ACTIVE
Knowledge-Powered Enterprise

Semantics:
Linked Data and Services
RDF(S), SPARQL, OWL, SKOS, HTTP

http://active-project.eu

Dr. Barry Norton,
Karlsruhe Institute of Technology (KIT)
barry.norton@kit.edu

BT
"Jožef Stefan" Institute
KIT Karlsruhe Institute of Technology
iSOCO
STI · INNSBRUCK
Microsoft Innovation Center Europe
EURESCOM
kea-pro
fit
accenture
cadence

Overview

- Introduction to Semantic Technologies
- RDF & RDF Schema
- OWL & SKOS Overview
- SPARQL
- HTTP
- Linked Data
- Linked Services

(Very Short) Introduction to Semantic Technologies

The World Wide Web

The 1990s saw the birth of the World Wide Web (WWW).

The initial proposal for a “hypertext project” was made by Sir Tim Berners-Lee, then a physicist at the CERN laboratory in Geneva, Switzerland, in March 1989.

In 1990, the first Web server went online and the first Web browser was released. By the end of 1991, the Web was publicly accessible.

The success of the World Wide Web was based on taking an existing idea: hypertext, and tying it to a global network: the Internet.

A Web for Humans

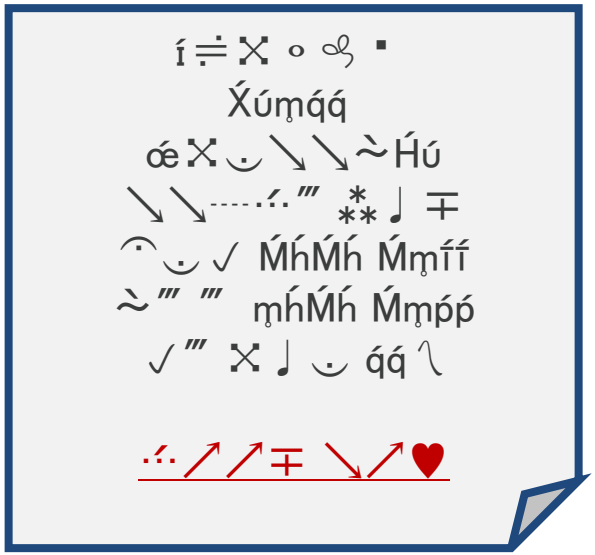
The Web has become a massive source of information for humans who are able to understand and interpret the content – and draw conclusions by bringing together information from different cues.

Flight AI288
Vienna-Innsbruck
Dep: **1st Jan 12:00**
Arr: **2nd Jan 12:55**
Price: **88€**

Book now

A human can look at this information and understand what it means...

... but for a machine it could just as well look like this



Searching the Human Web

As the Web grew, the problem of finding information on it became much more complex.

Early attempts to help (human) users navigate the Web were in the style of directories. This was the original model of Yahoo!

Very soon, however, it became infeasible to maintain such a directory as categories are inflexible (no changes by third parties)

Furthermore the contents of each category became too large to be able to find some particular site.

Yahoo! in 1997

Searching the Human Web

Attempts to provide machines with algorithms for finding information on the Web have focused on indexing the text on web pages and matching these with user-entered keywords.

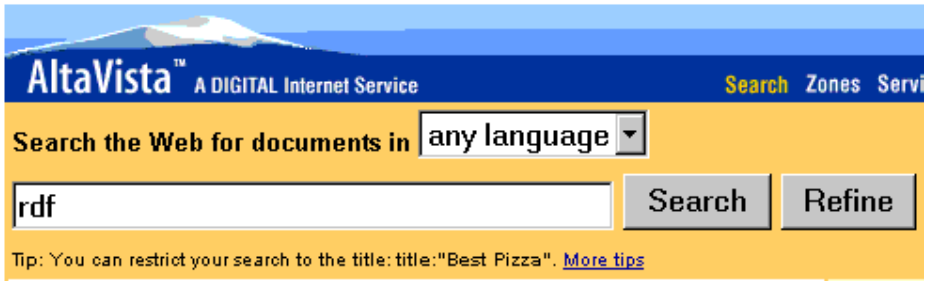
This is more flexible as searches are not bound to pre-defined categories. Keywords are ambiguous, however:

Java (coffee) vs.

Java (language) vs.

Java (island).

Synonyms are also missed.

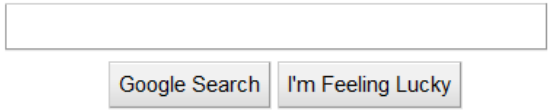


AltaVista in 1997

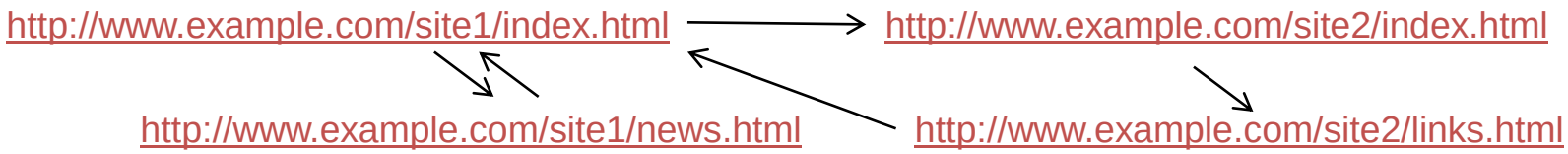
The problem of ranking within search results also predominates.

Google/PageRank

The World's most popular search engine, Google, achieved its better search results through a clever assumption: Web pages which are linked to more often from other web pages are better sources of information



This underlies the PageRank algorithm, which views the Web as a graph of documents, connected via their links:

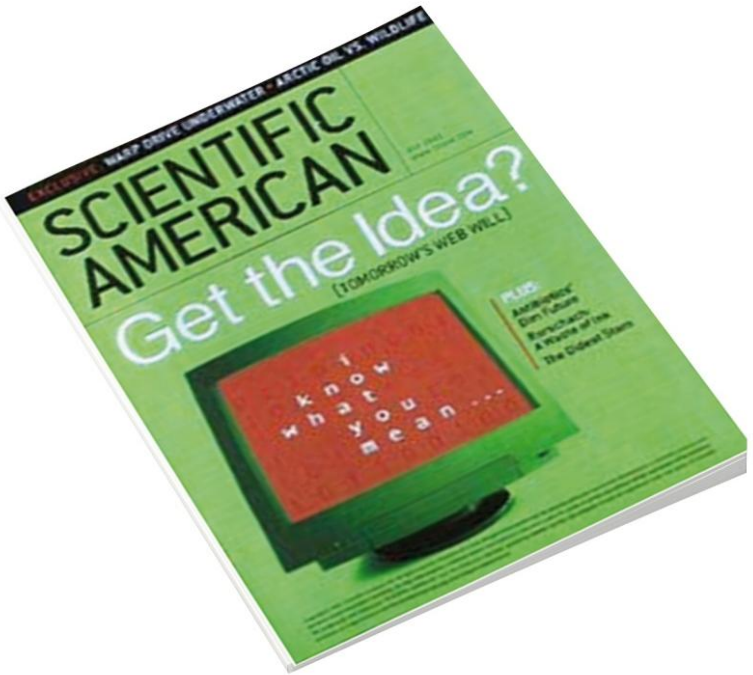


Note that documents are still understood as a collection (‘bag’) of keywords, and the links completely ambiguous (but always interpreted as a recommendation).

The Semantic Web

The vision of what was termed the “Semantic Web” first came to public attention through an article in Scientific American in May 2001.

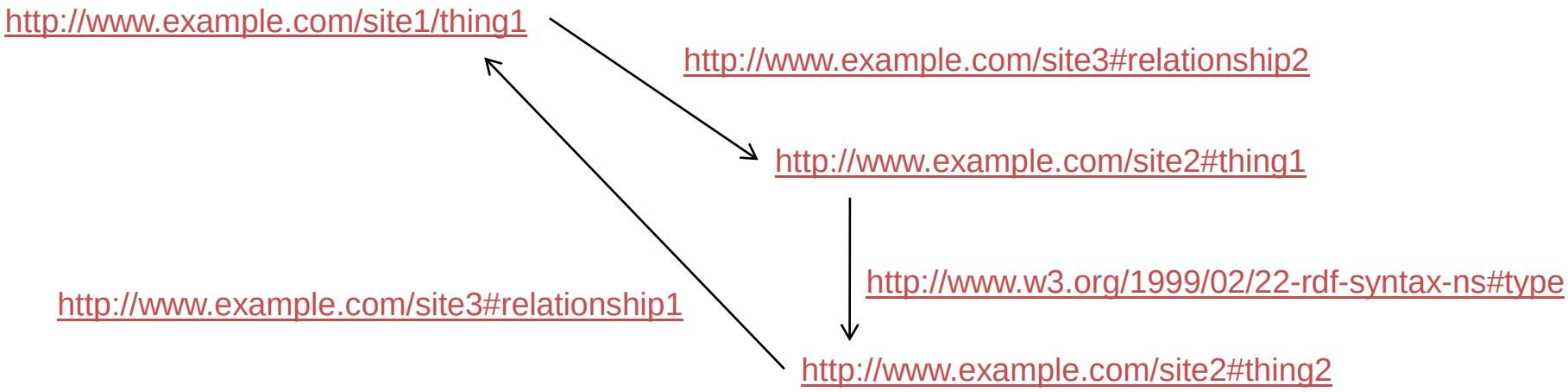
“The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation.”



Fundamental Semantics

The fundamental means to add semantics (= explicit meaning)

to the Web is by disambiguating the links between URI-identified resources (including Web documents, identified by URLs)



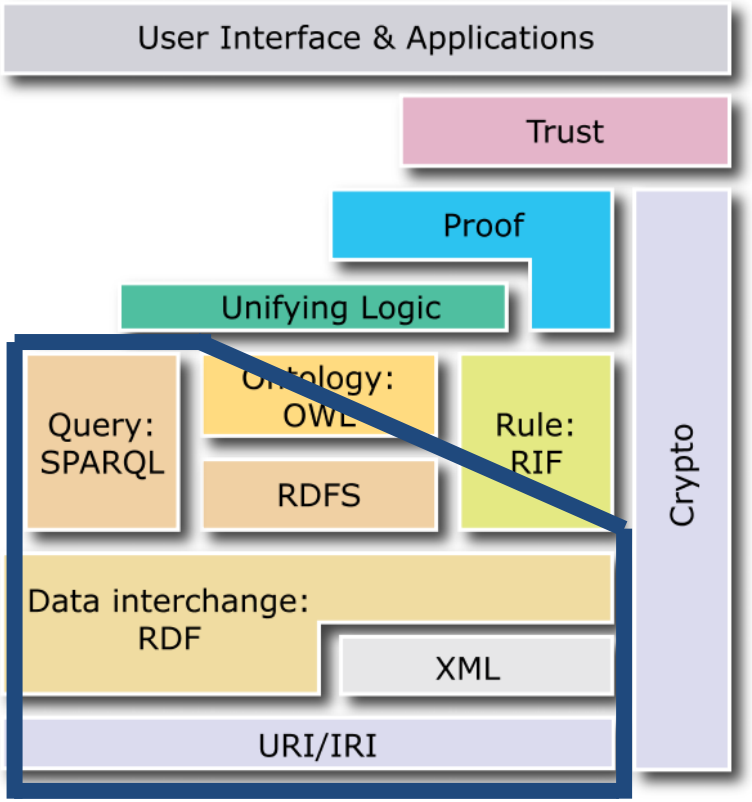
This is basis of the Resource Description Framework (RDF).

