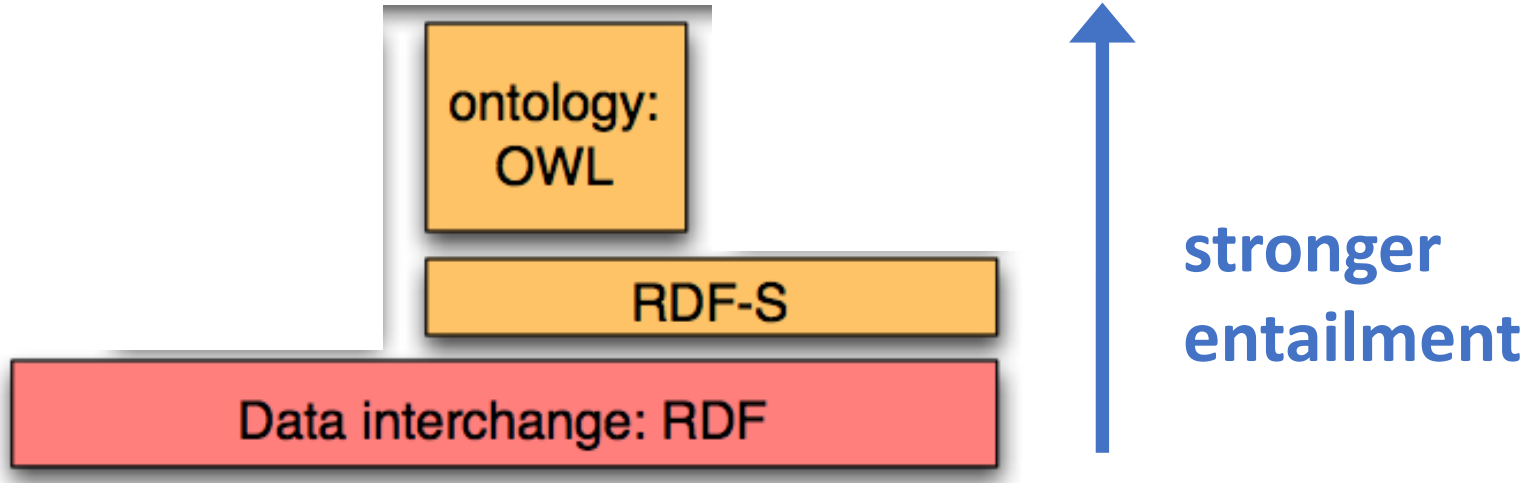
As we move up the “semantic stack” from **RDF** to **RDF-S** to **OWL** (Web Ontology Language), stronger entailment provides **greater opportunities to reason** but a greater danger of generating **inconsistencies**.

inconsistent cube/license



anonymous
CC Attribution-Share Alike

real

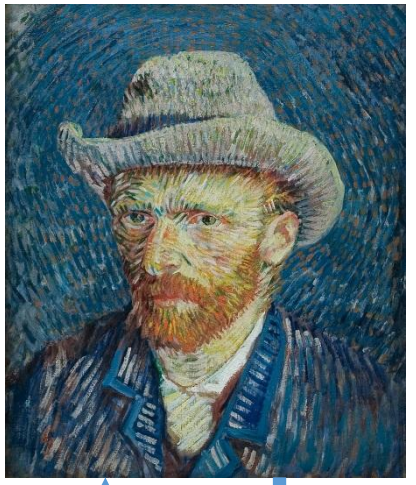


stronger entailment

http://commons.wikimedia.org/wiki/File:Impossible_cube.jpg

Example 1: FOAF Vocabulary and OWL

Stronger entailment → More information



The FOAF Vocabulary uses numerous properties from OWL.

```
foaf:maker owl:equivalentProperty dcterms:creator.
foaf:maker owl:inverseOf foaf:made.
```

dcterms: creator

foaf: made

If a graph includes the triple

```
viaf:9854560 foaf:made dbpedia:Starry_Night.
```

a reasoner can add the following triple to the graph

```
dbpedia:Starry_Night dcterms:creator viaf:9854560.
```

FOAF vocabulary http://xmlns.com/foaf/spec/

Paintings by Vincent van Gogh

Example 2: FOAF Vocabulary and OWL

Stronger entailment → More possibilities for inconsistencies

Triple describing TDWG as an organization:

```
<http://www.tdwg.org/> a foaf:Organization.
```

Triple describing the TDWG homepage:

```
<http://www.tdwg.org/> a foaf:Document.
```

But the FOAF vocabulary says:

```
foaf:Document owl:disjointWith foaf:Organization.
```

(i.e. it is inconsistent for a resource to be both a document and an organization).

RDF allows anyone to say these things about TDWG. But a client reasoning under owl-interpretation would detect an inconsistency.

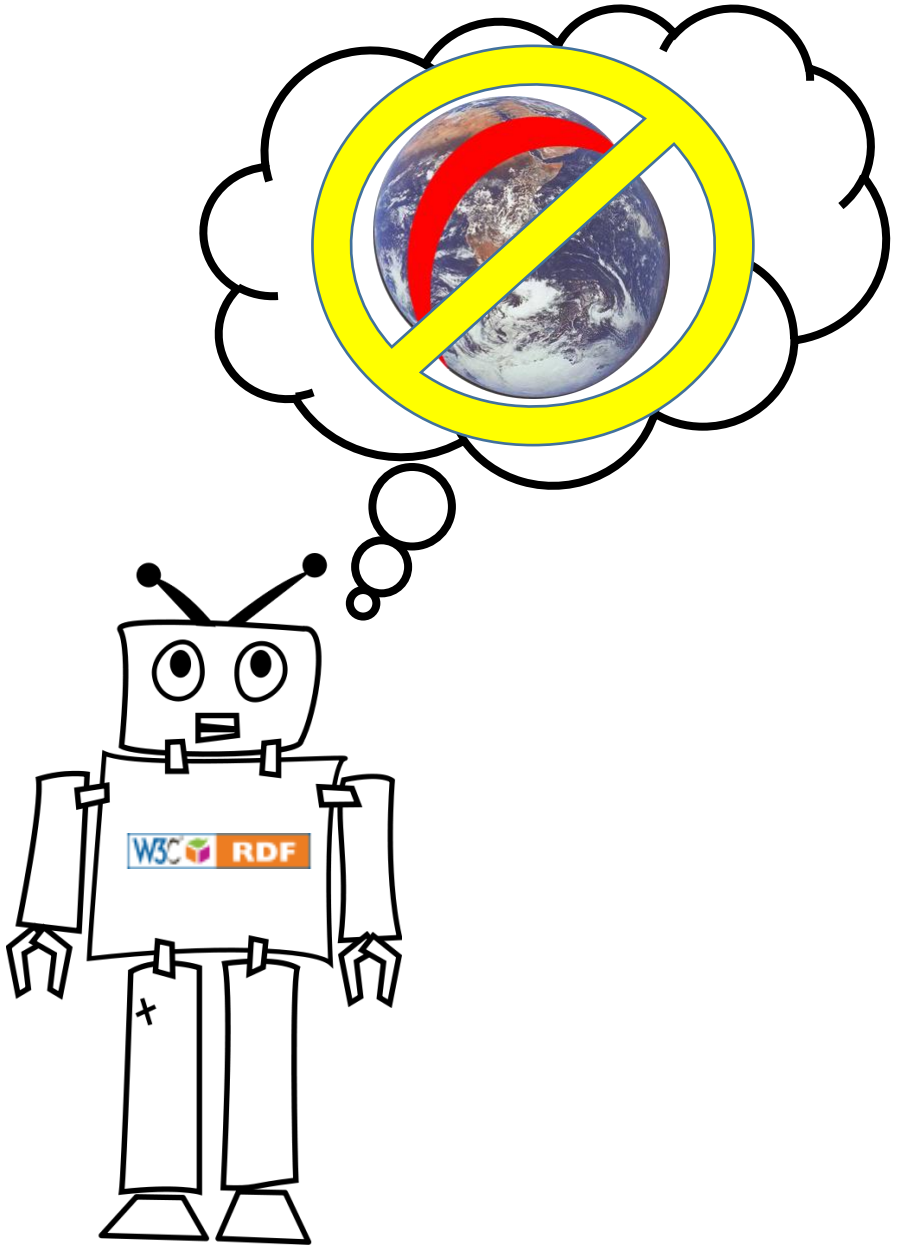
Definition of *foaf:Document* http://xmlns.com/foaf/spec/#term_Document

