FAKULTÄT FÜR !NFORMATIK

Faculty of Informatics

An Introduction to Linked Data

3rd Keystone Training School
Keyword Search in Big Linked Data

Vienna, August 21, 2017

Elmar Kiesling
Linked Data Lab, TU Wien

Motivation and Objectives

- Motivation
  - *Keyword search in Big **Linked Data***
  - Make use of abundant Linked Data available in some fields
  - Lots of ~~unresolved problems/challenges/~~research opportunities ;)

- Objectives:
  - Overview of what Linked Data is and where it may be heading
  - History and conceptual foundations
  - Learn how to query Linked Data
  - Mostly: basics required for the other lectures

- Not covered:
  - Linked Data lifecycle
  - Publishing, processing, integration, visualization of LD
  - Building applications on top of LD
  - Search in Linked LD
  - …

Outline

Part 1: Origins and vision

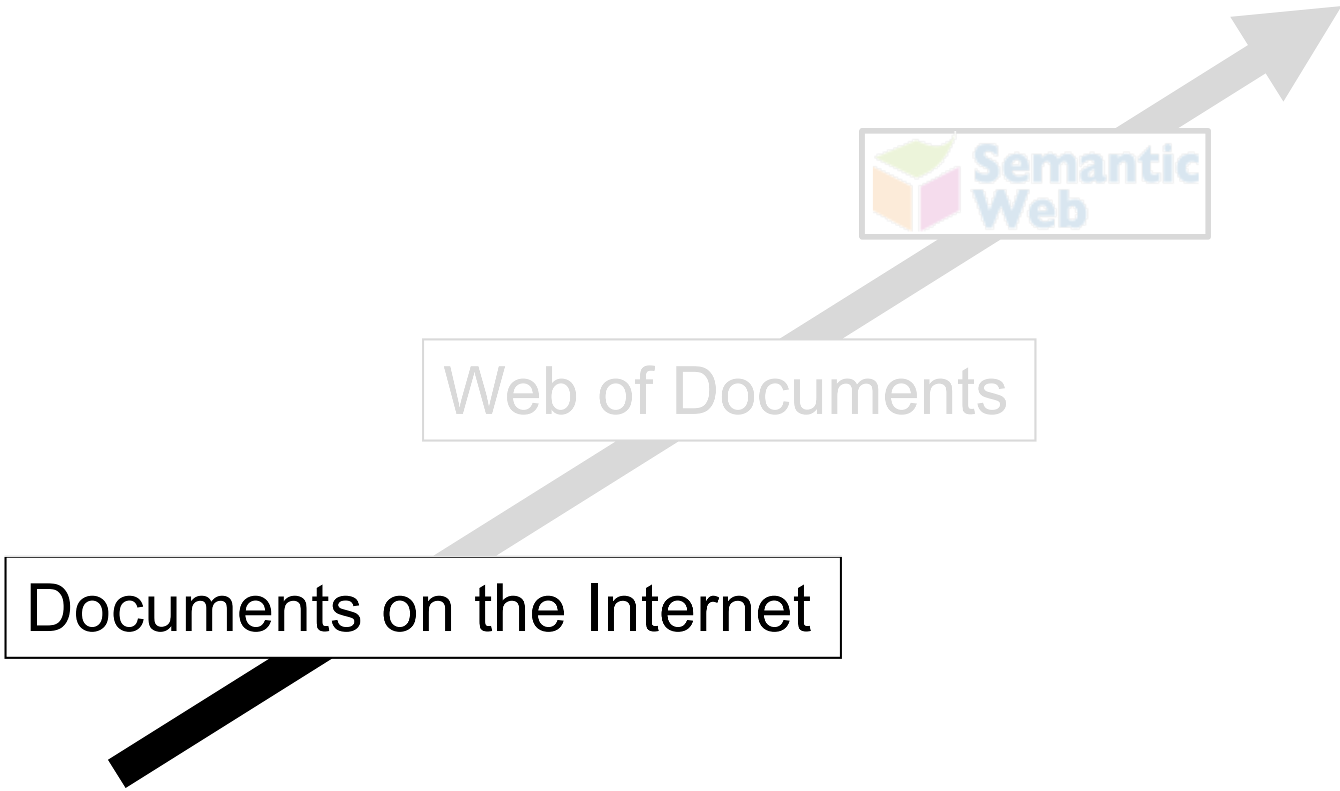
- A brief history of the Semantic Web
- Linked Data principles
- Adoption and applications today
- What does this have to do with keyword search?

Part 2: Deep dive

- Quick guided tour to the Semantic Web Technology Stack
- Querying Linked Data with SPARQL

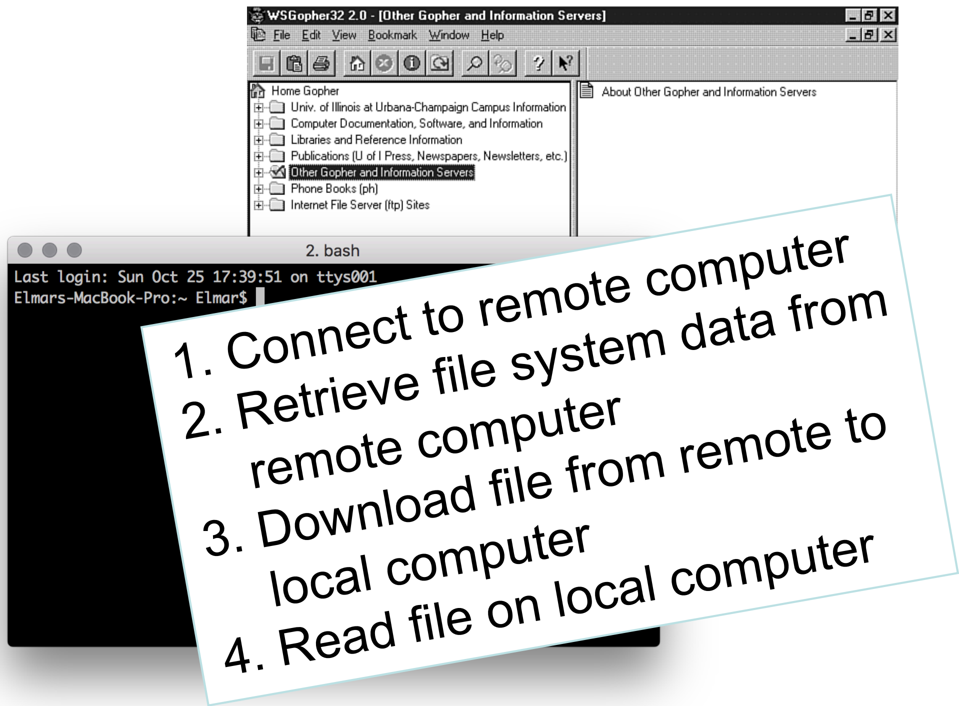
PART 1: ORIGINS AND VISION

The Semantic web vision
(or: the revolution that never was..)



History: Documents on the Internet

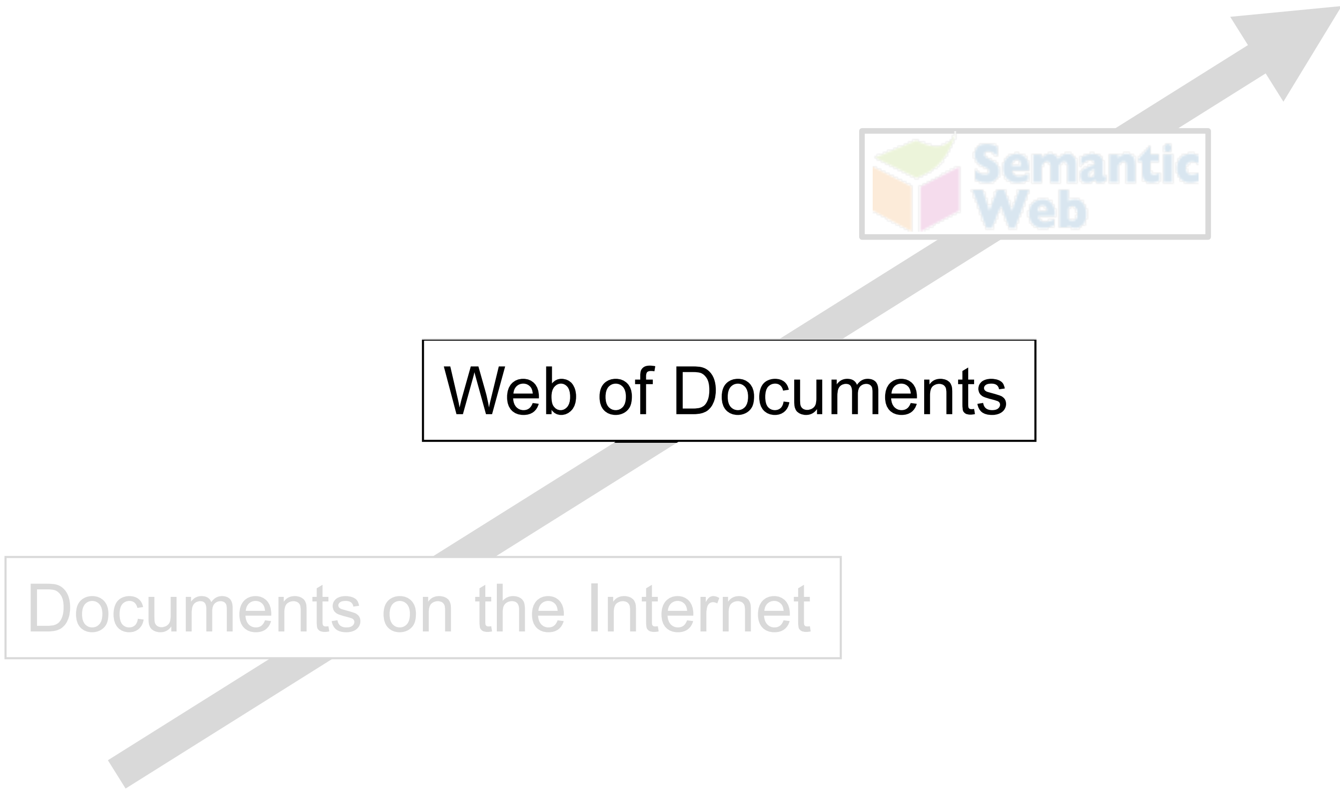
- Internet enabled remote documents retrieval
- Computer centered processing
- Services like FTP, Telnet, Gopher