The meaning of a link is given by (reused) URIs, called predicates, only a few of which (e.g., 'type') are from RDF itself.

Semantic Technologies

We can define what *inferences* should be drawn from the relationship of resources via predicates with languages such as:

- RDF Schema (RDFS),
  detailed later
- Web Ontology Language (OWL) and Simple Knowledge Organization System (SKOS),
  sketched later
- Rules defined according to the Rule Interchange Format (RIF),
  not covered here



- We can query for inter-related resources using SPARQL,
  detailed later.

URIs

Web-based Uniform Resource Locators (URLs), prefixed `http://` and `https://`, identify 'documents on the Web', that is accessible using the Hypertext Transfer Protocol (HTTP) from the named Internet server

Web URLs are a subset of general URLs; all specify a means to access a representation of the identified resource, e.g.:

`ftp://` URLs specify that content accessible via File Transfer Protocol

`file://` URLs specify that content is accessible on a file system

Uniform Resource Names (URNs) provide no such means to access a representation, but may identify abstract or physical resources:

`urn:isbn:0-470-02596-4` specifies a book, but not how/where to find it (or information about it, being the whole abstract edition)

Uniform Resource Identifiers (URIs) are made up of both URLs and URNs.

XML

eXtensible Mark-up Language specifies a means to represent structured context that generalises HTML (where mark-up concerns only presentation in a browser):

HTML

```
<html>
  <body>
    <p><b>Flight AI288</b></p>
<table>
 <tr><td>Dep.</td>
        <td>1st Jan 12:00</td></tr>
 <tr><td>Arr.</td>
        <td>1st Jan 12:55</td></tr>
</table>
<a href="/book.html>
                  book now</a>
```

XML

```
<flight>Flight AI288
  <from>Vienna</from>
  <to>Innsbruck</to>
  <dep>2012-01-01T12:00+02:00</dep>
  <arr>2012-01-01T12:55+01:00</arr>
  <price currency="EUR">88</price>
</flight>
```

XML Pros and Cons

- XML specifies a reusable means to *exchange structured data* and has a great deal of programming language support
- XML has a schema language that *constrains* valid representations for fixed elements
- XML allows the representation of the semantics of data, but these semantics are *implicit* and *application-specific*:
  - No particular inference can be drawn from nesting of elements and placement of attributes
  - There is no de facto way to identify, or to add further information outside the schema, to elements within a description
  - There is no general way to extend or combine schemas

RDF can be represented in XML, but overcomes these issues

Namespaces

- XML relies, in its use of URIs, a mechanism that will be reused in RDF: the idea of namespaces.

- Since (especially HTTP) URIs can be hierarchical, a prefix to the namespace is given a short name (within a document), e.g.:
  - `xs=http://www.w3.org/2001/XMLSchema#`

- Thereafter, where a URI is expected but a string of the form `ns:name` is encountered, the `ns` part is expanded to form a full URI, e.g.:
  - `xs:dateTime` becomes `http://www.w3.org/2001/XMLSchema#dateTime`

RDF

RDF provides a graph model where statements can be made about relationships between URI-identified resources in an extensible way, and allowing for the provision of inference by generic mechanisms.

```
<http://example.com/flights/AI288>
<http://example.com/vocab#from>
<http://dbpedia.org/resource/Vienna>

<http://example.com/flights/AI288>
<http://example.com/vocab#departs>
"01-01-2009T12:00"
```

RDFS

RDF Schema permits vocabularies (simple ontologies; models about concepts and their properties) to be defined, and thereby inference about the classification of resources.

Inference is the derivation of new triples from existing ones.

Schema

```
vocab:CommercialFlight
rdfs:subClassOf
vocab:Flight
```

```
vocab:from
rdfs:range
<http://dbpedia.org/ontology/City>
```

Existing fact

```
flights:AI288 rdf:type
vocab:CommercialFlight
```

```
<http://example.com/flights/AI288>
vocab:from
<http://dbpedia.org/resource/Vienna>
```

Inferred fact

```
flights:AI288 rdf:type
vocab:Flight
```

```
<http://dbpedia.org/resource/Vienna>
rdf:type
<http://dbpedia.org/ontology/City>
```

OWL Inference

The Web Ontology Language (OWL) first adds more powerful constructs, allowing further inference over RDF-based models.

We shall consider some OWL constructs in the context of Linked Data.

Ontology

```
<http://www.geonames.org/ontology#parentFeature>
rdf:type owl:TransitiveProperty
```

Existing facts

```
                owl:sameAs          http://dbpedia.org/resource/Vienna_International_Airport
http://sws.geonames.org/2761335/
                    ↓ http://www.geonames.org/ontology#parentFeature
http://sws.geonames.org/2770542/
                    ↓ http://www.geonames.org/ontology#parentFeature
http://sws.geonames.org/2782113/
                owl:sameAs          http://dbpedia.org/resource/Austria
```

Inferred fact

```
<http://dbpedia.org/resource/Vienna_International_Airport>
        <http://www.geonames.org/ontology#parentFeature>
        <http://dbpedia.org/resource/Austria>
```

OWL Consistency

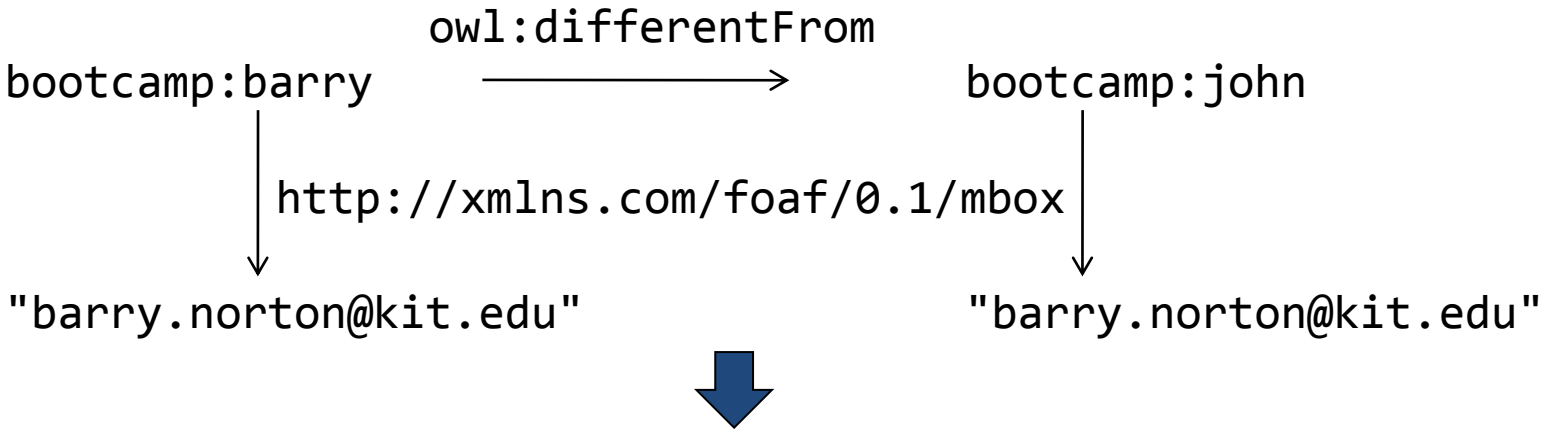
Unlike RDFS, OWL does not simply infer new triples over RDF models, but also adds a notion of *consistency* and axioms that *constrain* models.

This and other features of OWL will be considered in the STP course.

Ontology

```
<http://xmlns.com/foaf/0.1/mbox>
rdf:type owl:InverseFunctionalProperty
```

Existing facts



Inconsistency!

SKOS

OWL is seen as both too complex and too limited for the looser kind of knowledge organisation associated with thesauri, subject hierarchies, etc.

In particular *subsumption* may not be the most appropriate definition of subclassing, and *transitivity* may not hold

The Simple Knowledge Organization System (SKOS) introduces `skos:Concept` (different from owl:Class, which itself is distinguished from rdfs:Class).

Concepts are organised hierarchically by:

- `skos:narrower` and `skos:broader`, distinct from
  - `skos:narrowerTransitive` and `skos:broaderTransitive`

SPARQL

The final language that this introduction will cover is SPARQL, a language that allows querying over RDF-based data, according to matches between the data graph and patterns.

Example: *Is there a flight from Kotoka Airport to somewhere else in Ghana for a price for under 1000 Cedi?*

```
SELECT ?flight
WHERE
  ?flight vocab:from
      <http://dbpedia.org/resource/Kotoka_International_Airport> .
  ?flight vocab:to ?place .
  ?place geo:parentFeature <http://dbpedia.org/resource/Ghana> .
  ?flight vocab:price ?price ;
      vocab:currency <http://dbpedia.org/resource/Ghanaian_cedi> .
FILTER (?price < 1000)
```

SPARQL Construct

As well as retrieving relations (SELECT queries) a SPARQL query can CONSTRUCT a new graph based on the variable bindings in the WHERE clause.

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

CONSTRUCT
  { ?x foaf:name { fn:concat(?gn, " ", ?sn) } }

WHERE
  { foaf:givenname ?gn ; foaf:surname ?sn . }
```

RIF

Rules allow matching schemes over RDF-based data. In some forms they may allow new inferences to be defined, outside those implicit in given OWL axioms.

Among the differences between such (production) rules and SPARQL construct queries are the direct manipulation of the knowledge base and the ‘run to completion’ nature of rules.

```
if { ?x foaf:firstName ?first;
        foaf:surname ?last }
then { ?x foaf:family_name ?last;
          foaf:givenname ?first;
          foaf:name func:string-join(?first " " ?last) }
```

Resource Description Framework (RDF) in More Detail

Resource Description Framework

- RDF is the basis layer of the Semantic Web stack ("layer cake"), in which all statements are asserted in 3-tuples:
  - **Subject** – an identified (by URI) resource
  - **Object** – a resource or literal to which the subject is related
  - **Predicate** – a (URI) identified reused specification of the relationship

```
<http://www.aifb.kit.edu/id/Barry_Norton>
<http://www.w3.org/2002/07/owl#sameAs>
<http://semanticweb.org/id/Barry_Norton>
```

Resource Description Framework

- RDF is the basis layer of the Semantic Web stack ("layer cake"), in which all statements are asserted in 3-tuples:
  - **Subject** – an identified (by URI) resource
  - **Object** – a resource *or literal* to which the subject is related
  - **Predicate** – a (URI) identified reused specification of the relationship

```
<http://www.aifb.kit.edu/id/Barry_Norton>
<http://xmlns.com/foaf/0.1/firstName>
"Barry"
```

Turtle

- While normative syntax for RDF is in XML, Turtle (a subset of n3 is more readable)
- Since many URIs share same basis we introduce prefixes:

```
@prefix owl:<http://www.w3.org/2002/07/owl#>.
@prefix foaf:<http://xmlns.com/foaf/0.1/>.
```

- A simple shorthand is for class membership:

```
<http://www.aifb.kit.edu/id/Barry_Norton>
        a foaf:Person.
```

Turtle

- When multiple statements apply to same subject, we can abbreviate:

```
<http://www.aifb.kit.edu/id/Barry_Norton>
   foaf:firstName "Barry";
   owl:sameAs <http://semanticweb.org/id/Barry_Norton>,
         <http://people.kmi.open.ac.uk/barry/foaf.rdf>.
```

same subject

same subject and predicate

Turtle

- Turtle also allows datatypes and language tags for literals:

```
<http://www.aifb.kit.edu/id/Barry_Norton>
  foaf:age "33"^^xsd:int;
  foaf:member <http://http://www.aifb.kit.edu>.