Summary:

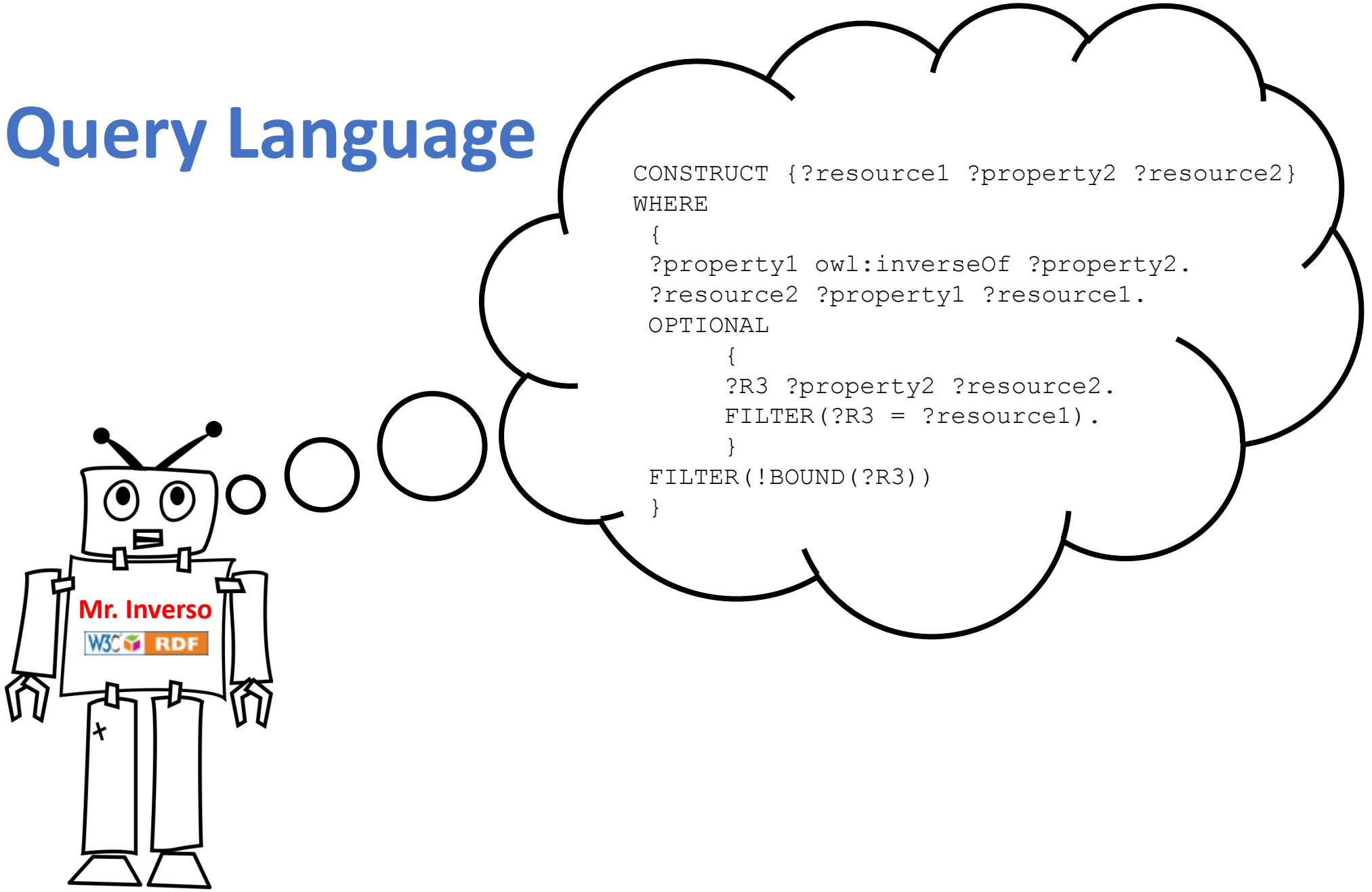
In RDF, one does not say "your triple is not allowed in my world".

Rather, RDF asks "what kind of world do I have after I include your triple?"

If there is **no kind of world** that can logically exist given a certain set of triples in a graph, then the graph is **inconsistent**.



Part II:
SPARQL Query Language



SPARQL: SPARQL Protocol and RDF Query Language

SPARQL is a query language that can screen triples by requiring that they conform to a pattern, such as

```
?Location  dwc:stateProvince "Hawaii".
```

?Location is a **variable** which can have any URI value.

A query using the SELECT query form

```
PREFIX dwc: <http://rs.tdwg.org/dwc/terms/>

SELECT ?Location WHERE
   {
   ?Location  dwc:stateProvince "Hawaii".
   }
Limit 20
```

query at uriburner
click on Advanced tab

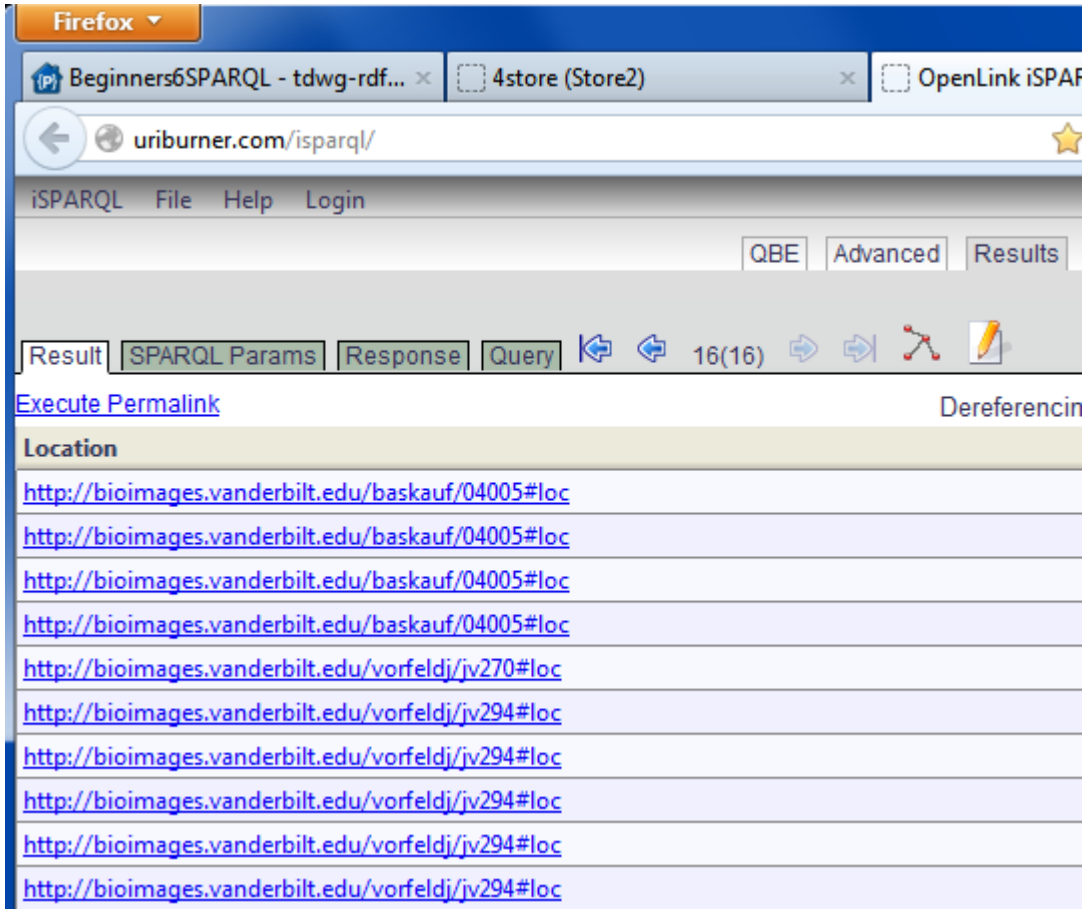
The **SELECT** query form returns variable **bindings**. This query shows all terms (URIs and literals) that are **bound** to `?Location` based on the triple **patterns** that follow WHERE.