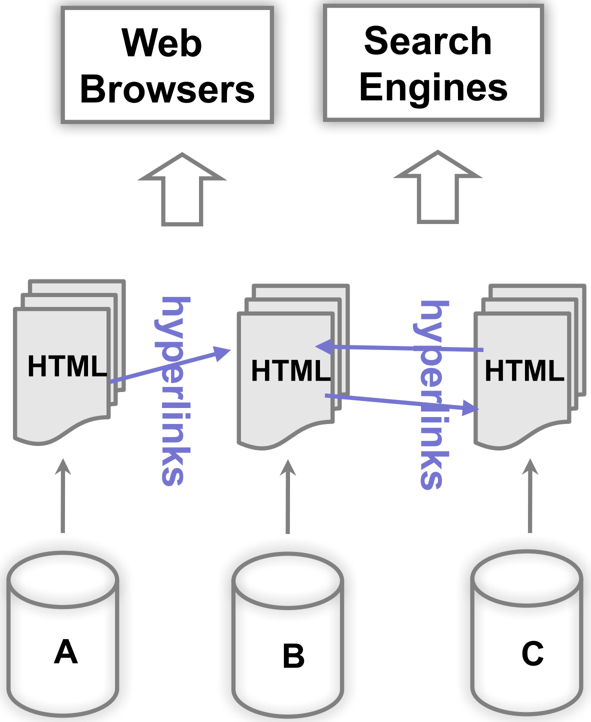
Limitations:

- Information access requires expert knowledge
- Information access is expensive
- Information retrieval is expensive

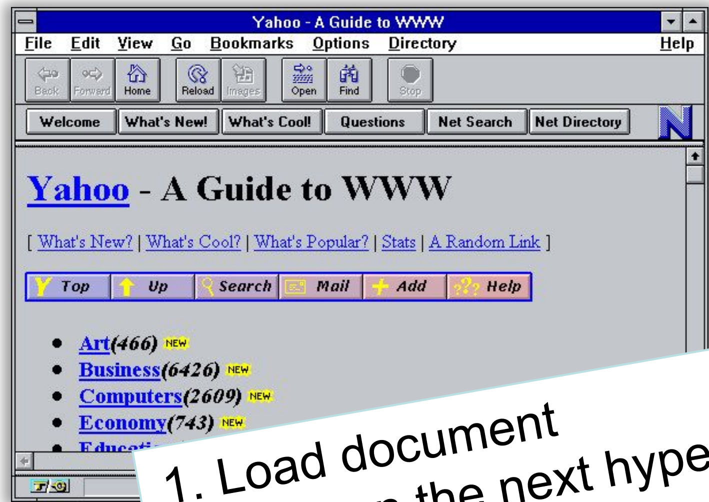
The Semantic web vision
(or: the revolution that never was..)



History: Web of documents



- Document-centered processing
- Single Global Information Space based on
  - URLs as
    - globally unique IDs
    - retrieval mechanism
  - HTML as shared content format
  - Hyperlinks



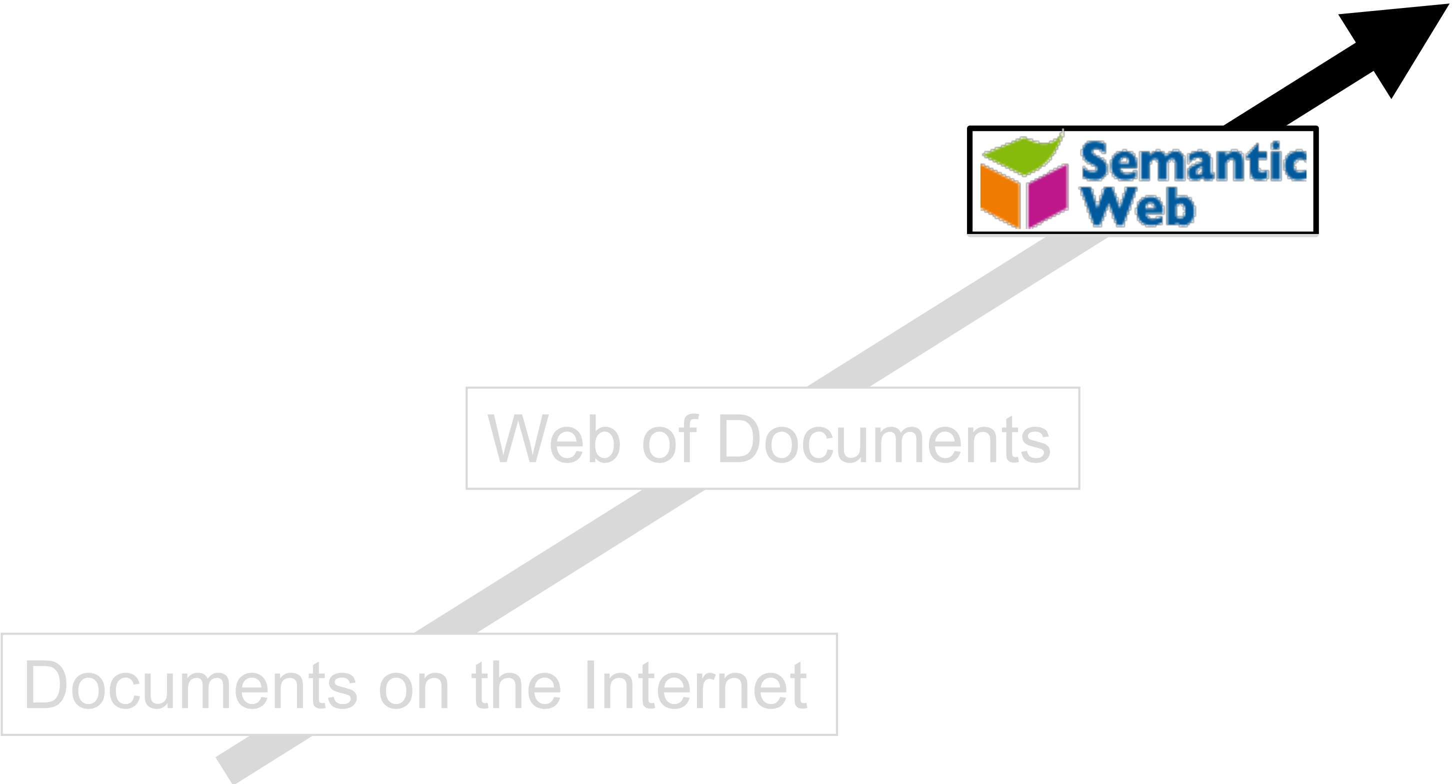
1. Load document
2. Click on the next hyperlink

…

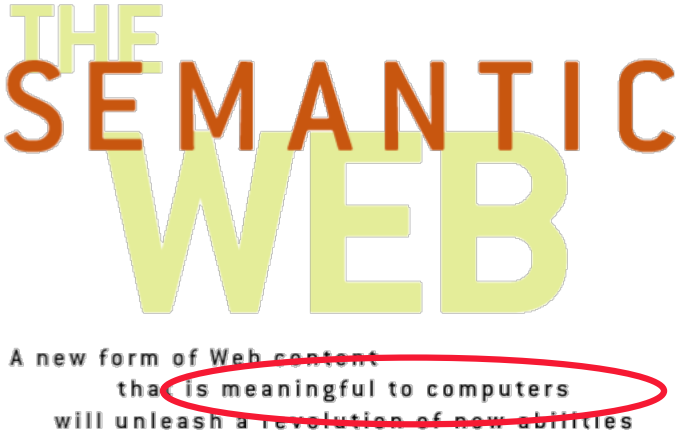
Advantages:

- No expert knowledge required
- Simplified information access
- Information retrieval via search engines

The Semantic web vision
(or: the revolution that never was..)