<http://http://www.aifb.kit.edu>
  rdfs:label
      "Institut für Angewandte Informatik und Formale
       Beschreibungsverfahren"@de,
      "Institute of Applied Informatics and Formal
       Descriptions Methods"@en.
```

Turtle

- Turtle also has two shorthands for

  anonymous = “blank nodes”:

```
<http://www.aifb.kit.edu/id/Barry_Norton>
 foaf:knows _:a1.
_:a1 foaf:firstName "Reto";
     foaf:lastName "Krummenacher";
     foaf:knows [foaf:firstName "Adrian";
                 foaf:lastName "Marte"].
```

Turtle

- And for lists:

```
<http://www.aifb.kit.edu/id/Barry_Norton>
  food:preferences (food:noodles food:rice food:millet).
```

```
<http://www.aifb.kit.edu/id/Barry_Norton>
  food:preferences [rdf:first food:noodles;
                    rdf:rest [rdf:first food:rice;
                              rdf:rest
                                [rdf:first food:millet;
                                 rdf:rest rdf:nil]]].
```

RDF Schema

RDF Schema

- RDF Schema introduces
  - particular resources and predicates with some (limited) inference, especially:
    - `rdfs:Resource`
    - `rdfs:Literal`
    - `rdfs:Class`, `rdfs:subClassOf`
    - `rdfs:Property`, `rdfs:subPropertyOf`
    - `rdfs:range`, `rdfs:domain`
  - some predicates with no inference:
    - `rdfs:comment`
    - `rdfs:label`
    - `rdfs:seeAlso`
    - `rdfs:isDefinedBy`

RDFS Inference

- Recall:

Schema

```
vocab:CommercialFlight
rdfs:subClassOf
vocab:Flight.
```

```
vocab:from
rdfs:range
<http://dbpedia.org/ontology/City>.
```

Existing fact

```
flights:AI288 rdf:type
vocab:CommercialFlight.
```

```
<http://example.com/flights/GH134>
vocab:from
<http://dbpedia.org/resource/Accra>.
```

Inferred fact

```
flights:AI288 rdf:type
vocab:Flight.
```

```
<http://dbpedia.org/resource/Accra>
rdf:type
<http://dbpedia.org/ontology/City>.
```

RDFS Inference

- Also:

Schema

```
:C1
rdfs:subClassOf :C2.

:C2
rdfs:subClassOf :C3.
```

Inferred facts

```
:C1
rdfs:subClassOf :C3.
```

```
:P1 rdfs:subPropertyOf :P2.

:i1 :P1 :i2.
```

```
:i1 :P2 :i2.

:P1 a rdfs:Property.
:P2 a rdfs:Property.
```

OWL & SKOS

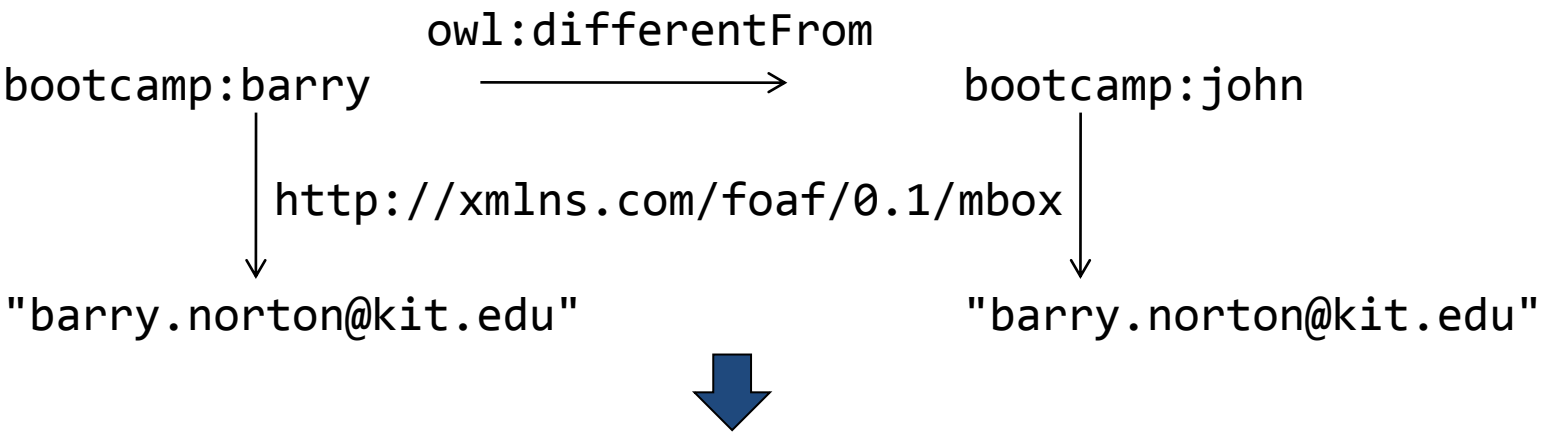
OWL Inference

- Recall

Ontology

```
<http://xmlns.com/foaf/0.1/mbox>
rdf:type owl:InverseFunctionalProperty.
```

Existing facts



Inconsistency!

Why is this?

OWL Inference

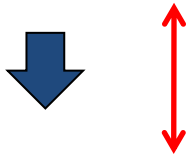
- Explanation:

Ontology

```
foaf:mbox a owl:InverseFunctionalProperty.
```

Existing facts

```
bootcamp:barry foaf:mbox "barry.norton@kit.edu".
bootcamp:john foaf:mbox "barry.norton@kit.edu".
bootcamp:barry owl:differentFrom bootcamp:john.
```



Inconsistency!

Inferred fact

```
bootcamp:barry owl:sameAs bootcamp:john.
```

OWL Inference

- What are the consequences of `sameAs`?

Ontology `:i1 owl:sameAs :i2.`

Existing facts `:i1 a :C1.` `:i2 :p2 :i3.`

Inferred facts `:i2 a :C1.` `:i1 :p2 :i3.`

OWL Property Axioms

OWL properties can be declared: `functional`, `inverseFunctional`, `transitive`, `reflexive`, `symmetric` (and mutually `inverse`)

Recall:

Ontology

```
<http://www.geonames.org/ontology#parentFeature>
rdf:type owl:TransitiveProperty
```

Existing facts

```
                owl:sameAs        http://dbpedia.org/resource/Vienna_International_Airport
http://sws.geonames.org/2761335/
                ↓ http://www.geonames.org/ontology#parentFeature
http://sws.geonames.org/2770542/
                ↓ http://www.geonames.org/ontology#parentFeature
http://sws.geonames.org/2782113/
                owl:sameAs        http://dbpedia.org/resource/Austria
```

Inferred fact

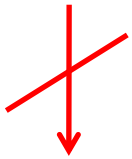
```
<http://dbpedia.org/resource/Vienna_International_Airport>
        <http://www.geonames.org/ontology#parentFeature>
        <http://dbpedia.org/resource/Austria>
```

OWL Property Cardinalities

OWL properties can also be constrained with respect to cardinality – `cardinality, minCardinality, maxCardinality`

Note:

Ontology	`:p1 owl:minCardinality "2".`	`:p2 owl:maxCardinality "1".`
Existing facts	`:i1 :p1 :i1.`	`:i2 :p2 :i3.` `:i2 :p2 :i4.`



Inconsistency!

Inconsistency!

OWL Class Axioms

- OWL classes:
  - can be declared:
    - `equivalent, disjoint`
  - can be defined
    - by property values:
      - `hasValue, someValuesFrom, allValuesFrom`
    - by *construction* from existing classes:
      - `complementOf, intersectionOf, unionOf`

Linked Data tends to stick to `sameAs` and property axioms

SKOS

The following properties from SKOS tend to be used in Linked Data:

- `skos:prefLabel` – a subpropertry of `rdfs:label` that indicates the preferred term for a concept, versus
- `skos:altLabel` – another subpropertry of `rdfs:label` that indicates alternatives labels

- `skos:notation` – often used to give values in a non-URI identification scheme, where the datatype tag is used to give a URI for the scheme (see later ICAO and IATA airport codes)

Other (non-W3C) vocabularies are often used, in particular be aware of the **Dublin Core**, a common set of metadata properties that preceded RDF(S), but was updated to use it

SPARQL

SPARQL Overview

- SPARQL is often introduced by analogy to SQL:
  - SQL acts over relational databases
  - SPARQL based on **graph patterns**, rather than relations but shares syntax:

```
SELECT ?v1 ?v2
FROM ...
WHERE
```

  for SELECT queries, however, are only one of four query types:

  - SELECT
    - CONSTRUCT
      - ASK
        - DESCRIBE (future versions will also include UPDATE)

SPARQL Triple Patterns

- All three query types are based on graph patterns, which are {}-bracketed collections of triple patterns
- SPARQL uses Turtle for triple definition but allows also variables, prefixed with ?

```
{<http://www.aifb.kit.edu/id/Barry_Norton> foaf:knows ?x}
```

Matched by

```
<http://www.aifb.kit.edu/id/Barry_Norton>
        foaf:knows <http://www.aifb.kit.edu/web/Rudi_Studer>.
```

And

```
<http://www.aifb.kit.edu/id/Barry_Norton>
        foaf:knows [foaf:firstName "Reto" ...]
```

SPARQL Triple Patterns

- All three query types are based on graph patterns, which are {}-bracketed collections of triple patterns
- SPARQL uses Turtle for triple definition but allows also variables, prefixed with ?
- Graph patterns may be optional

```
{<http://www.aifb.kit.edu/id/Barry_Norton> foaf:knows ?x.
 OPTIONAL {?x foaf:depiction ?z}}
```

Matched by

```
<http://www.aifb.kit.edu/id/Barry_Norton>
        foaf:knows <http://www.aifb.kit.edu/web/Rudi_Studer>.
<http://www.aifb.kit.edu/web/Rudi_Studer> foaf:depiction
<http://www.aifb.kit.edu/images/0/0c/Studer.jpeg>.
```

And

```
<http://www.aifb.kit.edu/id/Barry_Norton>
          foaf:knows [foaf:firstName "Reto" ...]
```

SPARQL Triple Patterns

- All three query types are based on graph patterns, which are {}-bracketed collections of triple patterns
- SPARQL uses Turtle for triple definition but allows also variables, prefixed with ?
- Graph patterns may be optional
- Graph patterns may be combined by disjuntion

(‘./;/,’ = conjunction)

```
{<http://www.aifb.kit.edu/id/Barry_Norton> foaf:knows ?x.
{{?x foaf:firstName ?y} UNION {?x foaf:givenName ?y}}}
```

Matched twice by