SPARQL results

The query is submitted to a SPARQL HTTP service **endpoint** that screens a graph for the **triple pattern**.

```
<?xml version="1.0"?>
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
  <head>
    <variable name="Location"/>
  </head>
  <results>
    <result>
      <binding
name="Location"><uri>http://bioimages.vanderbilt.edu/baskauf/04313#loc</uri></binding>
    </result>
    <result>
      <binding
name="Location"><uri>http://bioimages.vanderbilt.edu/baskauf/04103#loc</uri></binding>
    </result>
...
</sparql>
```

raw XML returned from endpoint



results presented in iSPARQL interface

What can we do with these URIs???

DESCRIBE query form

The **DESCRIBE** query form returns all triples that contain the bound URI in any position.

http://bioimages.vanderbilt.edu/vorfeldj/jv294#eve	
ns9:locatedAt	http://bioimages.vanderbilt.edu/vorfeldj/jv294#loc

URI as object

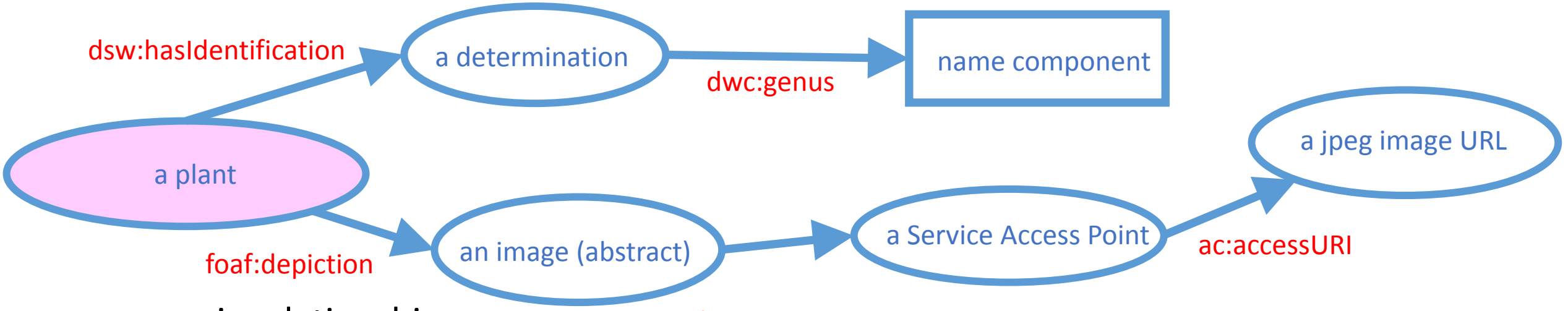
http://bioimages.vanderbilt.edu/vorfeldj/jv294#loc	
rdf:type	http://purl.org/dc/terms/Location
geo:lat	19.63847
geo:long	-155.9334
dwc:continent	OC
dwc:coordinateUncertaintyInMeters	1500
dwc:countryCode	US
dwc:county	Hawaii
dwc:decimalLatitude	19.63847
dwc:decimalLongitude	-155.9334
dwc:geodeticDatum	EPSG:4326
dwc:locality	Holualoa, Waiaha Springs Forest Reserve
dwc:stateProvince	Hawaii
dwc:georeferenceRemarks	Location determined from Google maps.
ns9:locates	http://bioimages.vanderbilt.edu/vorfeldj/jv294#eve
ns11:inDescribedPlace	http://sws.geonames.org/5855765/

URI as subject

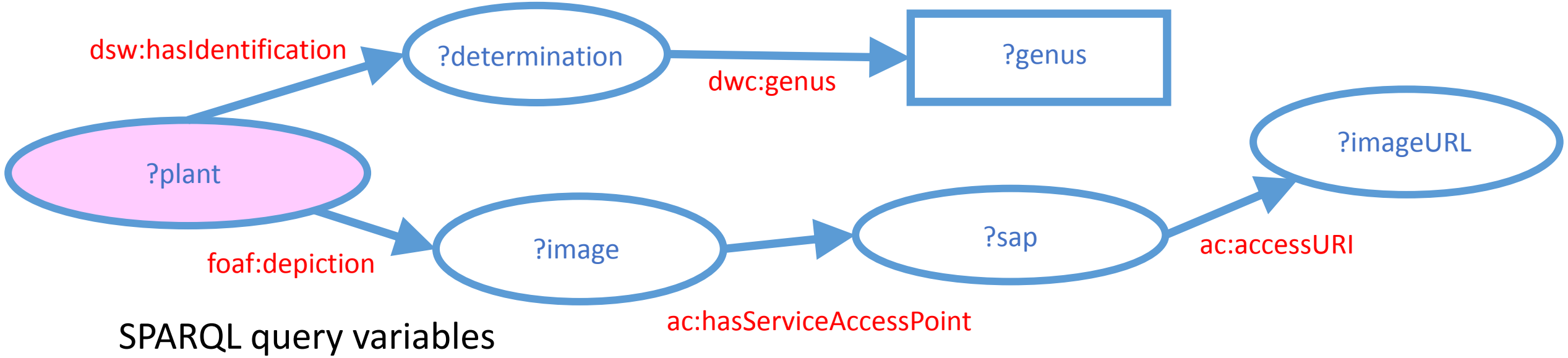
```
DESCRIBE <http://bioimages.vanderbilt.edu/vorfeldj/jv270#loc>
```

SPARQL endpoints can return results in various forms, including XML and JSON.