The Semantic Web vision



- *"The Semantic Web is an **extension of the current web** in which information is given well-defined meaning, better enabling computers and people to work in cooperation."* *
- *"I have a dream for the Web [in which computers] become capable of analyzing all the data on the Web – the content, links, and transactions between people and computers.“*
- *"..creating an environment where **software agents roaming from page to page** can readily carry out sophisticated tasks for users."*

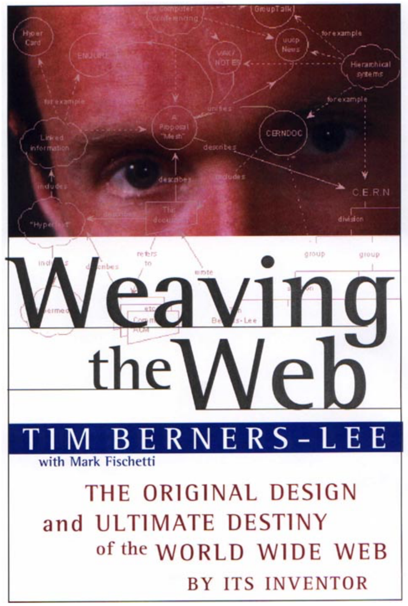
Tim Berners Lee 1999

Berners-Lee, Hendler, and Lassila, *The Semantic Web*, Scientific American, May 2001

The Semantic Web vision

"I have a dream for the Web... and it has two parts."

1. The first Web enables communication between people
2. The new Web will bring computers into the action



Step 1: Describe - putting data on the Web in machine-understandable form to form a Semantic Web

- RDF (based on XML)
- Master list of terms used in a document (RDF schema)
- Each document mixes global standards and local agreed-upon terms (namespaces)

Step 2: Infer and reason - apply logic inference

- Operate on partial understanding
- Answering why
- Heuristics

Adapted from C. Knoblock: The Semantic Web (based on slides by Yolanda Gil, Ian Horrocks, Jose Luis Ambite, and Tom Russ)
https://www.isi.edu/integration/courses/csci548_2010/slides/Semantic_Web.pdf

So where is the Semantic Web?

“The Semantic Web, with its neat ontologies and its syllogistic logic, is a nice vision. However, like many visions that project future benefits ***but ignore present costs****, it* ***requires too much coordination*** *and too much energy to be effective* ***in the real world****, where* ***deductive logic is less effective and shared worldview is harder to create than we often want to admit****."*

“A world of exhaustive, reliable metadata would be a utopia.”

Clay Shirkey

Clay Shirkey: The Semantic Web, Syllogism, and Worldview, Nov. 2003
http://www.shirky.com/writings/semantic_syllogism.html

Hypotheses on why the SW did not materialize (yet)

- **Conceptual issues:**
  No agreement on *how exactly* to specify intended meaning
  - There is always more than one way to represent facts
  - There is no natural, "neutral" schema for anything

Hypotheses on why the SW did not materialize (yet)

- **Conceptual issues:**
  No agreement on *how exactly* to specify intended meaning
  - There is always more than one way to represent facts
  - There is no natural, "neutral" schema for anything

- **Technical issues:**
  - Low tolerance for partial, context-sensitive, inconsistent information
  - Reasoning can easily break in an open environment
  - Intelligent agents are more difficult than envisioned
  - Upper layers of the stack not (sufficiently) developed
  - Computational tractability, scalability, reliability, robustness..

Hypotheses on why the SW did not materialize (yet)

- **Conceptual issues:**
  No agreement on *how exactly* to specify intended meaning
  - There is always more than one way to represent facts
  - There is no natural, "neutral" schema for anything

- **Technical issues:**
  - Low tolerance for partial, context-sensitive, inconsistent information
  - Reasoning can easily break in an open environment
  - Intelligent agents are more difficult than envisioned
  - Upper layers of the stack not (sufficiently) developed
  - Computational tractability, scalability, reliability, robustness..

- **Practical/economic issues:**
  - Architecture and standards overly complex and poorly understood
  - Incomplete tool stack
  - Ontology development is expensive
  - **Adoption incentives?**
  - **Business models?**

"Pragmatic shift" towards Linked Data

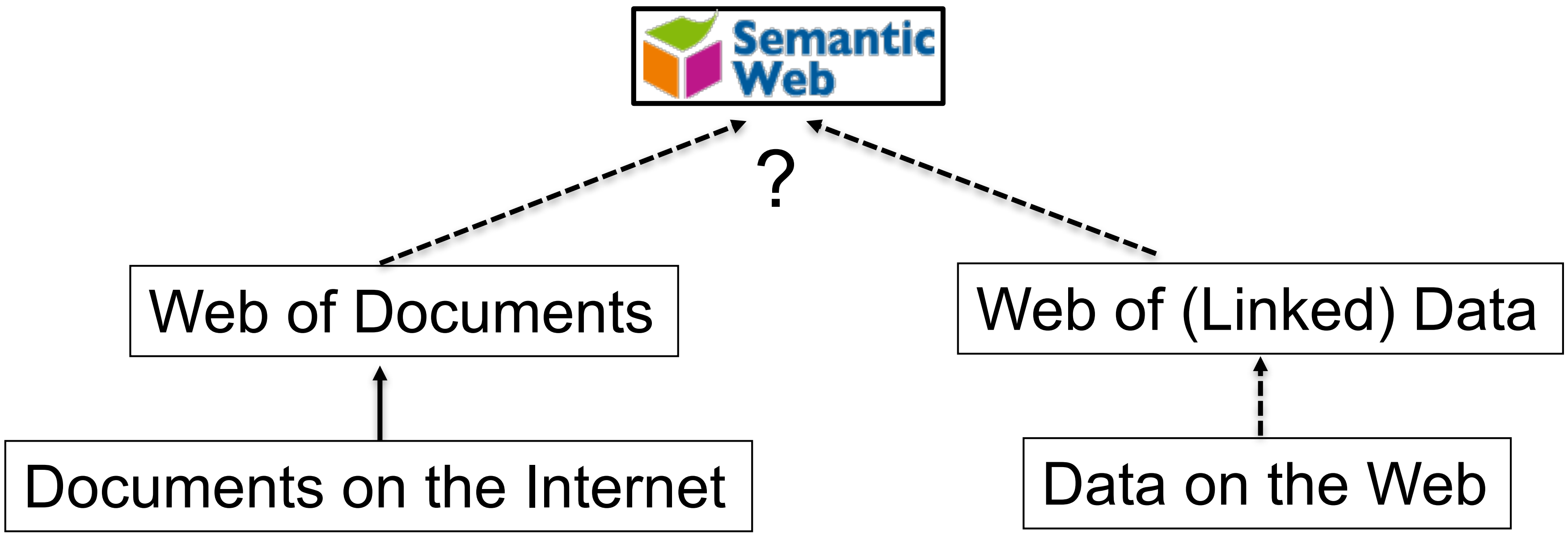
"Much of the proposed value of the Semantic Web is coming, but it is not coming because of the Semantic Web.

The amount of meta-data we generate is increasing dramatically, and it is being exposed for consumption by machines as well as, or instead of, people. But it is being designed ***a bit at a time, out of self-interest and without regard for global ontology****."*

Clay Shirkey

Clay Shirkey: The Semantic Web, Syllogism, and Worldview, Nov. 2003
http://www.shirky.com/writings/semantic_syllogism.html