```
<http://www.aifb.kit.edu/id/Barry_Norton>
  foaf:knows [foaf:firstName "Reto"], [foaf:givenName "Adrian"].
```

SPARQL ASK

- ASK queries are the simplest and merely evaluate whether there is *any* match for a graph pattern over a given dataset

```
ASK {<http://www.aifb.kit.edu/id/Barry_Norton> foaf:knows ?x.
{{?x foaf:firstName ?y} UNION {?x foaf:givenName ?y}}}
```

Over

```
<http://www.aifb.kit.edu/id/Barry_Norton>
  foaf:knows [foaf:firstName "Reto"], [foaf:givenName "Adrian"].
```

True

SPARQL SELECT

- SELECT queries, as in SQL, give (sub)set(s) of variable bindings under which the graph patterns are matched

```
SELECT *
WHERE {<http://www.aifb.kit.edu/id/Barry_Norton> foaf:knows ?x.
       ?x foaf:hasFirstName ?y}
```

Over

```
<http://www.aifb.kit.edu/id/Barry_Norton>
  foaf:knows
    <http://www.aifb.kit.edu/id/Rudi_Studer>,
    [foaf:firstName "Reto"].
<http://www.aifb.kit.edu/id/Rudi_Studer> foaf:hasFirstName "Rudi".
```

?x	?y
<http://www.aifb.kit.edu/id/Rudi_Studer>	"Rudi"
_:id (auto-generated and not reusable)	"Reto"

SPARQL SELECT

- SELECT queries, as in SQL, give (sub)set(s) of variable bindings under which the graph patterns are matched
- Note that we can use named graphs to include existing knowledge

```
SELECT *
FROM <http://www.aifb.kit.edu/id/Rudi_Studer>
WHERE {<http://www.aifb.kit.edu/id/Barry_Norton> foaf:knows
       []}
```

Over

```
<http://www.aifb.kit.edu/id/Barry_Norton>
  foaf:knows
    <http://www.aifb.kit.edu/id/Rudi_Studer>,
    [foaf:firstName "Reto"].
<http://www.aifb.kit.edu/id/Rudi_Studer> foaf:hasFirstName "Rudi".
```

(Further syntactical constructs – **NAMED**, **GRAPH**, etc. – allow more explicit and dynamic use of 'named graphs')

SPARQL CONSTRUCT

- CONSTRUCT query makes a new graph based on
  - the whole set of matches and the variable bindings in each
  - a second graph pattern which shares some of the same variables

```
CONSTRUCT {<http://www.linkedopenservices.org/blog/> dc:editor ?x}
FROM ...
WHERE {<http://www.aifb.kit.edu/id/Barry_Norton> foaf:knows ?x .
       ?x foaf:currentProject <http://www.linkedopenservices.org/>}
```

```
<http://www.linkedopenservices.org/blog/> dc:editor ..., ...
```

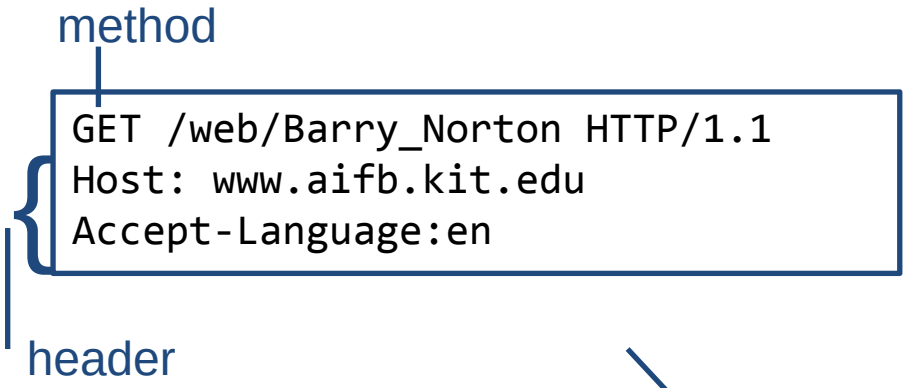
HTTP

HTTP Overview

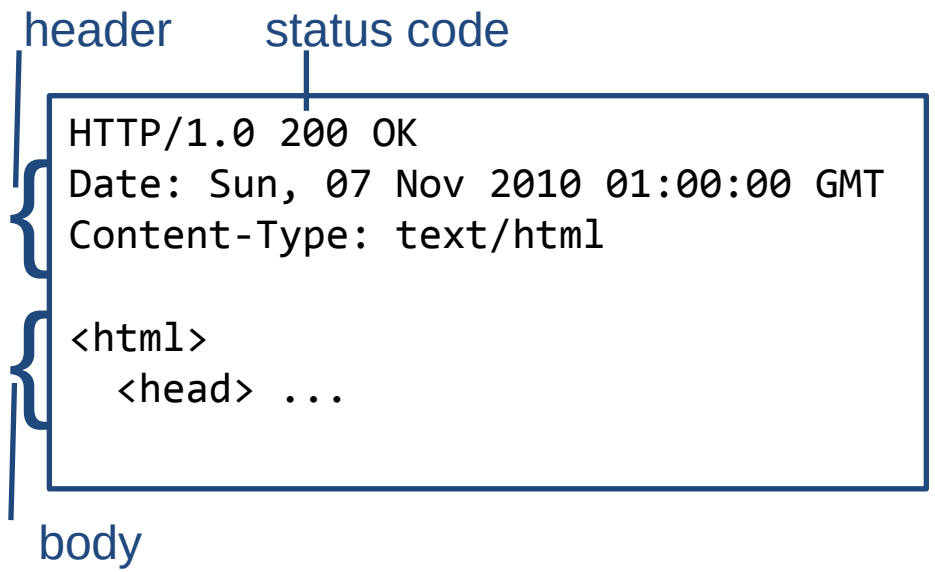
- HTTP, by which all documents on the WWW are served, is a client server protocol.
- Every interaction based on:
  - Request
    - Method
      - GET
      - PUT
      - POST
      - PATCH
      - DELETE (+ OPTIONS, HEADER, TRACE, CONNECT)
    - URL
    - Header
    - [Optional] Body (with POST, PUT, PATCH)
  - Reponse
    - Reponse code (integer)
    - Header
    - [Optional] Body

HTTP GET Example

- Retrieval example:



- Can negotiate on (human) language, but also...



HTTP Coneg Example

- Content negotiation (coneg) example:

```
GET /id/Barry_Norton HTTP/1.1
Host: www.aifb.kit.edu
Accept:text/html
```

```
HTTP/1.0 302 Moved Temporarily
Date: Sun, 07 Nov 2010 00:30:00 GMT
Location:
http://www.aifb.kit.edu/web/Barry_Norton
```

```
GET /id/Barry_Norton HTTP/1.1
Host: www.aifb.kit.edu
Accept:application/rdf+xml
```

```
HTTP/1.0 302 Moved Temporarily
Date: Sun, 07 Nov 2010 00:45:00 GMT
Location:
http://www.aifb.kit.edu/portal/index.php?
title=Spezial:Exportiere_RDF/Barry_Norton
```

HTTP PUT/PATCH Examples

- PUT/PATCH example:

```
PUT /web/Barry_Norton HTTP/1.1
Host: www.aifb.kit.edu
Content-Type: text/html

<html> ...
```

(new resource or complete update)

```
HTTP/1.0 200 OK (or 201 CREATED)
Date: Sun, 07 Nov 2010 00:10:00 GMT
```

```
PATCH /web/Barry_Norton HTTP/1.1
Host: www.aifb.kit.edu
Content-Type: text/html

Change...
```

(partial update)

```
HTTP/1.0 200 OK
Date: Sun, 07 Nov 2010 00:10:00 GMT
```

HTTP PUT/PATCH/POST Example

- POST-compute example:

```
POST /web/Barry_Norton HTTP/1.1
Host: www.aifb.kit.edu
Content-Type: something

Input ...
```

(input -> computation -> output)

```
HTTP/1.0 200 OK
Date: Sun, 07 Nov 2010 00:10:00 GMT
Content-Type: something

Result....
```

- POST-append example:

```
POST /web/Barry_Norton HTTP/1.1
Host: www.aifb.kit.edu
Content-Type: text/html

<html> ...
```

(new related resource)