Constructing a more interesting SPARQL query



generic relationships



SPARQL query variables

The more interesting SPARQL query:

This query finds plants having the name "Echinacea simulata" and gives access URIs of their images.

query at uriburner

click on Advanced tab

```
PREFIX dwc: <http://rs.tdwg.org/dwc/terms/>
PREFIX dsw: <http://purl.org/dsw/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX ac: <http://rs.tdwg.org/ac/terms/>

SELECT ?plant ?imageURL WHERE {
?plant dsw:hasIdentification ?determination.
?determination dwc:genus "Echinacea".
?determination dwc:specificEpithet "simulata".
?plant foaf:depiction ?image.
?image ac:hasServiceAccessPoint ?sap.
?sap ac:variantDescription "Lower Quality".
?sap ac:accessURI ?imageURL.
}
Limit 20
```

Result | SPARQL Params | Response | Query | 11(12)

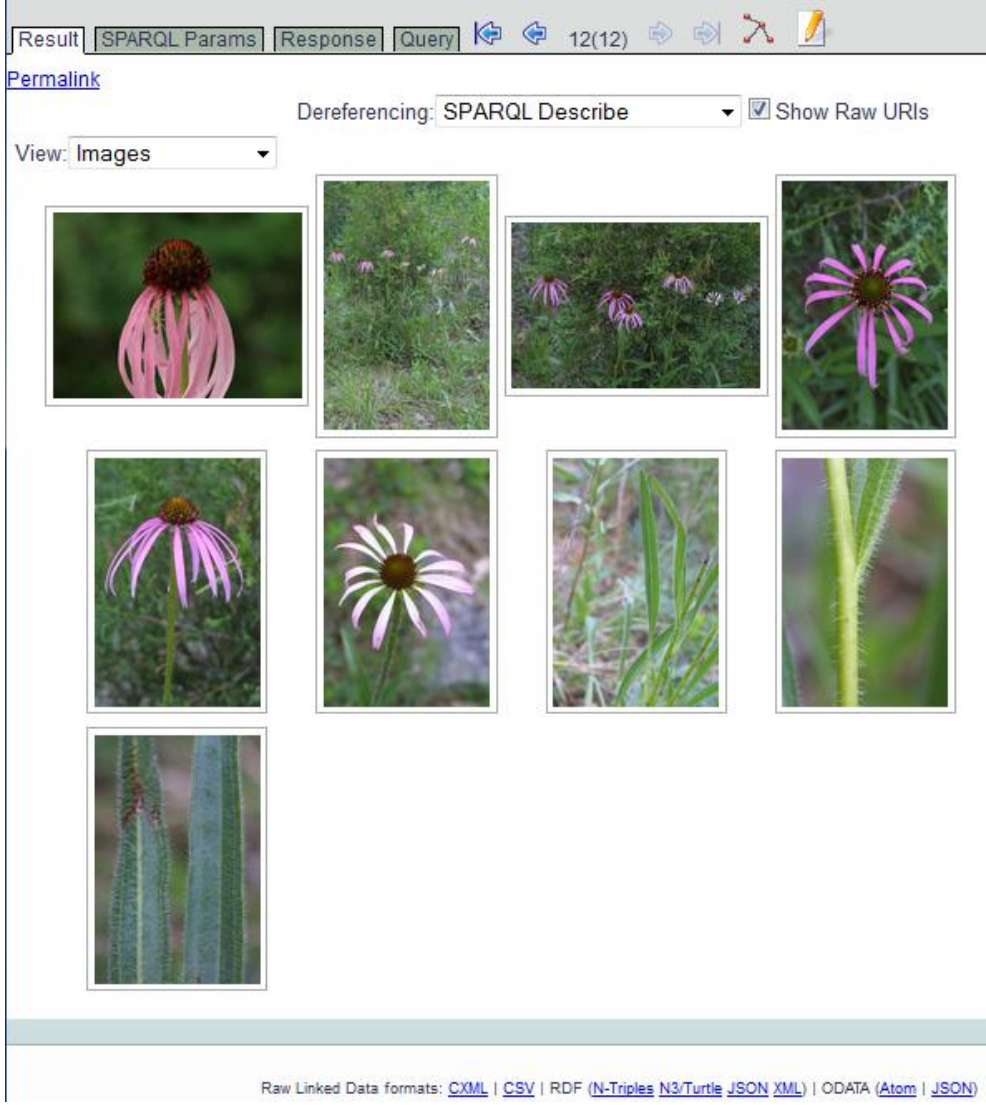
Execute Permalink

Dereferencing: SPARQL Describe

plant	imageURL
http://bioimages.vanderbilt.edu/ind-hessd/e5415	http://bioimages.vanderbilt.edu/lq/hessd/wecsi--flside0149-de5415.jpg
http://bioimages.vanderbilt.edu/ind-baskauf/25157	http://bioimages.vanderbilt.edu/lq/baskauf/wecsi--wp25158.jpg
http://bioimages.vanderbilt.edu/ind-baskauf/25157	http://bioimages.vanderbilt.edu/lq/baskauf/wecsi--flseveral25163.jpg
http://bioimages.vanderbilt.edu/ind-baskauf/25157	http://bioimages.vanderbilt.edu/lq/baskauf/wecsi--flfront25166.jpg
http://bioimages.vanderbilt.edu/ind-baskauf/25157	http://bioimages.vanderbilt.edu/lq/baskauf/wecsi--flside25174.jpg
http://bioimages.vanderbilt.edu/ind-baskauf/25157	http://bioimages.vanderbilt.edu/lq/baskauf/wecsi--flfront25181.jpg
http://bioimages.vanderbilt.edu/ind-baskauf/25157	http://bioimages.vanderbilt.edu/lq/baskauf/wecsi--lf25188.jpg
http://bioimages.vanderbilt.edu/ind-baskauf/25157	http://bioimages.vanderbilt.edu/lq/baskauf/wecsi--st25190.jpg
http://bioimages.vanderbilt.edu/ind-baskauf/25157	http://bioimages.vanderbilt.edu/lq/baskauf/wecsi--lfupper-lower25195.jpg

Results of more interesting query

Software can interface with the endpoint to use query results (e.g. JSON) to create software output for users.



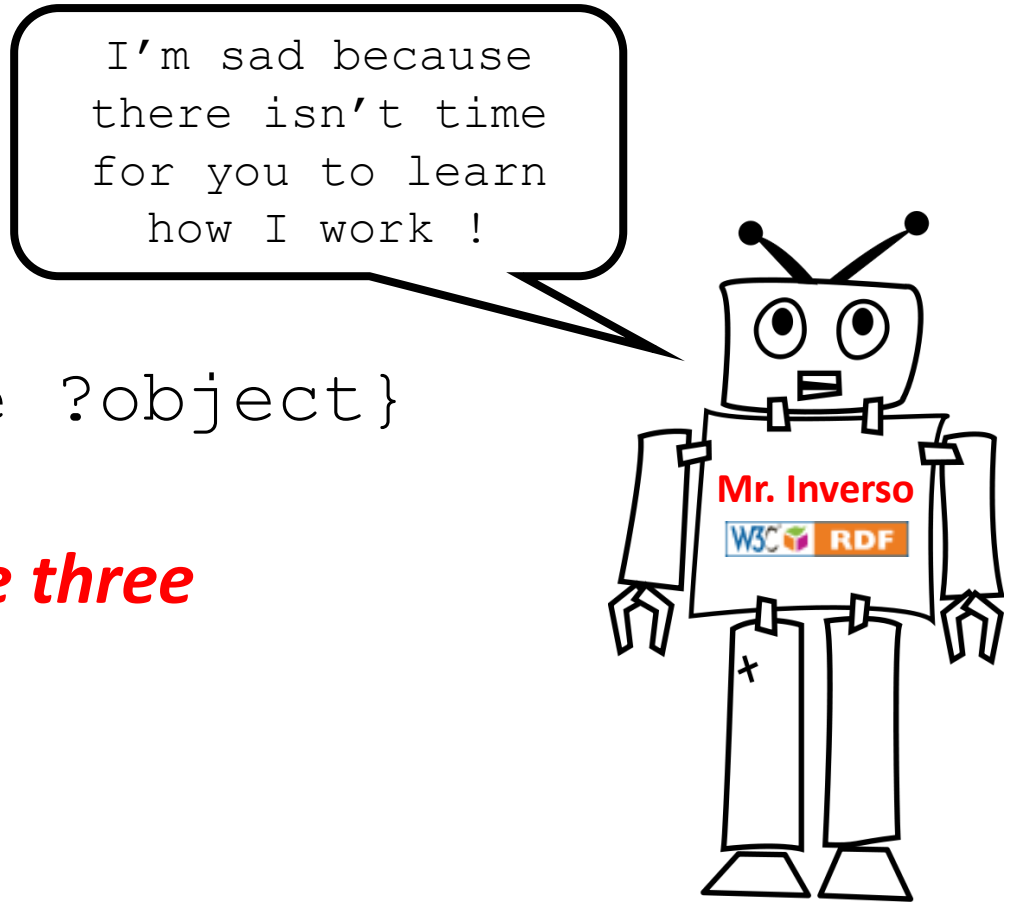
Change first line of query to

```
DESCRIBE ?imageURL WHERE {
```