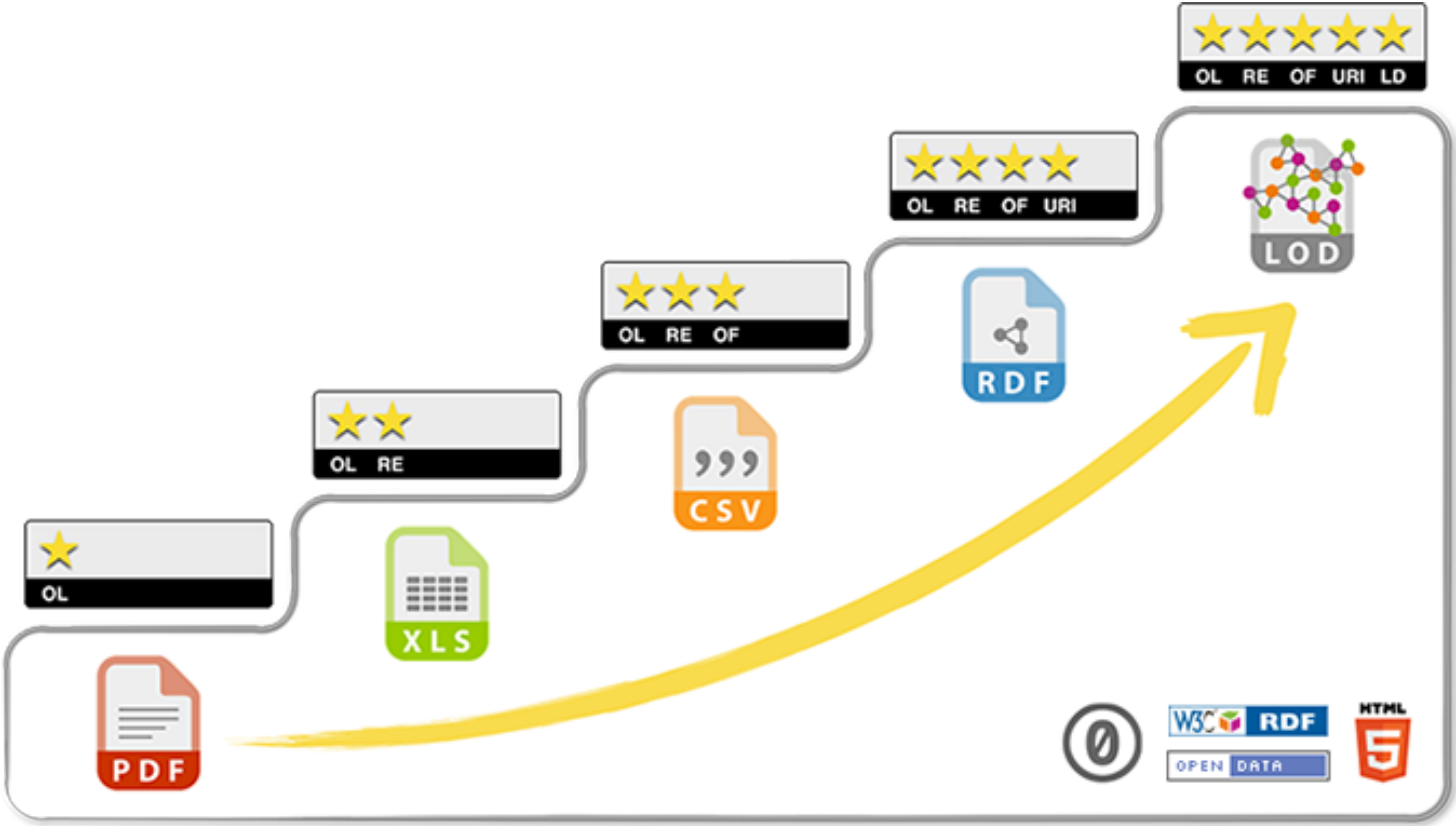
Evolution of the web



(Open) Data on the web

- Publication of structured data on the web
- Allows sharing, remix, reuse
- Typically used formats: csv, xml, json, pdf, shp, kmz, …
- Semantics typically not well-defined
- Many national and international Open Data initiatives and portals

5 Star Deployment scheme for Open Data



http://5stardata.info/en/

5 ⭐ Deployment Scheme for Open Data

Data is…   ..if it…

5 ⭐ Deployment Scheme for Open Data

Data is… ..if it…

..is available on the Web under an **open license**

5 ⭐ Deployment Scheme for Open Data

Data is… ..if it…

..is available on the Web under an **open license**

..is available as **machine-readable** structured data

5 ⭐ Deployment Scheme for Open Data

Data is…	..if it…
	..is available on the Web under an **open license**
	..is available as **machine-readable** structured data
	..is available in **non-proprietary formats**

5 ⭐ Deployment Scheme for Open Data

Data is… ..if it…

..is available on the Web under an **open license**

..is available as **machine-readable** structured data

..is available in **non-proprietary formats**

..uses **URIs** so people can point to individual items

5 ⭐ Deployment Scheme for Open Data

Data is… ..if it…

..is available on the Web under an **open license**

..is available as **machine-readable** structured data

..is available in **non-proprietary formats**

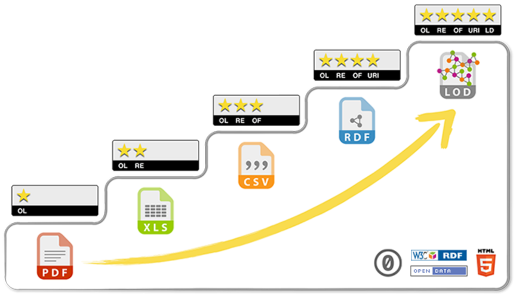
..uses **URIs** so people can point to individual items

..**linked** to other data to provide context

Adapted from EUCLID learning materials – “Introduction to Linked Data”

Quick Poll: I have published…

1. Data with an open license
2. Machine-readable structured data
3. Data in open formats
4. Data identified with open standards
5. Data linked to other provider's data



Linked Data – Definition

*"Linked Data describes a method of **publishing structured data so that it can be interlinked** and become more useful.*

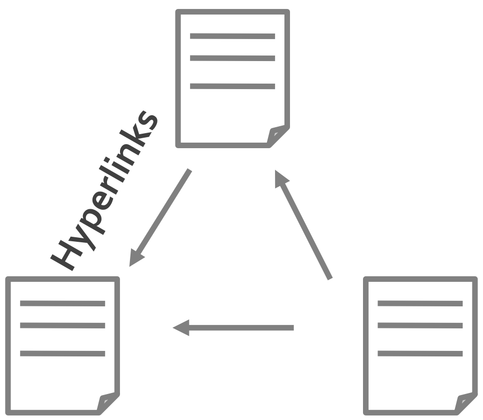
*It builds upon **standard Web technologies** such as HTTP and URIs, but rather than using them to serve web pages for human readers, it **extends them to share information in a way that can be read automatically by computers**.*

This enables data from different sources to be connected and queried."

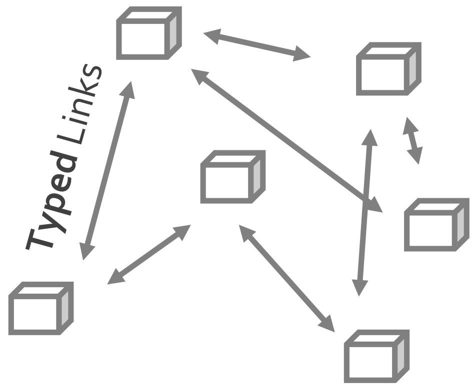
Bizer, Heath, Berners-Lee (2009) "Linked Data – The Story So Far"
International Journal on Semantic Web and Information Systems 5 (3)

Linked Data from 10,000 foot…

Web of Documents



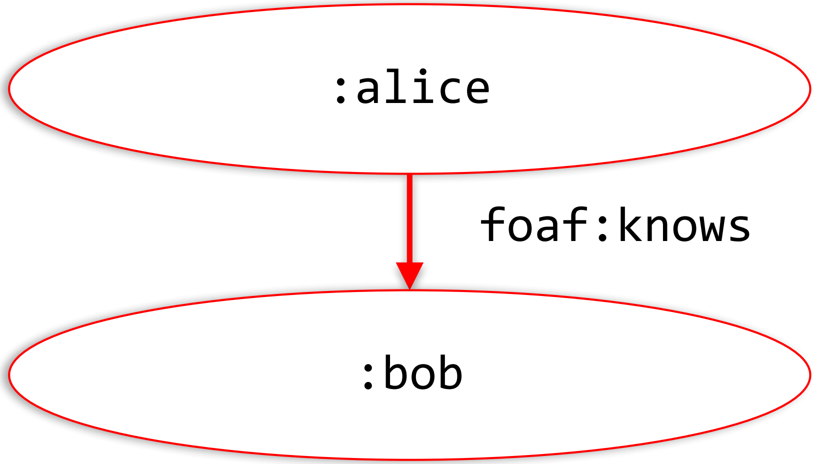
Web of Data



- Best practices for publishing and connecting structured data on the Web
- Goal: Create a global data space

... and up-close

- Graph-based data model
- Subject-predicate-object triples
- Use of URIs as globally unique identifiers



- Graph-based data model
- Subject-predicate-object triples
- Use of URIs as globally unique identifiers

Linked Data Principles

Design Issues: Linked Data notes, Tim Berners-Lee

Linked Data Principles

1. Use URIs to identify things

Design Issues: Linked Data notes, Tim Berners-Lee

Linked Data Principles

1. Use URIs to identify things
2. Use HTTP URIs so that people can look up those names

Design Issues: Linked Data notes, Tim Berners-Lee

Linked Data Principles

1. Use URIs to identify 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..)

Design Issues: Linked Data notes, Tim Berners-Lee

Linked Data Principles

1. Use URIs to identify 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..)
4. Include links to other URIs so that they can discover more things

Design Issues: Linked Data notes, Tim Berners-Lee

LD Principle 1

1. Use URIs as names for things

- All things or distinct entities within the data must be named
- According to the Linked Data principles, the standard mechanism to name entities is the URI