```
HTTP/1.0 201 CREATED
Date: Sun, 07 Nov 2010 00:10:00 GMT
Location:http://www.aifb.kit.edu/...
```

Linked Data

Linked Data

Linked Data Principles:

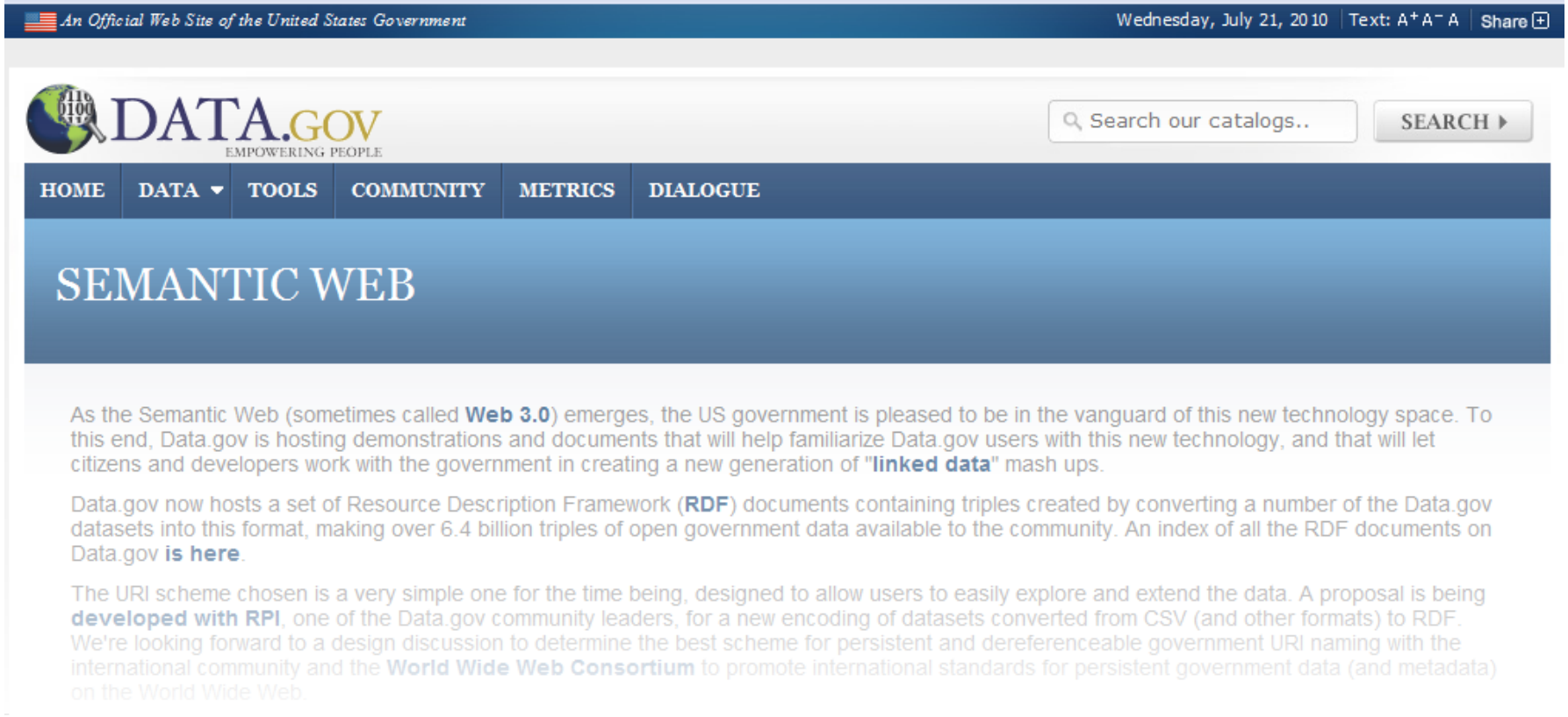
1. Use URIs as names for things
2. Use HTTP URIs so that people can look up those names.
3. When someone looks up a URI, provide useful information, using the standards (RDF*, SPARQL)
4. Include links to other URIs. so that they can discover more things.

- Also includes ‘best practice’, such as:
  - Separate URIs for real world entities (non-informational resources) from representations (informational resources), either by #URIs, or redirection (as shown before, but usually Status Code 303)

What RDF should be returned?

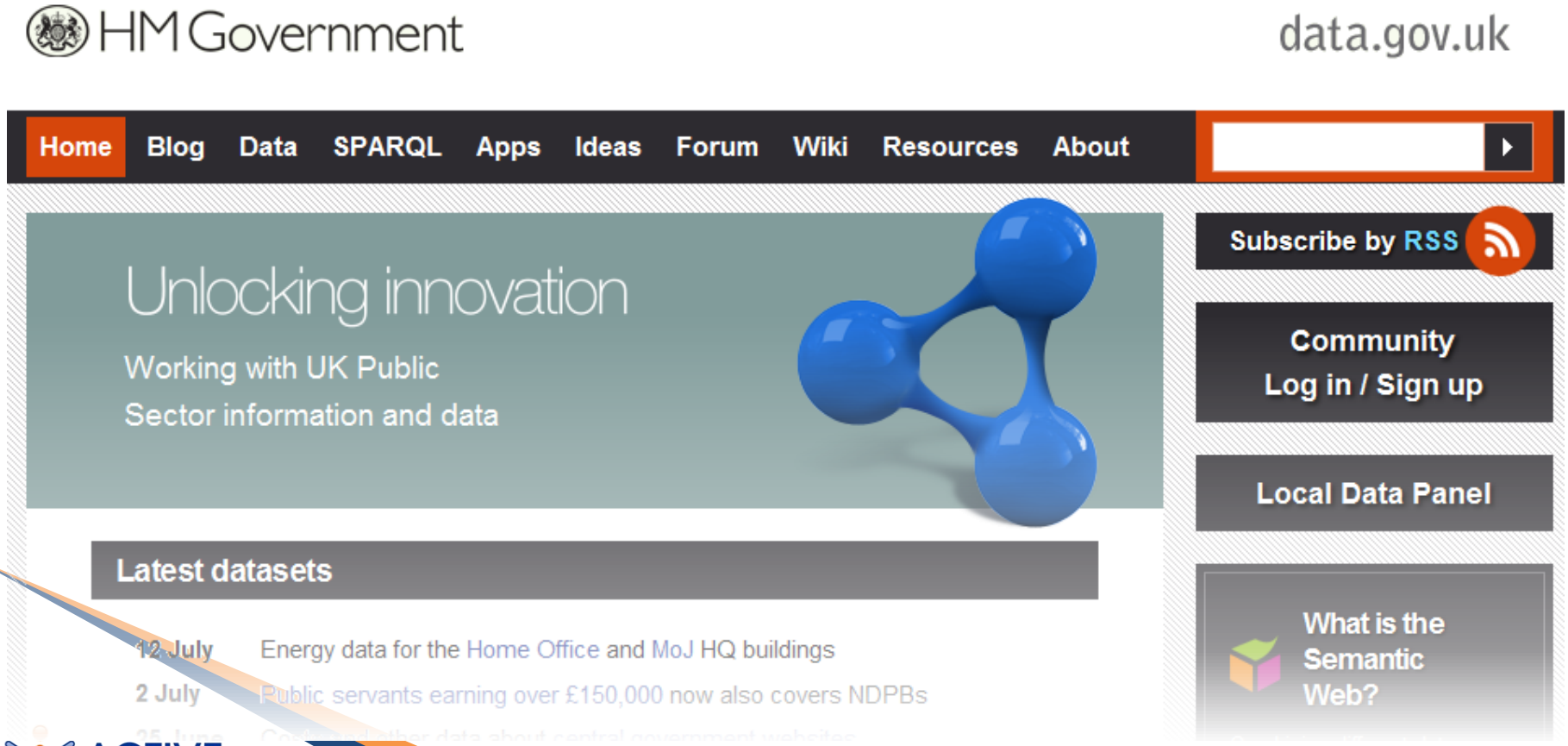
- **The (immediate) description:** All triples that have the resource's URI as the subject.
- **Backlinks:** All triples that have the resource's URI as the object. This is redundant, but it allows bi-directional traversal.
- **Related descriptions**: Anything about related resources that may be of interest in typical usage scenarios; use prudence.
- **Metadata:** Any metadata such as the author and licensing information.
- **Syntax:** At least RDF descriptions as RDF/XML which is the only official syntax for RDF.
  - As RDF/XML is not very human-readable, the data could additionally be provided in other formats; e.g., for MIME-type application/x-turtle.

Data.gov


An Official Web Site of the United States Government
Wednesday, July 21, 2010
Text: A+ A- A
Share
DATA.GOV
EMPOWERING PEOPLE
Search our catalogs..
SEARCH
HOME
DATA
TOOLS
COMMUNITY
METRICS
DIALOGUE
SEMANTIC WEB
As the Semantic Web (sometimes called Web 3.0) emerges, the US government is pleased to be in the vanguard of this new technology space. To this end, Data.gov is hosting demonstrations and documents that will help familiarize Data.gov users with this new technology, and that will let citizens and developers work with the government in creating a new generation of "linked data" mash ups.
Data.gov now hosts a set of Resource Description Framework (RDF) documents containing triples created by converting a number of the Data.gov datasets into this format, making over 6.4 billion triples of open government data available to the community. An index of all the RDF documents on Data.gov is here.
The URI scheme chosen is a very simple one for the time being, designed to allow users to easily explore and extend the data. A proposal is being developed with RPI, one of the Data.gov community leaders, for a new encoding of datasets converted from CSV (and other formats) to RDF. We're looking forward to a design discussion to determine the best scheme for persistent and dereferenceable government URI naming with the international community and the World Wide Web Consortium to promote international standards for persistent government data (and metadata) on the World Wide Web.

Data.gov.uk & Others

- Other governments are getting on board
  - UK an enthusiastic adopter



BBC & Media

- Value of resources increased by Linked Data

« Previous | Main | Next »

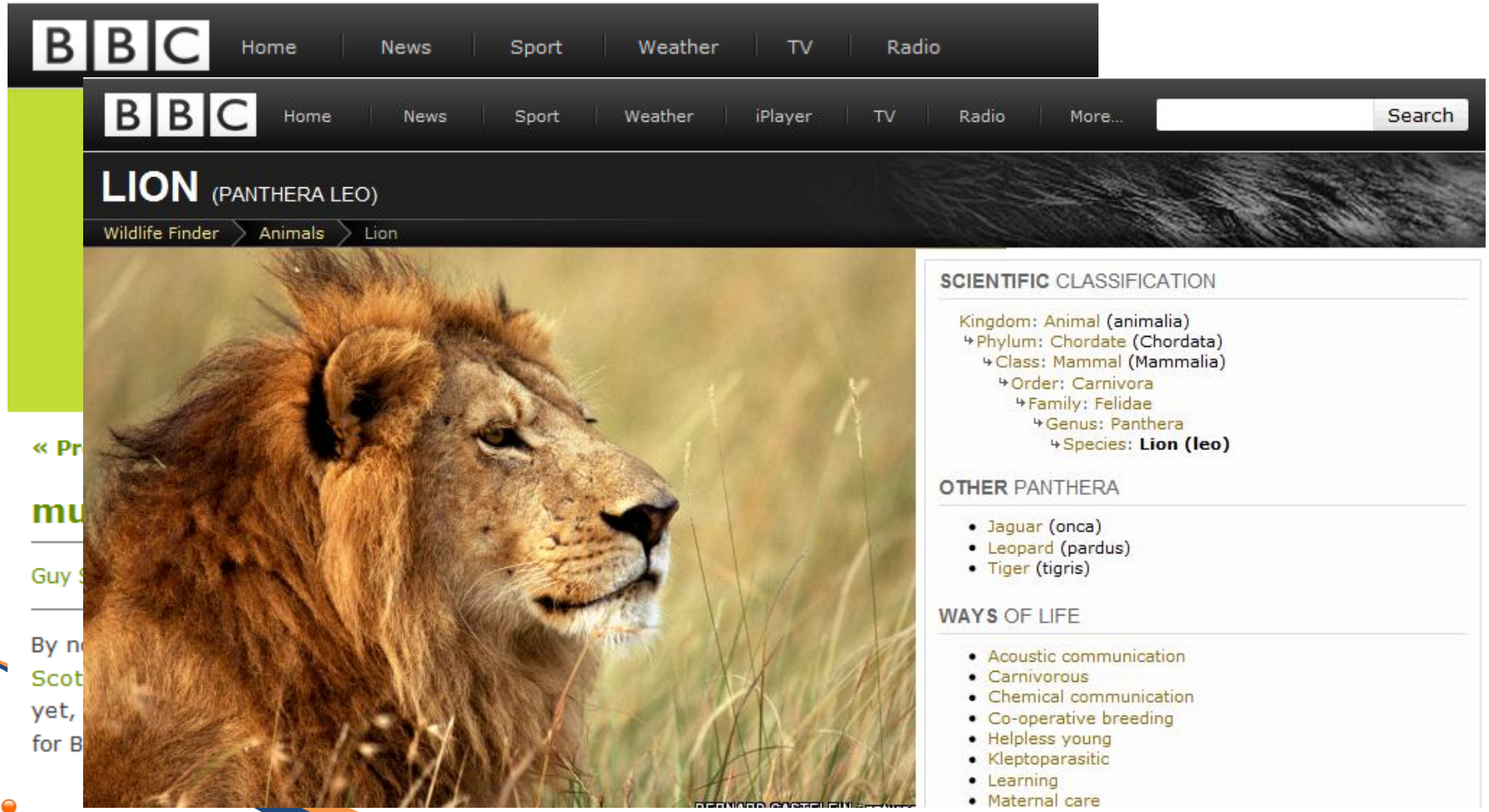
music beta and linked data

Guy Strelitz | 14:58 UK time, Wednesday, 30 July 2008

By now you may well have found the new BBC Music beta site - Matthew Shorter and Tom Scott have both blogged about it, and it's shown up on TechCrunch. If you haven't seen it yet, I strongly urge you read the blogs and take a look - it really is a huge step forward for BBC Music online, and for the data infrastructure of bbc.co.uk as a whole.

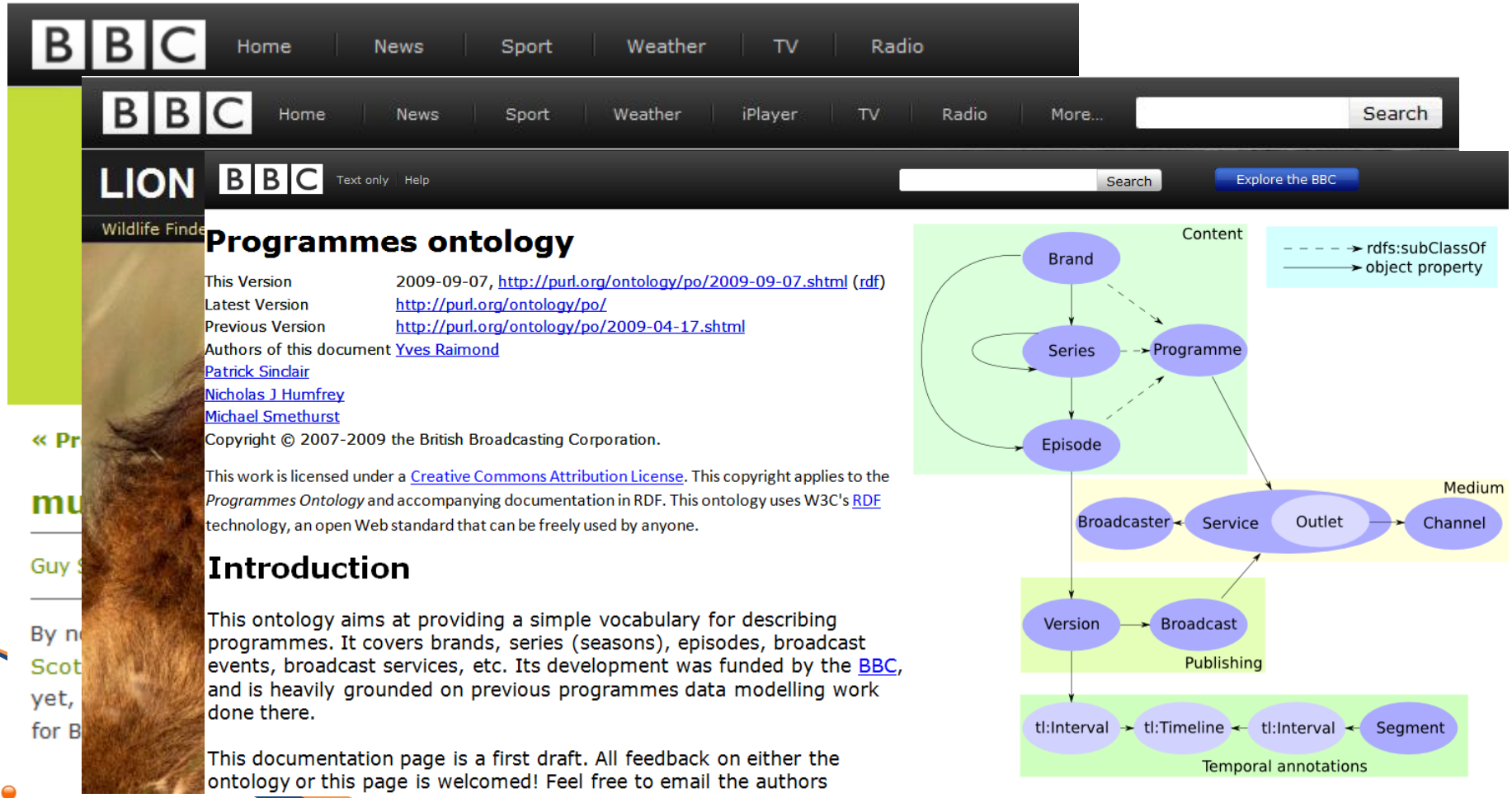
BBC & Media

- Value of resources increased by Linked Data



BBC & Media

- Value of resources increased by Linked Data



BestBuy

How Best Buy is Using The Semantic Web

Written by Richard MacManus / July 1, 2010 6:00 AM / 5 Comments

« Prior Post Next Post »

Yesterday we wrote about the increasing usage of Semantic Web technologies by large commercial companies like Facebook, Google and Best Buy. The Semantic Web is a Web of added meaning, which ultimately enables smarter and more personalized web apps to be built. In this post we explore how a leading U.S. retailer, Best Buy, is using a Semantic Web markup language called RDFa to add semantics to its webpages.

322 tweets retweet

This is not just an academic exercise for Best Buy. As we will see, semantic technology has already led to increased traffic and better service to its customers. We spoke to Jay Myers, Lead Web Development Engineer at BestBuy.com, to find out how.

BestBuy

How Best Buy is Using The Semantic Web

Written by Richard MacManus / July 1, 2010 6:00 AM / 5 Comments