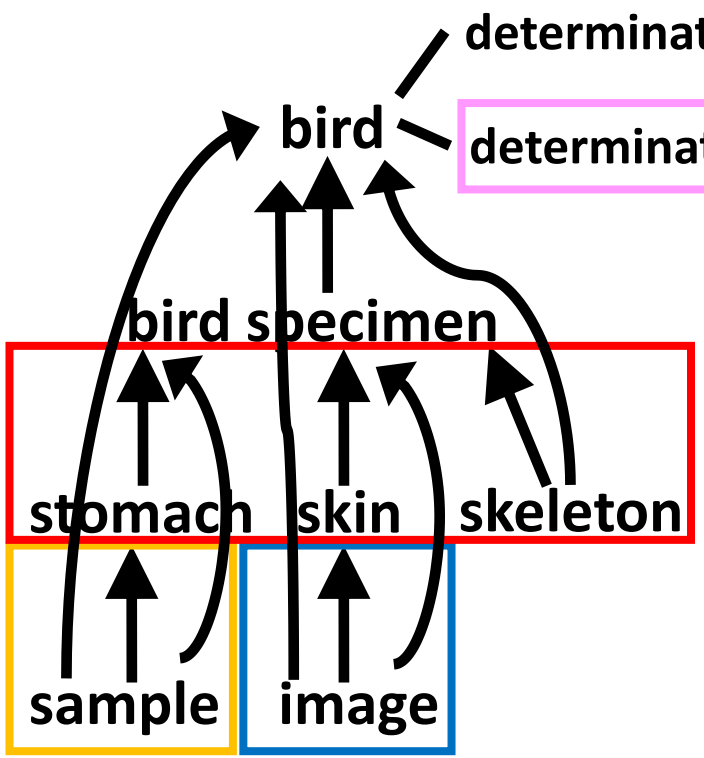
and select View: Images in the iSPARQL interface.

Building triples with the CONSTRUCT query

SPARQL can be used to **build RDF graphs** by creating triples using the **CONSTRUCT** query form.




```
CONSTRUCT {?subject ?predicate ?object}
WHERE {
(some triple pattern that binds terms to the three variables)
}
```



The real potential in RDF and SPARQL is that with properly structured RDF it is easy to query and discover information submitted **by other institutions.**

We can use SPARQL to answer competency questions:

1. Report new determinations for any organisms from whom images were derived and archived in CalPhotos.

```
SELECT DISTINCT ?resource ?determiner
?sciName
WHERE
{
 ?resource dwcuri:inCollection
<http://calphotos.berkeley.edu/void>.
 ?resource dsw:derivedFrom
?individual.
 ?individual dsw:hasIdentification
?id.
 ?id dwc:scientificName ?sciName.
 ?id dwc:identifiedBy ?determiner.
 ?id dwc:dateIdentified ?date.
 FILTER( ?date >= "2014-01-01")
 }
```

2. Report the URI and types of resources derived from a particular organism after a certain date.

```
SELECT DISTINCT ?resource ?type ?date
WHERE
 {
 ?resource dsw:derivedFrom
<http://arctos.database.museum/guid/MVZ
:Bird:21465#ind>.
 ?record dcterms:references ?resource.
 ?resource a ?type.
 ?record dcterms:modified ?date.
 FILTER( ?date >= xsd:dateTime("2015-
01-01T00:00:00"))
 }
```

Transitive *derivedFrom* properties link one-step derived resources.

Reasoning is used to infer *derivedFrom* relationships that directly link all resources to the bird

SPARQL Conclusions:

- SPARQL is a **standard** query language for RDF.
- SPARQL can be used to **find** resources in a triplestore (RDF database) using the SELECT query form.
- SPARQL can be used to **extract** a subset of triples from a graph using the DESCRIBE query form.
- The CONSTRUCT query form of SPARQL can be used to build graphs consisting of inferred triples (**simple reasoning tasks**) or triples to meet many other purposes.
- SPARQL queries can be built into user-friendly **applications** to allow for the **discovery** of new information from a community triplestore accessed through a SPARQL endpoint.

Acknowledgements

Steve's participation in the Semantics for Biodiversity Symposium was supported by the Research Coordination Network for the Genomic Standards Consortium (RCN4GSC, DBI-0840989) and the Scientific Observations Network (SONet , NSF #0753144, OCI-Interop).

RDF Primer video:
http://youtu.be/XAGifYBiXMY

URI of Beginners Guide to RDF:
http://code.google.com/p/tdwg-rdf/wiki/Beginners

The following slides were cut to fit the time limit but may be interesting to people who want to explore more on their own.

Building triples with the CONSTRUCT query

Example: look up GeoNames URI for a county

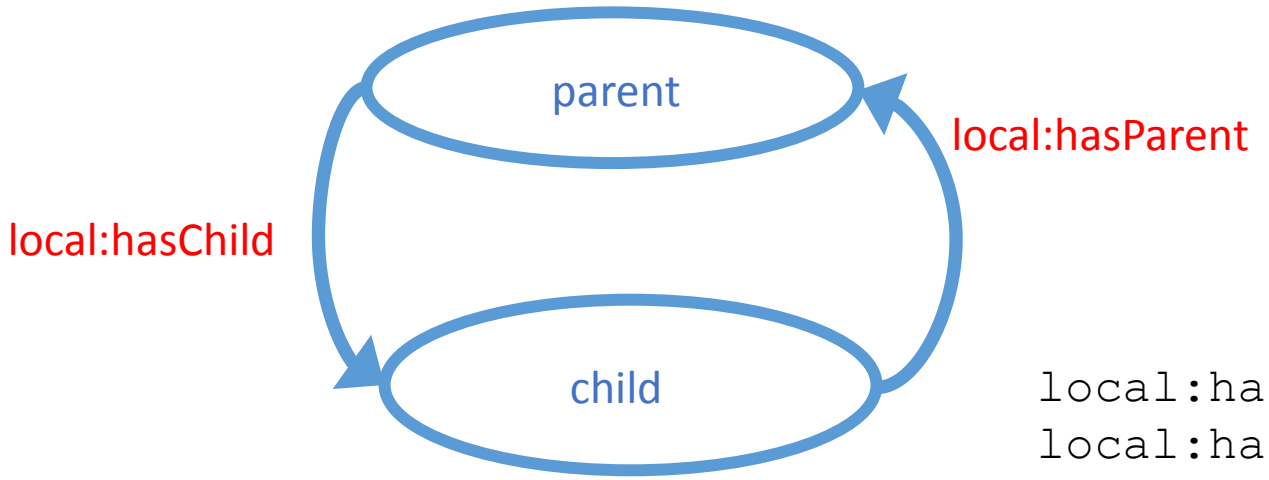
A database may contain only string values and the provider may wish to link to URIs.

```
CONSTRUCT {?location dwcuri:atDescribedPlace ?placeURI}
WHERE {
?location dwc:stateProvince ?stateString.
?location dwc:county ?countyString.
?placeURI gn:name ?countyString.
?placeURI gn:parentFeature ?stateURI.
?stateURI gn:name ?stateString.
}
```

The resulting triples can be added to the provider's database graph.