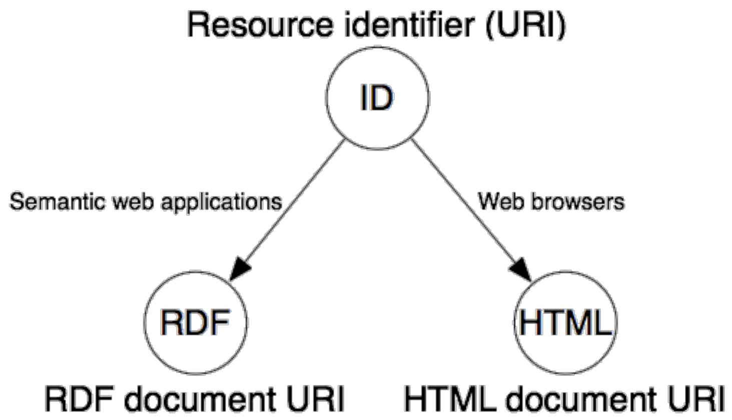
Designing Cool URIs*:

1. Simplicity: short, mnemonic URIs
2. Stability: maintain the URIs as long as possible
3. Manageability: issue URIs in a way that you can manage

*Source: http://www.w3.org/TR/cooluris/

2. Use HTTP URIs so users can look up those names

Content negotiation:
Client specifies preferred format in HTTP header, e.g.




```
Accept: application/xhtml+xml

Accept: application/rdf+xml
```

HTTP 303 redirects and location header should be used to point the client to another URL where the appropriate representation is found

Image: http://www.w3.org/TR/cooluris/

3. When someone looks up a URI, provide useful information, using the standards (RDF*, SPARQL, Turtle).

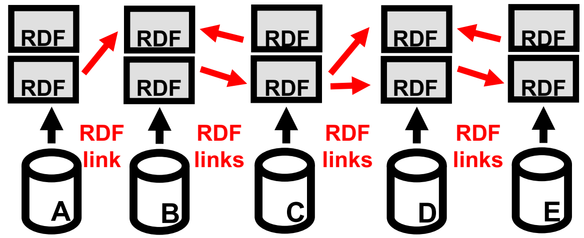
What to return for a URI?

- **Immediate description:** triples where the URI is the subject
- **Backlinks:** triples where the URI is the object
- **Related descriptions:**
  information of interest in typical usage scenarios
- **Metadata:** information as author and licensing information
- **Syntax:** RDF descriptions as RDF/XML and human-readable formats

Source: *How to Publish Linked Data on The Web*
Chris Bizer, Richard Cyganiak, Tom Heath.

LD Principle 4

4. Include links to other URIs, so that users can discover more things.



Reuse URIs:

- direct **reuse**
- (OWL) **sameAs**
- (RDFS) **seeAlso**

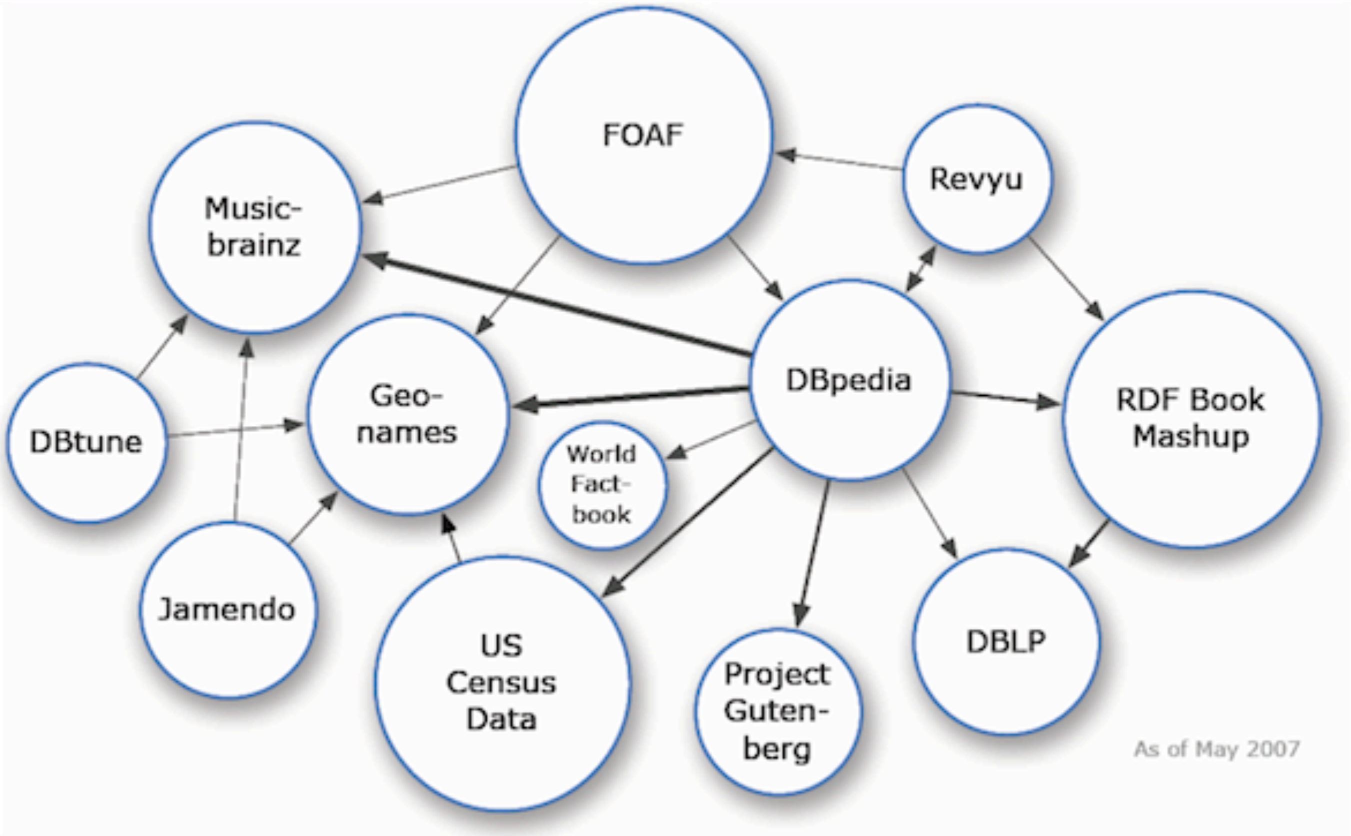
Instance Level

- direct **reuse** of class/property
- (RDFS) **sub**-class/-property
- (OWL) **equivalent** class/property
- SKOS **broad match**

Schema Level

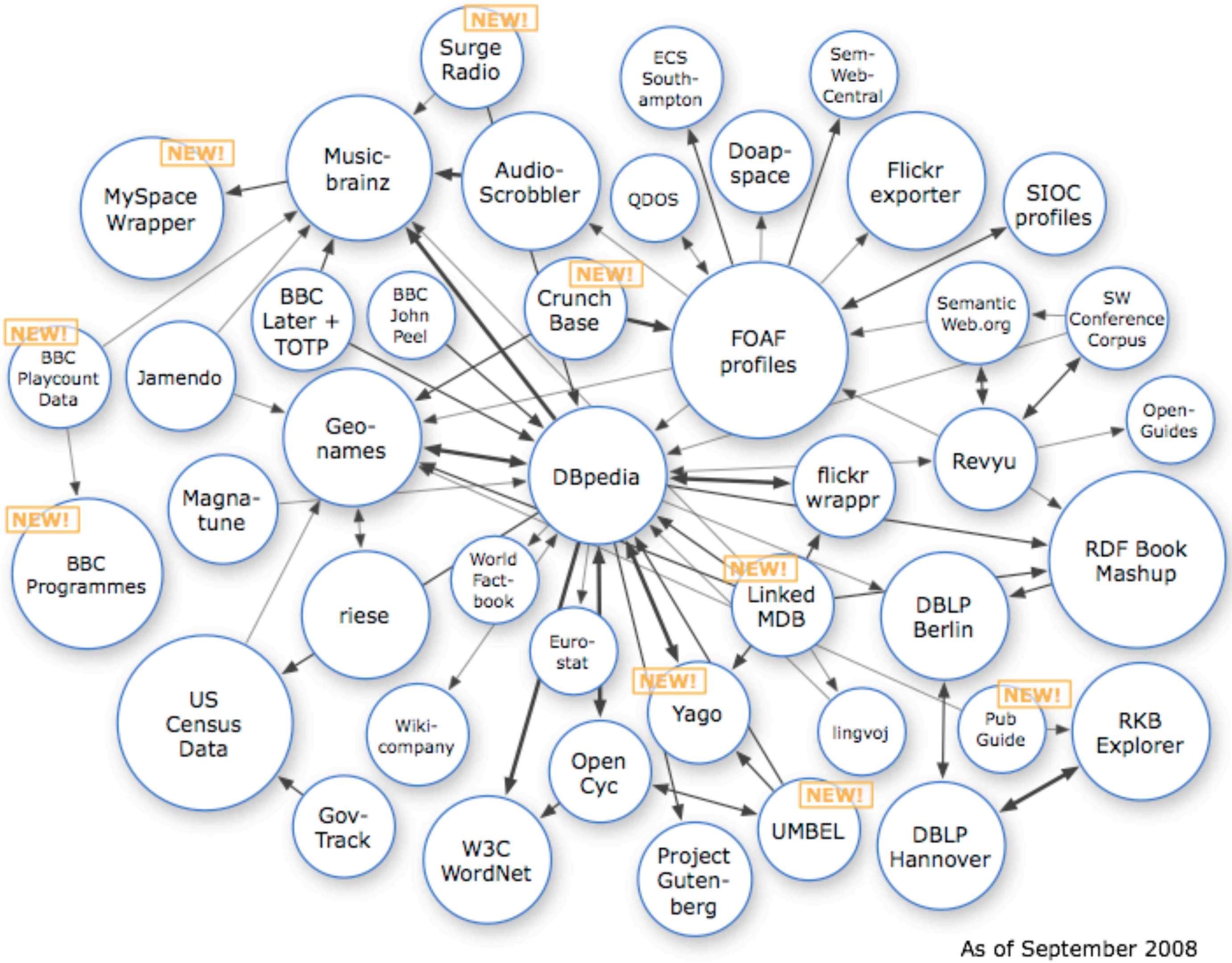
Adapted from EUCLID learning materials – "Introduction to Linked Data"