« Prior Post Next Post »

Yesterday we wrote about the increasing usage of Semantic Web technologies by large commercial companies like Facebook, Google and Best Buy. The Semantic Web is a Web of added meaning, which ultimately enables smarter and more personalized web apps to be built. In this post we explore how a leading U.S. retailer, Best Buy, is using a Semantic Web markup language called RDFa to add semantics to its webpages.

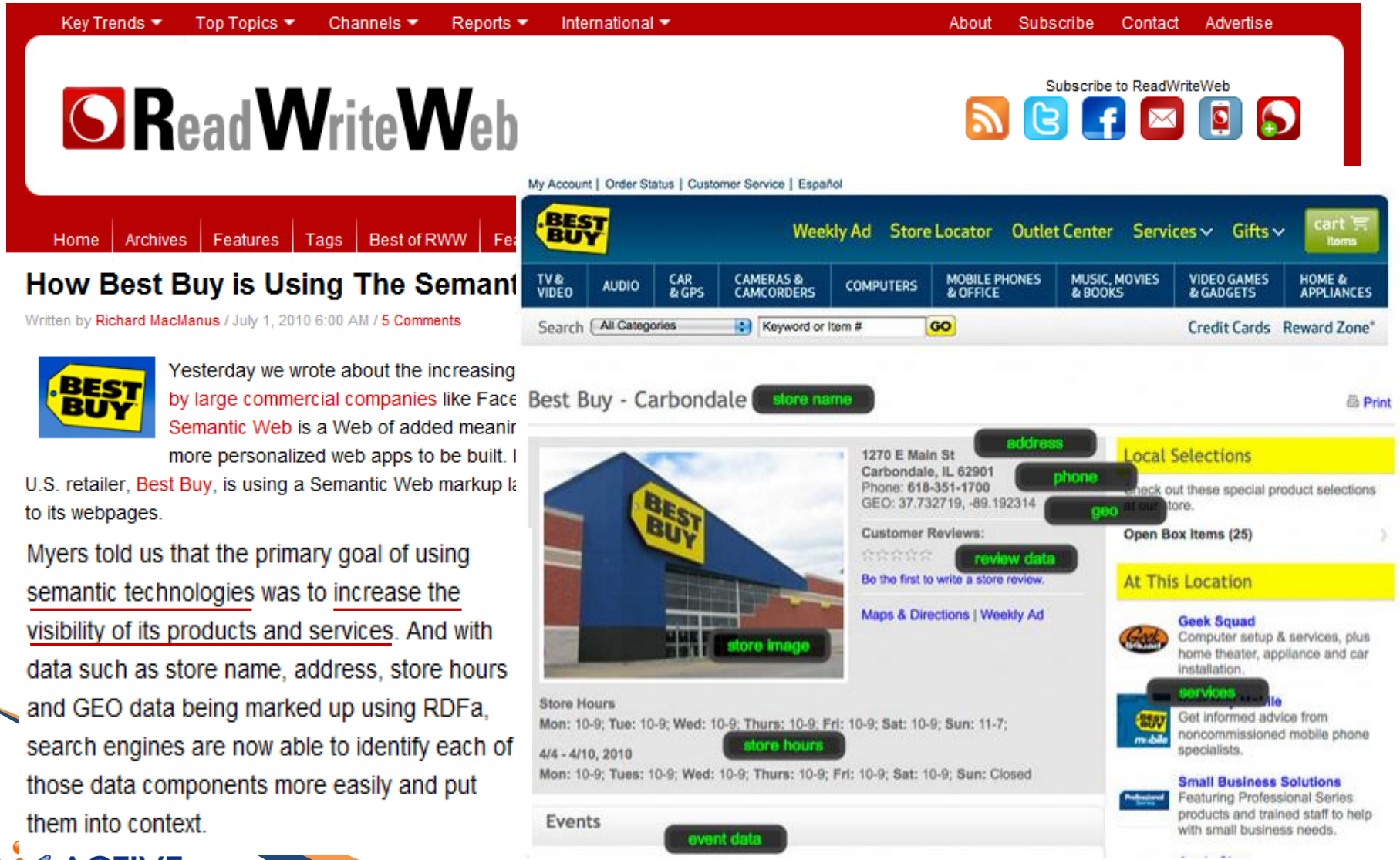
322 tweets

retweet

Myers told us that the primary goal of using semantic technologies was to increase the visibility of its products and services. And with data such as store name, address, store hours and GEO data being marked up using RDFa, search engines are now able to identify each of those data components more easily and put them into context.

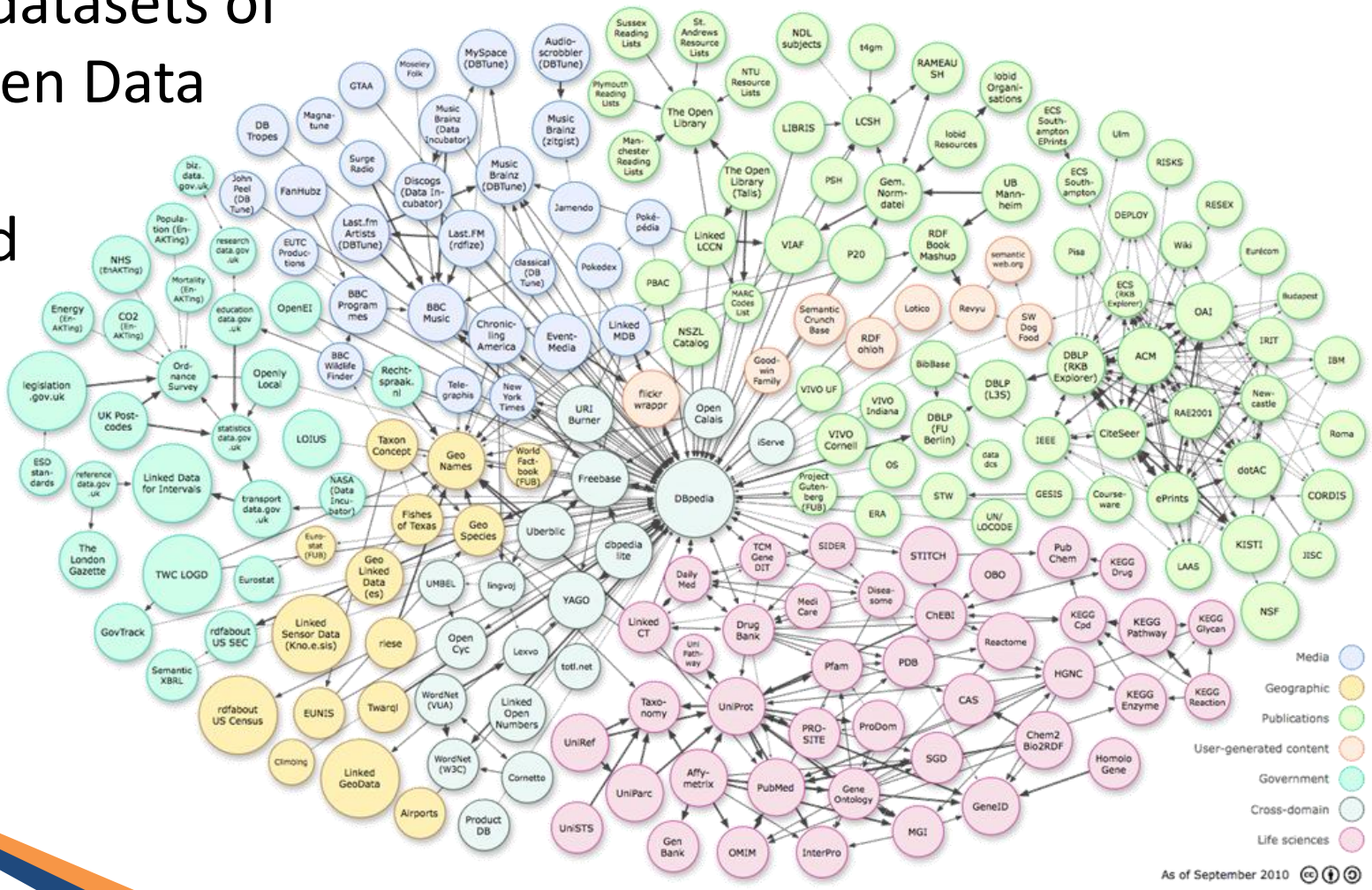
ll see, semantic technology customers. We spoke to Jay o find out how.

BestBuy


Key Trends
Top Topics
Channels
Reports
International
About
Subscribe
Contact
Advertise
ReadWriteWeb
Subscribe to ReadWriteWeb
Home
Archives
Features
Tags
Best of RWW
How Best Buy is Using The Semant
Written by Richard MacManus / July 1, 2010 6:00 AM / 5 Comments
Yesterday we wrote about the increasing
by large commercial companies like Face
Semantic Web is a Web of added meanir
more personalized web apps to be built. I
U.S. retailer, Best Buy, is using a Semantic Web markup la
to its webpages.
Myers told us that the primary goal of using
semantic technologies was to increase the
visibility of its products and services. And with
data such as store name, address, store hours
and GEO data being marked up using RDFa,
search engines are now able to identify each of
those data components more easily and put
them into context.
My Account | Order Status | Customer Service | Español
Weekly Ad
Store Locator
Outlet Center
Services
Gifts
cart
TV & VIDEO
AUDIO
CAR & GPS
CAMERAS & CAMCORDERS
COMPUTERS
MOBILE PHONES & OFFICE
MUSIC, MOVIES & BOOKS
VIDEO GAMES & GADGETS
HOME & APPLIANCES
Search
All Categories
Keyword or Item #
GO
Credit Cards
Reward Zone
Best Buy - Carbondale
store name
Print
address
1270 E Main St
Carbondale, IL 62901
phone
Phone: 618-351-1700
geo
GEO: 37.732719, -89.192314
Customer Reviews:
review data
Be the first to write a store review.
Maps & Directions | Weekly Ad
store image
Local Selections
Open Box Items (25)
At This Location
Geek Squad
Computer setup & services, plus home theater, appliance and car installation.
services
Get informed advice from noncommissioned mobile phone specialists.
Small Business Solutions
Featuring Professional Series products and trained staff to help with small business needs.
Store Hours
Mon: 10-9; Tue: 10-9; Wed: 10-9; Thurs: 10-9; Fri: 10-9; Sat: 10-9; Sun: 11-7;
4/4 - 4/10, 2010
store hours
Mon: 10-9; Tues: 10-9; Wed: 10-9; Thurs: 10-9; Fri: 10-9; Sat: 10-9; Sun: Closed
Events
event data

Linking Open Data Cloud

Together datasets of Linked Open Data form an interlinked ‘cloud’



Linked Services

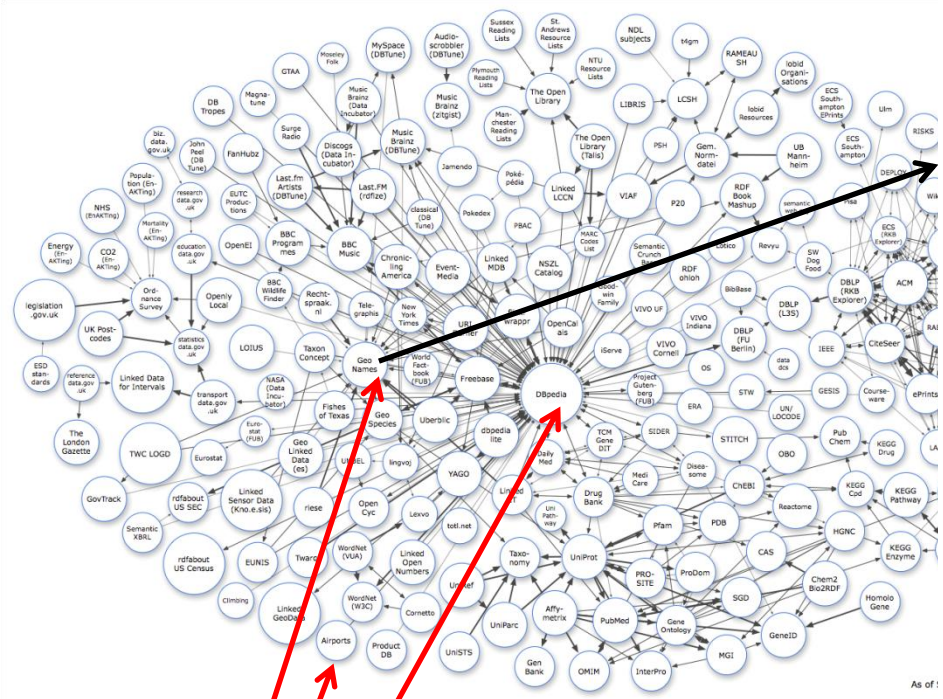
Services & Linked Data

- A problem can be seen in the current Linked Data sphere when it comes to services/APIs/functionalities
- The standards are often not then used
- The results of service interaction do not contribute to the Linked Data cloud
- Developers have to work with heterogeneous representations

GeoNames WebServices overview

	WebService	XML	JSON	RDF	CSV	TXT	RSS	
1	astergdem	XML	JSON			TXT		
2	children	XML	JSON					
3	cities	XML	JSON					
4	countryCode	XML	JSON			TXT		
5	countryInfo	XML	JSON		CSV			
6	countrySubdivision	XML	JSON					
7	earthquakes		JSON					
8	extendedFindNearby	XML						
9	findNearby	XML	JSON					
10	findNearbyPlaceName	XML	JSON					
11	findNearbyPostalCodes	XML	JSON					
12	findNearbyStreets US-only	XML	JSON					
13	findNearbyStreetsOSM	XML	JSON					
14	findNearByWeather Note-1	XML	JSON					
15	findNearbyWikipedia	XML	JSON				RSS	
16	findNearestAddress US-only	XML	JSON					
17	findNearestIntersection US-only	XML	JSON					
18	findNearestIntersectionOSM	XML	JSON					
19	get	XML	JSON					
20	gtopo30	XML	JSON			TXT		
21	hierarchy	XML	JSON					
22	neighbourhood US-only	XML	JSON					
23	neighbours	XML	JSON					
24	ocean	XML	JSON					
25	postalCodeCountryInfo	XML	JSON					
26	postalCodeLookup		JSON					
27	postalCodeSearch	XML	JSON					
28	rssToGeo			RDF			RSS	KML
29	search	XML	JSON					
30	siblings	XML	JSON					
31	srtm3	XML	JSON			TXT		
32	timezone Note-1	XML	JSON					
33	weather		JSON					
34	weatherIcao		JSON					
35	wikipediaBoundingBox	XML	JSON					
36	wikipediaSearch	XML	JSON					
	Total	31	34	1	1	4	2	1

Geonames Weather