However, this would be an unreliable method if there were any variation in the strings.

CONSTRUCT queries and primitive forms of reasoning.



Given:

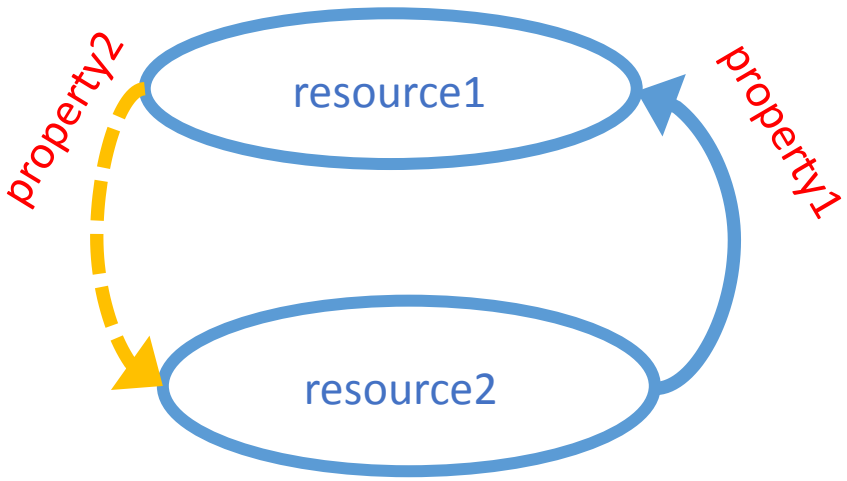
```
local:hasParent owl:inverseOf local:hasChild.
local:hasChild owl:inverseOf local:hasParent.
```

Example: infer a specific inverse property

```
CONSTRUCT {?child local:hasParent ?parent}
WHERE {
?parent local:hasChild ?child.
}
```

This only creates inferred triples for a single inverse relationship.

CONSTRUCT triples entailed by inverse properties



Example: infer inverse properties generally

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
CONSTRUCT {?resource1 ?property2 ?resource2}
WHERE {
 ?property1 owl:inverseOf ?property2.
 ?resource2 ?property1 ?resource1.
 OPTIONAL
       {
       ?R3 ?property2 ?resource2.
       FILTER(?R3 = ?resource1).
       }
 FILTER(!BOUND(?R3))
 }
```

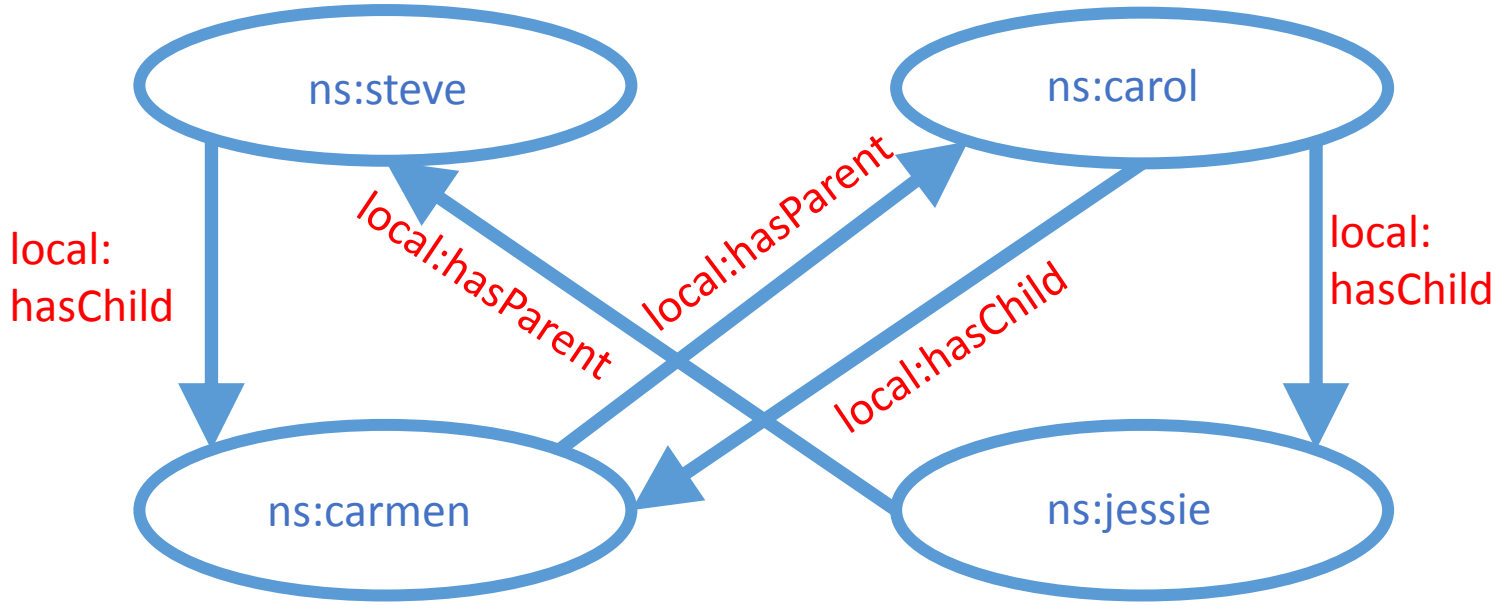
The filtering section at the end of the query prevents the construction of triples that already exist in the graph.

Example

Full query:

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX local:
<http://bioimages.vanderbilt.edu/rdf/local#>
CONSTRUCT {?resource1 ?property2 ?resource2}
FROM <http://tdwg-
rdf.googlecode.com/svn/trunk/example/inverse.rdf>
WHERE
 {
 ?property1 owl:inverseOf ?property2.
 ?resource2 ?property1 ?resource1.
 OPTIONAL
       {
       ?R3 ?property2 ?resource2.
       FILTER(?R3 = ?resource1).
       }
 FILTER(!BOUND(?R3))
 }
Limit 10
```

This query is restricted to the graphs listed after "FROM".