Evolution of the Linked Data Cloud: 2007

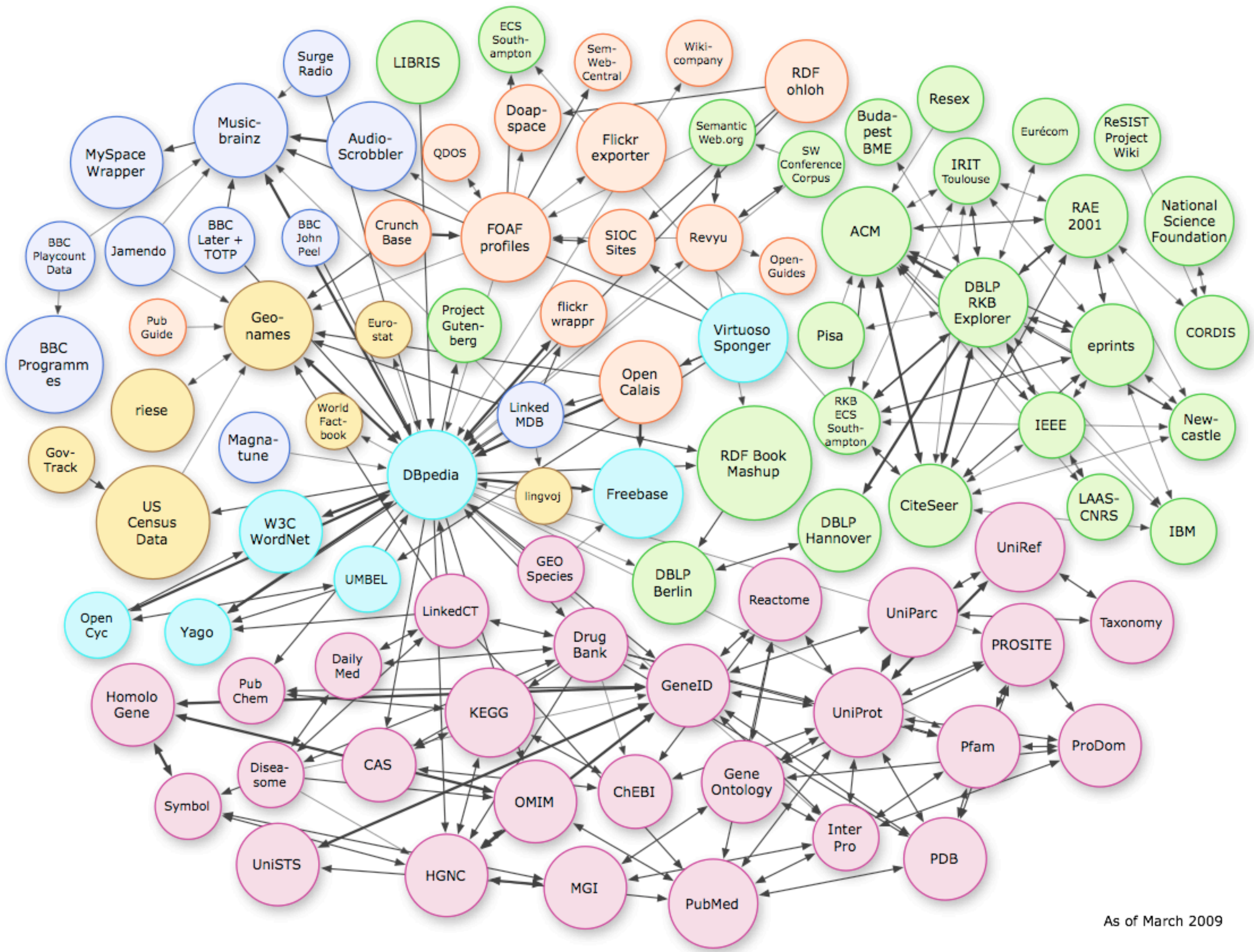


http://lod-cloud.net

Evolution of the Linked Data Cloud: 2008

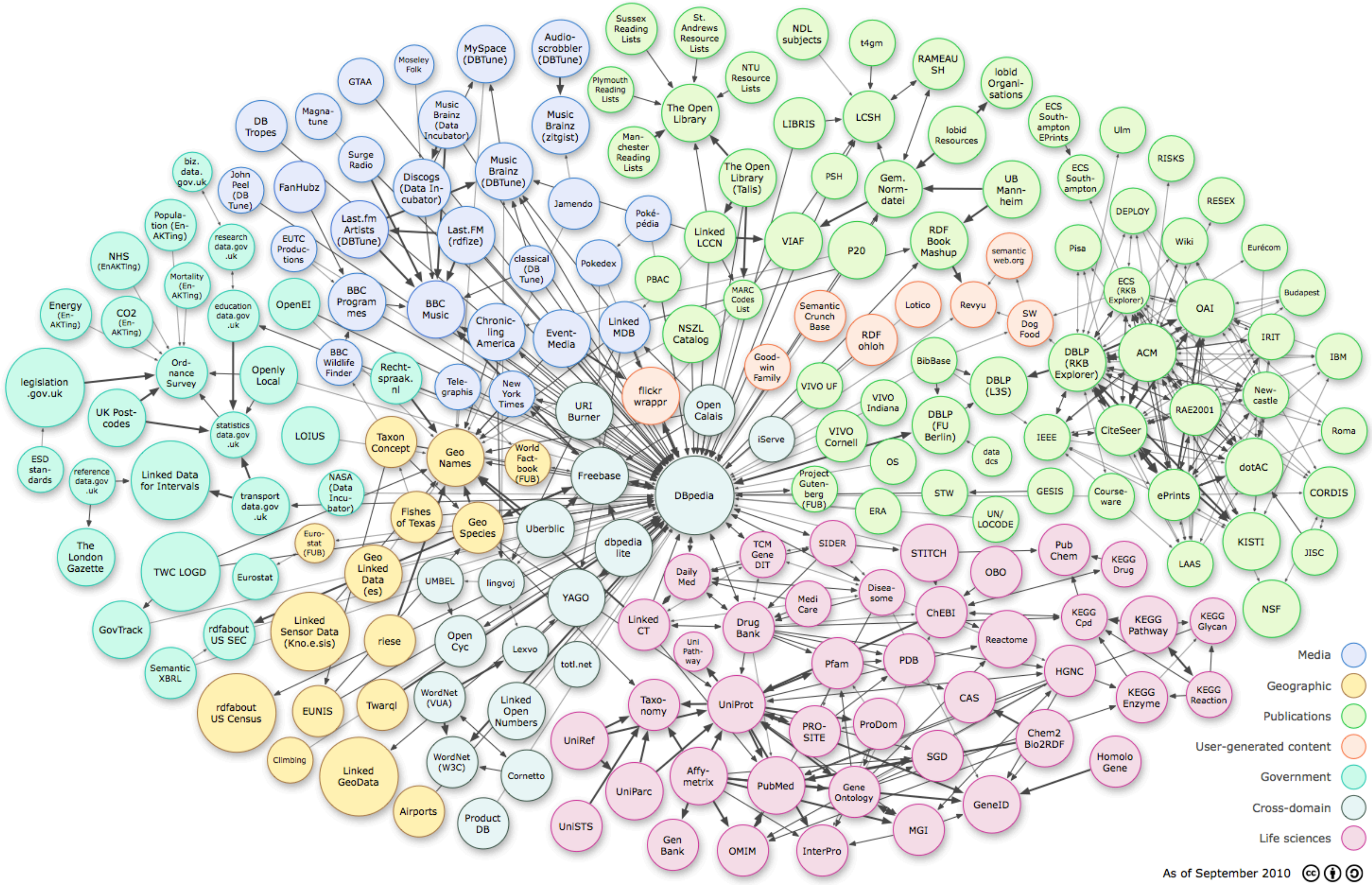


Evolution of the Linked Data Cloud: 2009



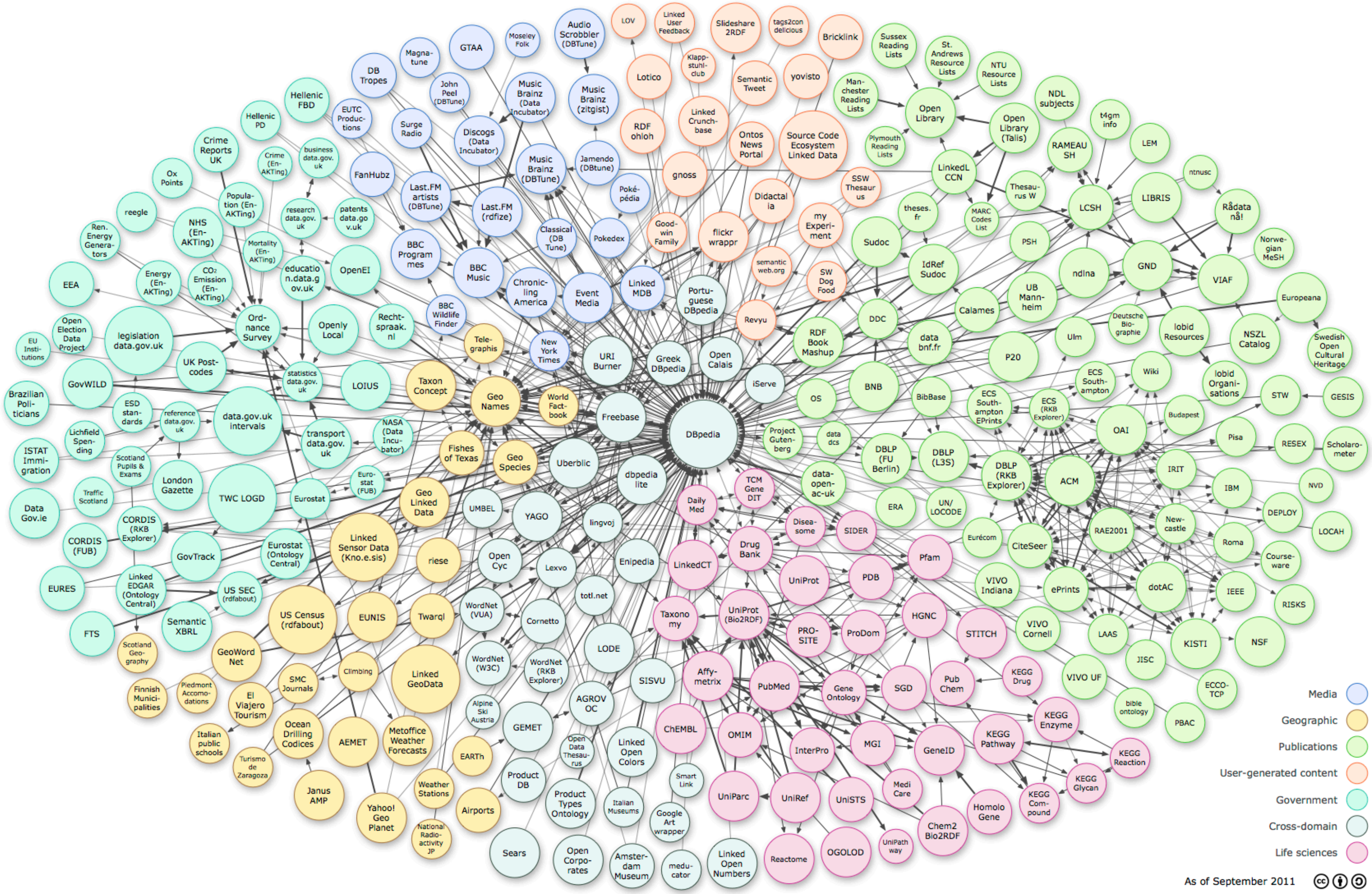
http://lod-cloud.net

Evolution of the Linked Data Cloud: 2010

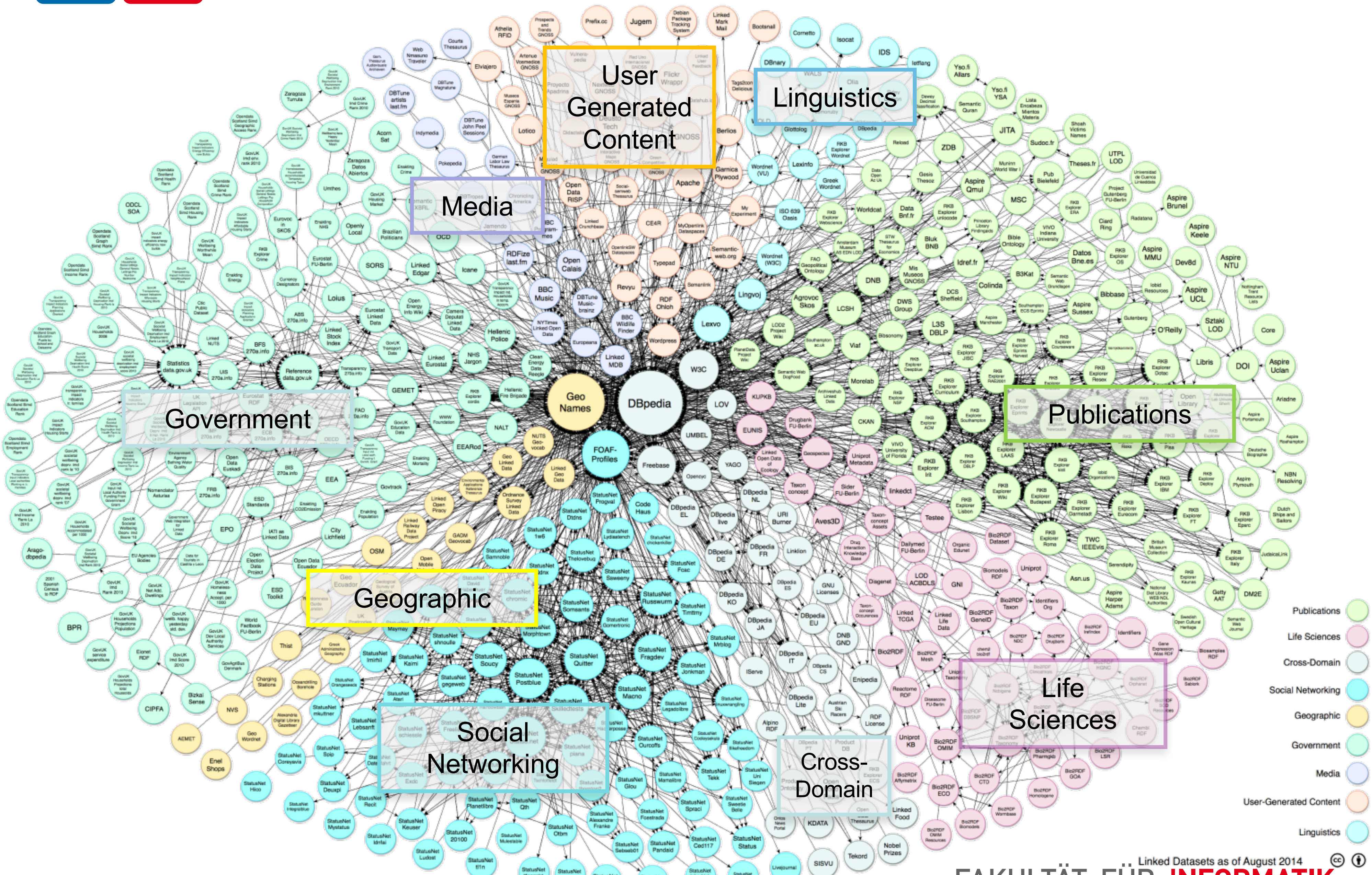


http://lod-cloud.net

Evolution of the Linked Data Cloud: 2011



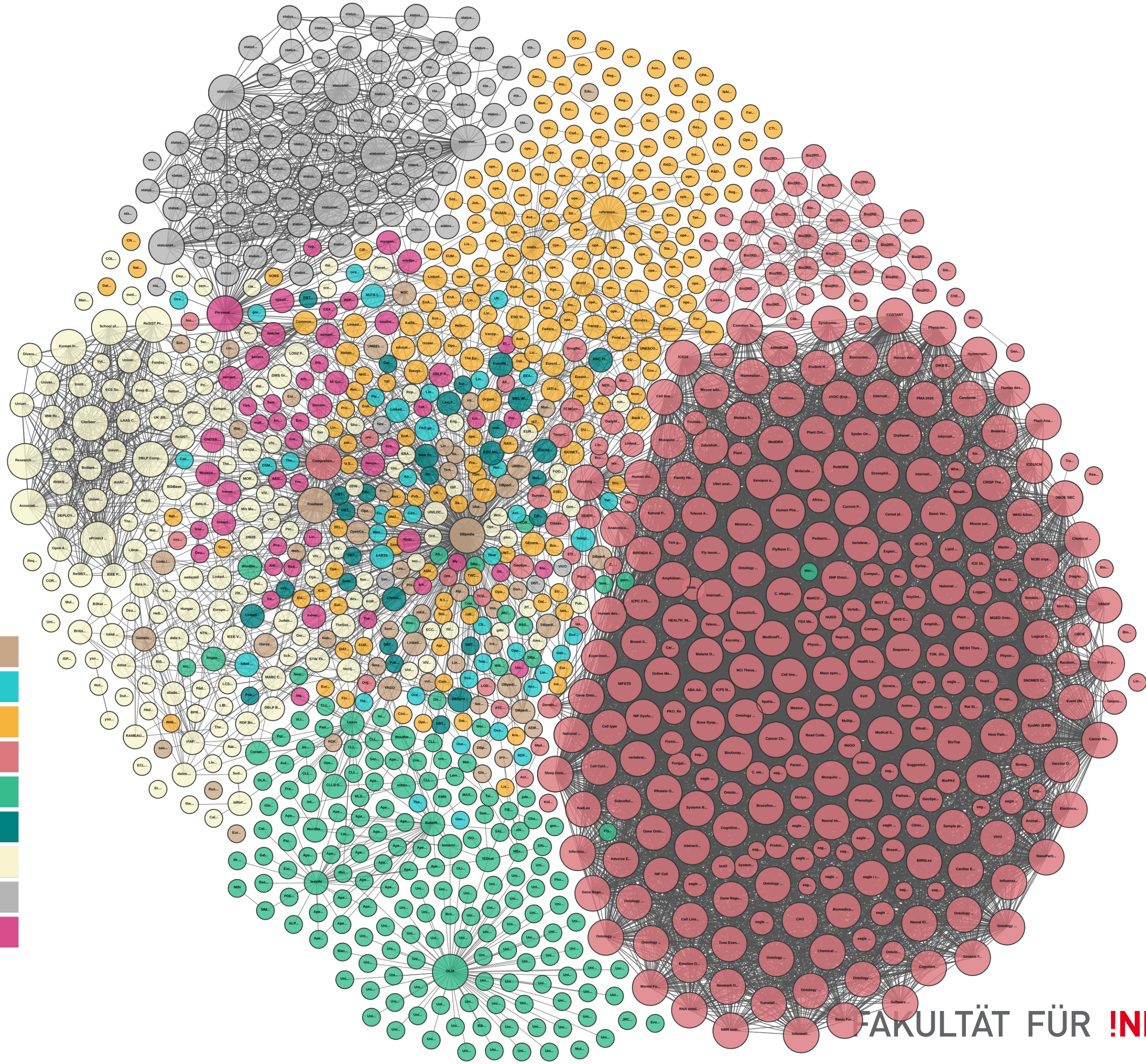
Evolution of the Linked Data Cloud: 8/2014



http://lod-cloud.net

FAKULTÄT FÜR !NFORMATIK