```
{"weatherObservation":
 {"clouds":"broken clouds",
  "weatherCondition":"drizzle",
  "observation":"LESO 251300Z 03007KT
                 340V040 CAVOK 23/15 Q1010",
  "windDirection":30,
  "ICAO":"LESO", ...
```

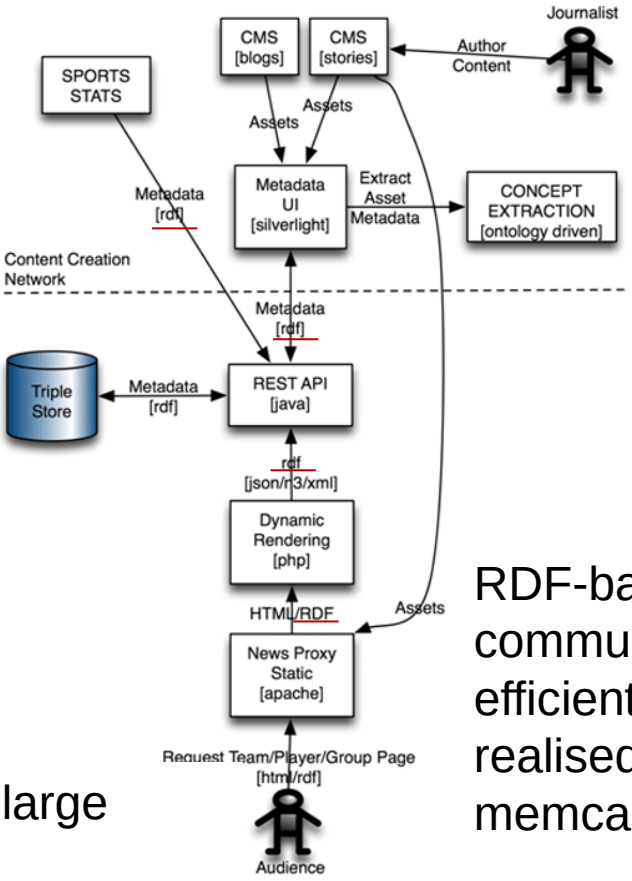
GeoNames WebServices overview

		XML	JSON	RDF	CSV	TXT	RSS	KI
1	astergdem	XML	JSON			TXT		
2	children	XML	JSON					
3	cities	XML	JSON					
4	countryCode	XML	JSON			TXT		
5	countryInfo	XML	JSON		CSV			
6	countrySubdivision	XML	JSON					
7	earthquakes		JSON					
8	extendedFindNearby	XML						
9	findNearby	XML	JSON					
10	findNearbyPlaceName	XML	JSON					
11	findNearbyPostalCodes	XML	JSON					
12	findNearbyStreets US-only	XML	JSON					
13	findNearbyStreetsOSM	XML	JSON					
14	findNearByWeather Note-1	XML	JSON					
15	findNearbyWikipedia	XML	JSON				RSS	
16	findNearestAddress US-only	XML	JSON					
17	findNearestIntersection US-only	XML	JSON					
18	findNearestIntersectionOSM	XML	JSON					
19	get	XML	JSON					
20	gtopo30	XML	JSON			TXT		
21	hierarchy	XML	JSON					
22	neighbourhood US-only	XML	JSON					
23	neighbours	XML	JSON					
24	ocean	XML	JSON					
25	postalCodeCountryInfo	XML	JSON					
26	postalCodeLookup		JSON					
27	postalCodeSearch	XML	JSON					
28	rssToGeo						RSS	KML
29	search	XML	JSON	RDF				
30	siblings	XML	JSON					
31	srtm3	XML	JSON			TXT		
32	timezone Note-1	XML	JSON					
33	weather		JSON					
34	weatherIcao		JSON					
35	wikipediaBoundingBox	XML	JSON					
36	wikipediaSearch	XML	JSON					
	Total	31	34	1	1	4	2	1

RDF Services at the BBC

- This is not a problem of scale, efficiency or speed, as a recent BBC application demonstrates:

Real-time updates to a large (ferocious) audience



RDF-based communication efficiently realised using memcached

REST Principles

- 1. Application state and functionality is divided into resources
- 2. Every resource is uniquely addressable
- 3. All resources share a uniform interface:
  - a) A constrained set of well-defined operations
  - b) A constrained set of content types

cf. RDF

URIs
again, cf. RDF

GET, PUT,
PATCH, POST,
DELETE

Inc.:
application/rdf+xml
text/n3 (and application/x-turtle)
application/sparql-results+xml

Linked Open Services

aim to promote this style of service, bringing together:

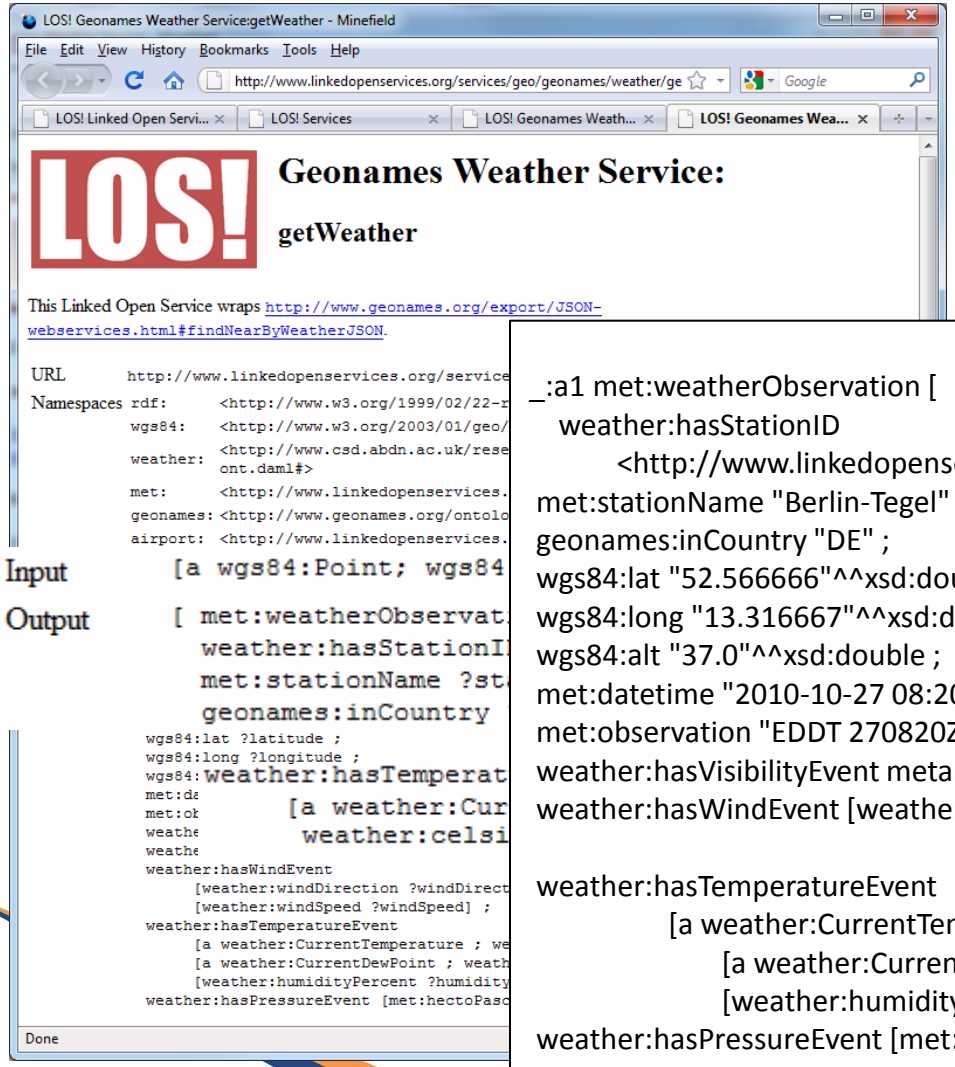
- RESTful services (respecting Web architecture)
  - resource-oriented
  - manipulated with HTTP verbs
    - GET, PUT (, PATCH), POST, DELETE
  - Negotiate representations
- Linked data
  - Uniform use of URIs
  - Use of RDF and SPARQL

Linked Open Service Principles

- Concretely, Linked Open Services come with a set of guiding principles:
  - 1. Describe services as **LOD prosumers** with input and output descriptions as **SPARQL graph patterns**
  - 2. **Communicate RDF** by RESTful content negotiation
  - 3. Include the **implicit knowledge contribution** that results from interactions in service descriptions and communications
- Associated with the last principle is an optional fourth:
  - 4. When wrapping non-LOS services, describe the lifting/lowering/mapping with SPARQL queries

http://www.linkedopenservices.org/blog/?page_id=2

LOS Weather Description