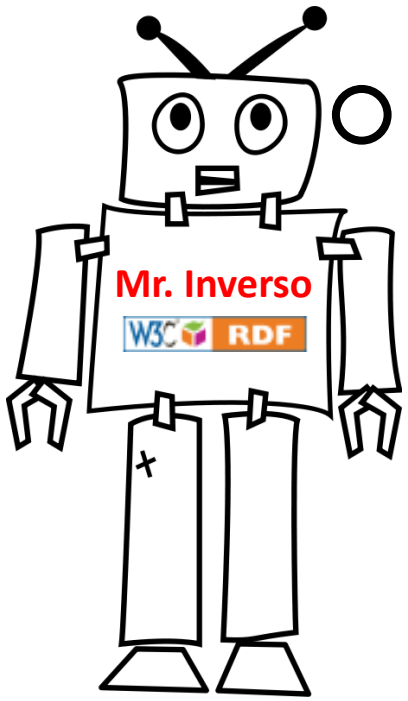


Resulting triples (translated from XML to Turtle):

```
ns:jessie
     local:hasParent ns:carol.
ns:carmen
     local:hasParent ns:steve.
ns:steve
     local:hasChild ns:jessie.
```

Hooray! We have programmed a reasoner by leveraging the capabilities of SPARQL !!!

```
CONSTRUCT {?resource1 ?property2 ?resource2}
WHERE
 {
 ?property1 owl:inverseOf ?property2.
 ?resource2 ?property1 ?resource1.
 OPTIONAL
       {
       ?R3 ?property2 ?resource2.
       FILTER(?R3 = ?resource1).
       }
 FILTER(!BOUND(?R3))
 }
```

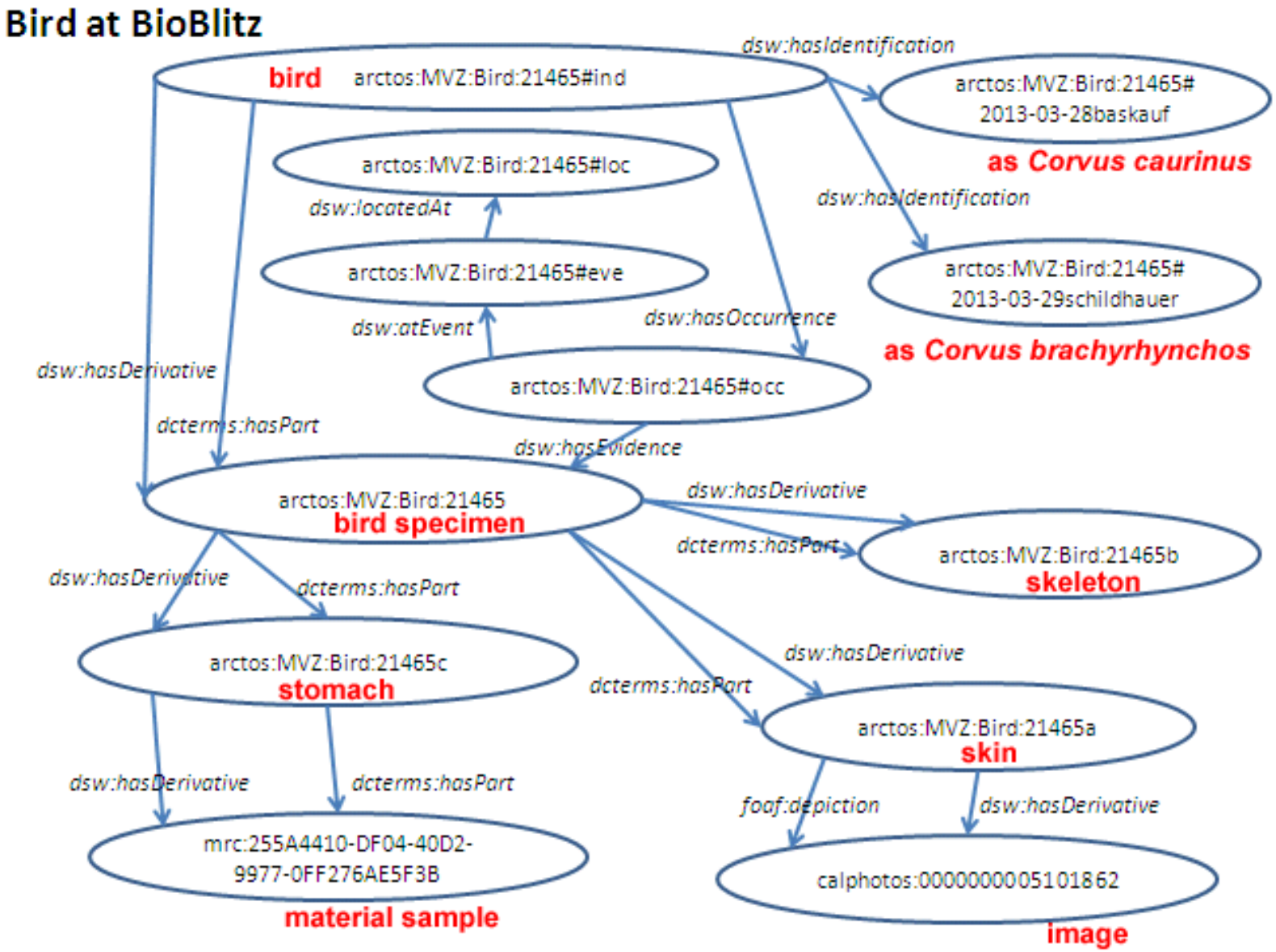


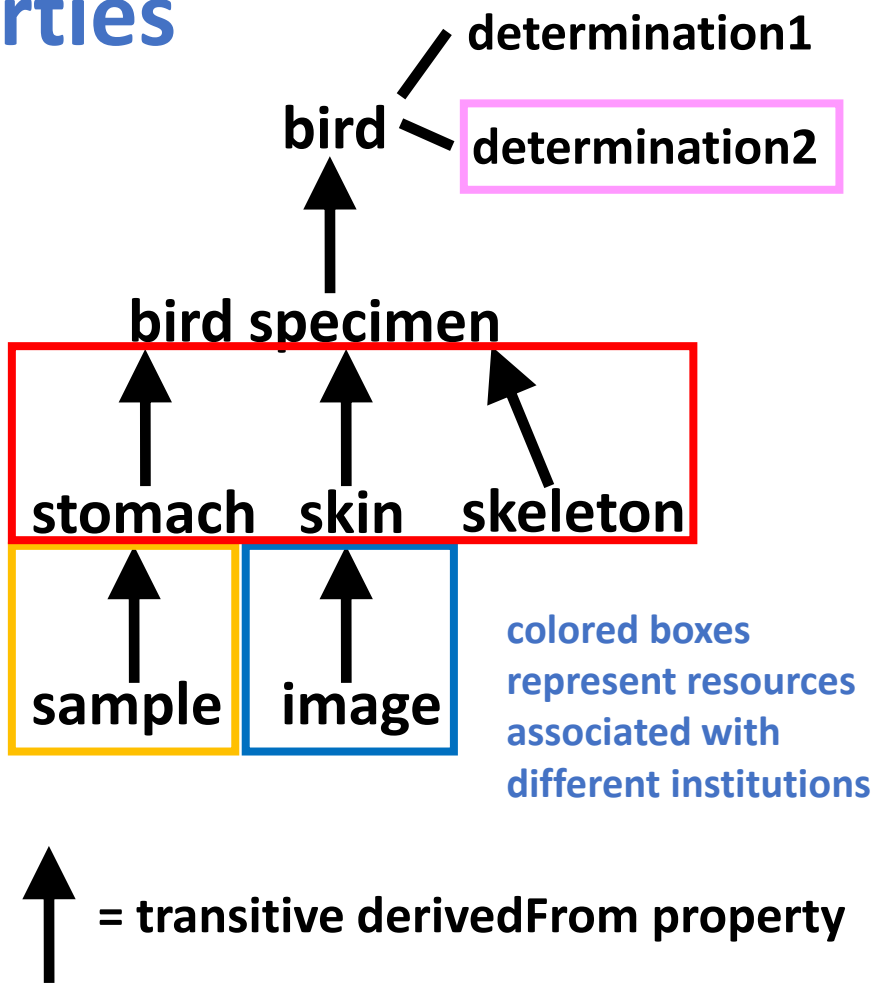
[Try this query yourself!]