Evolution of the Linked Data Cloud: 2017



"Linking Open Data cloud diagram 2017, by Andrejs Abele, John P. McCrae, Paul Buitelaar, Anja Jentzsch and Richard Cyganiak. http://lod-cloud.net/"

Legend

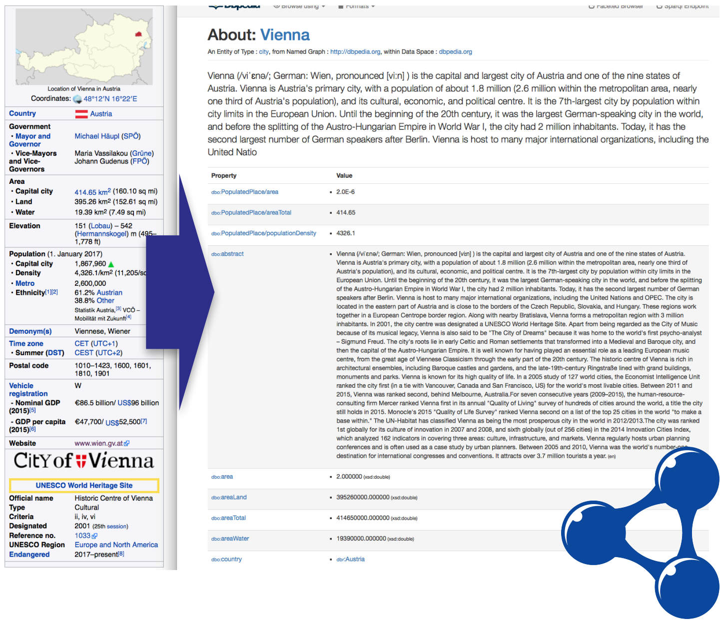
- Cross Domain
- Geography
- Government
- Life Sciences
- Linguistics
- Media
- Publications
- Social Networking
- User Generated
- Incoming Links
- Outgoing Links

http://lod-cloud.net

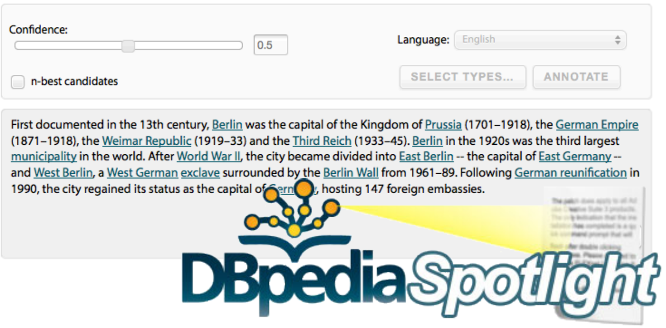
FAKULTÄT FÜR !NFORMATIK

Linked Data Applications: DBpedia