```
_:a1 met:weatherObservation [
   weather:hasStationID
        <http://www.linkedopenservices.org/services/geo/SpatialResources/point/ICAO/EDDT> ;
met:stationName "Berlin-Tegel" ;
geonames:inCountry "DE" ;
wgs84:lat "52.566666"^^xsd:double ;
wgs84:long "13.316667"^^xsd:double ;
wgs84:alt "37.0"^^xsd:double ;
met:datetime "2010-10-27 08:20:00" ;
met:observation "EDDT 270820Z 18007KT CAVOK 06/02 Q1025 NOSIG" ;
weather:hasVisibilityEvent metar:BrokenCloud ;
weather:hasWindEvent [weather:windDirection "180"^^xsd:short] ,
                                        [weather:windSpeedKnots "07"^^xsd:short] ;
weather:hasTemperatureEvent
            [a weather:CurrentTemperature ; weather:celsiusTemperature "6"^^xsd:short] ,
                 [a weather:CurrentDewPoint ; weather:celsiusTemperature "2"^^xsd:short] ,
                 [weather:humidityPercent "75"^^xsd:short ;
weather:hasPressureEvent [met:hectoPascal "1025"^^xsd:integer] ] .
```

Linked Open Services Weather

www.linkedopenservices.org/services/geo/SpatialResources/point/ICAO/DGAA/page/

Spatial Resources:

ACCRA/KOTOKA INT

ACCRA/KOTOKA INT (http://www.linkedopenservices.org/services/geo/SpatialResources/point/ICAO/DGAA)

- lat: 5.5833335
- long: -0.16666667
- alt: 69.0
- sameAs:
  - http://dbpedia.org/resource/Kotoka_International_Airport

Current weather report may be available →

www.linkedopenservices.org/services/geo/SpatialResources/point/ICAO/DGAA/weather/

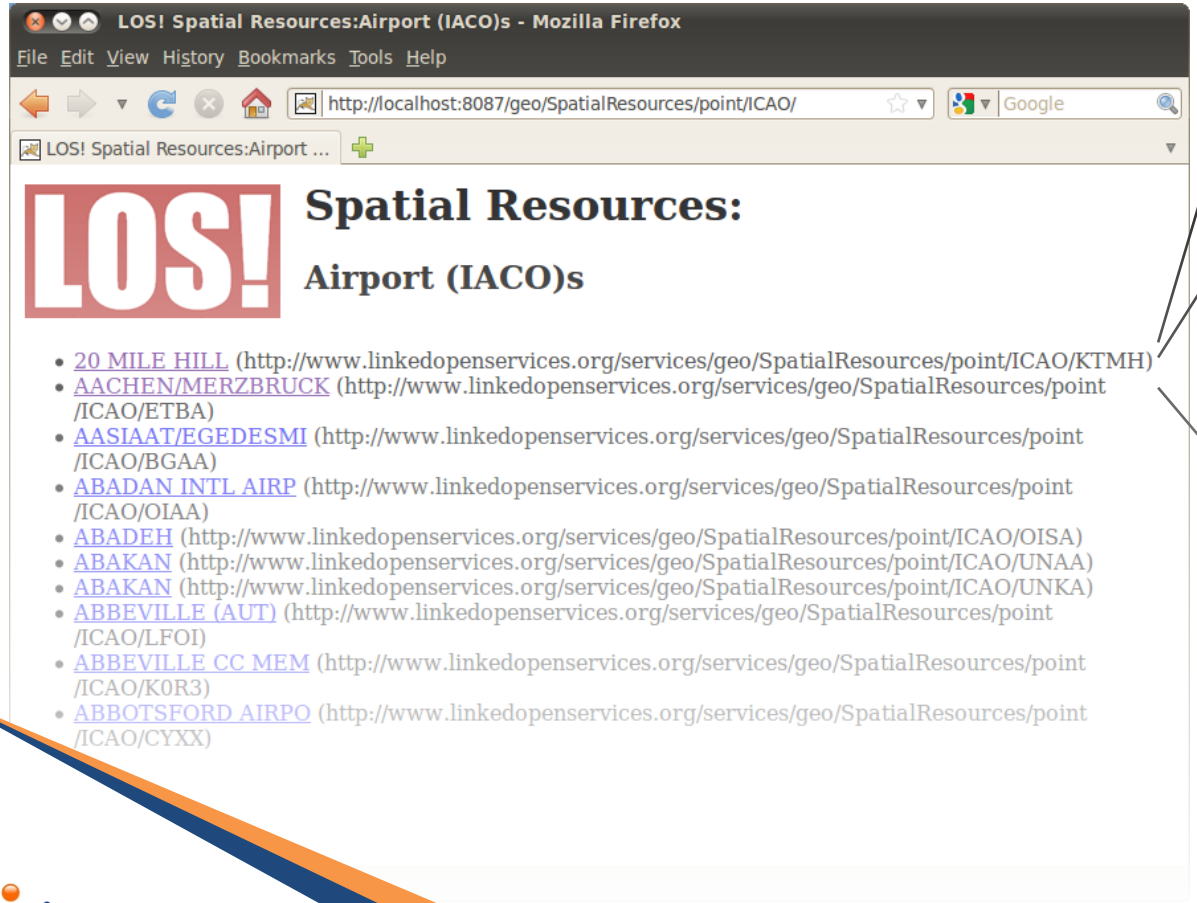
Weather at Accra

32℃

Full report available in RDF

Linked Open Services Weather

ICAO and IATA resources add 45 500 explicit static triples plus 17 750 more inferred (via `sameAs`)



GET
Accept: text/html

↓

303 REDIRECT /page

GET
Accept:
application/rdf+xml
(or text/n3)

↓

303 REDIRECT /data

Linked Data

GET /weather
Accept:
application/rdf+xml
(or text/n3)

200 <rdf:Description>

Linked Service

ACTIVE
Knowledge-Powered Enterprise
www.active-project.eu

Questions?

http://active-project.eu