Is it fair to say that? **Yes!** Recall that:

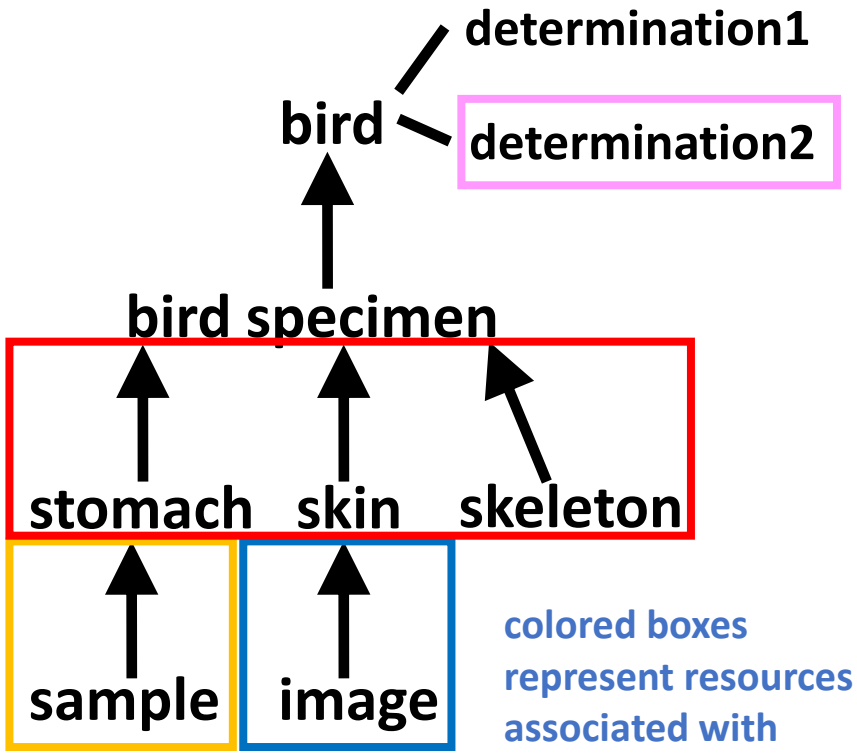
"A client is **not** required to apply all possible entailment rules ...
The choice of rules is placed on the programmer and/or user."

CONSTRUCT entailed by transitive properties





[Try this query yourself!]



colored boxes represent resources associated with different institutions

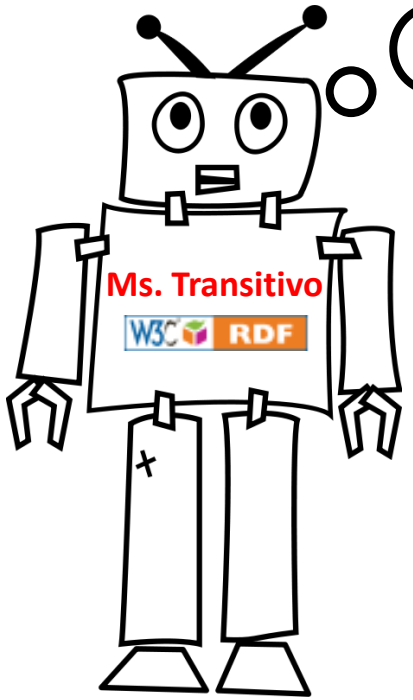
= transitive derivedFrom property

therefore:
bird specimen derivedFrom bird
stomach derivedFrom bird
sample derivedFrom bird
etc.

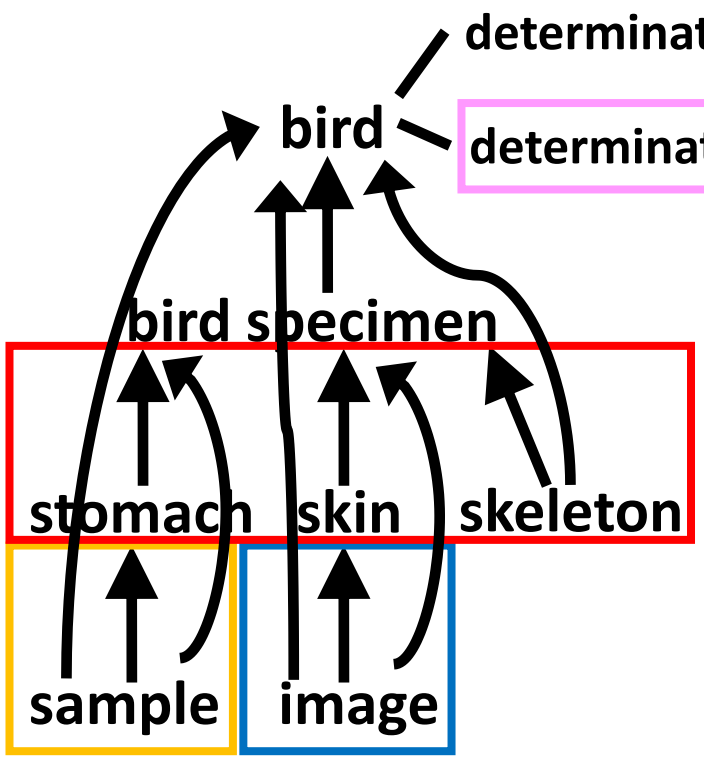
CONSTRUCT triples entailed by transitive properties (cont.)

```
CONSTRUCT {?R1 dsw:derivedFrom ?R3}
WHERE
 {
 ?R1 dsw:derivedFrom ?R2.
 ?R2 dsw:derivedFrom ?R3. OPTIONAL
      {
      ?trash dsw:derivedFrom ?R3.
      FILTER(?trash = ?R1)
      }
 FILTER(!bound(?trash))
}
```

repeat as necessary



Hooray! We have programmed a reasoner by leveraging the capabilities of SPARQL !!!



all derived resources have a direct *derivedFrom* relationship with the bird

Because each derived resource has a direct connection to the bird, it is easy to query and discover information submitted **by other institutions.**

We can answer competency questions:

1. Report new determinations for any organisms from whom images were derived and archived in CalPhotos.

```
SELECT DISTINCT ?resource ?determiner
?sciName
WHERE
{
 ?resource dwcuri:inCollection
<http://calphotos.berkeley.edu/void>.
 ?resource dsw:derivedFrom
?individual.
 ?individual dsw:hasIdentification
?id.
 ?id dwc:scientificName ?sciName.
 ?id dwc:identifiedBy ?determiner.
 ?id dwc:dateIdentified ?date.
 FILTER( ?date >= "2014-01-01")
 }
```

2. Report the URI and types of resources derived from a particular organism after a certain date.

```
SELECT DISTINCT ?resource ?type ?date
WHERE
 {
 ?resource dsw:derivedFrom
<http://arctos.database.museum/guid/MVZ
:Bird:21465#ind>.
 ?record dcterms:references ?resource.
 ?resource a ?type.
 ?record dcterms:modified ?date.
 FILTER( ?date >= xsd:dateTime("2015-
01-01T00:00:00"))
 }
```