- DBpedia extracts structured information from Wikipedia and publishes it as a Linked Data graph
- "Nucleus" and linking hub of the web of Linked Data



Many Applications ..

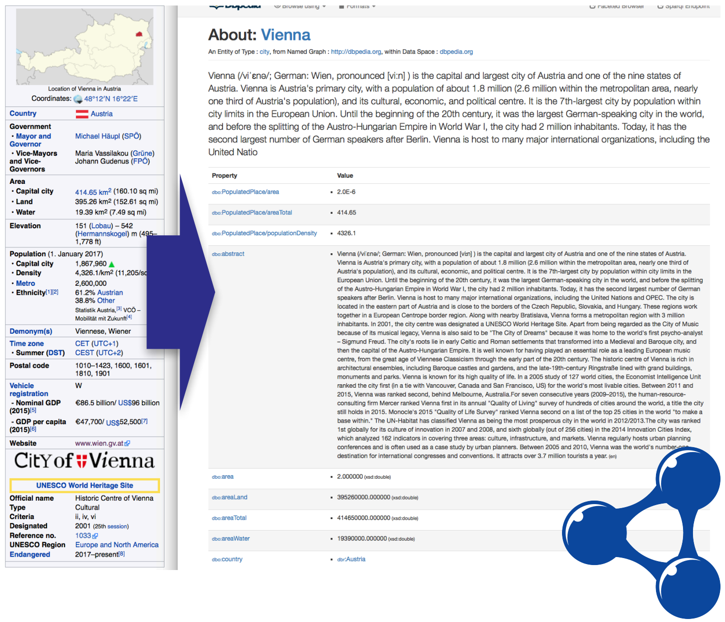


http://dbpedia-spotlight.github.io/demo/

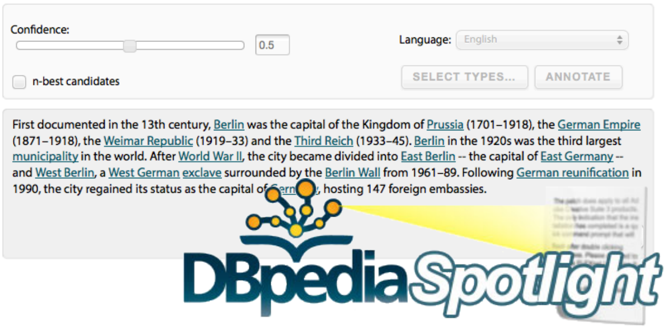
DBpedia spotlight automatically spots and disambiguates words or phrases and annotates them with DBPedia URIs

Linked Data Applications: DBpedia

- DBpedia extracts structured information from Wikipedia and publishes it as a Linked Data graph
- "Nucleus" and linking hub of the web of Linked Data



Many Applications ..



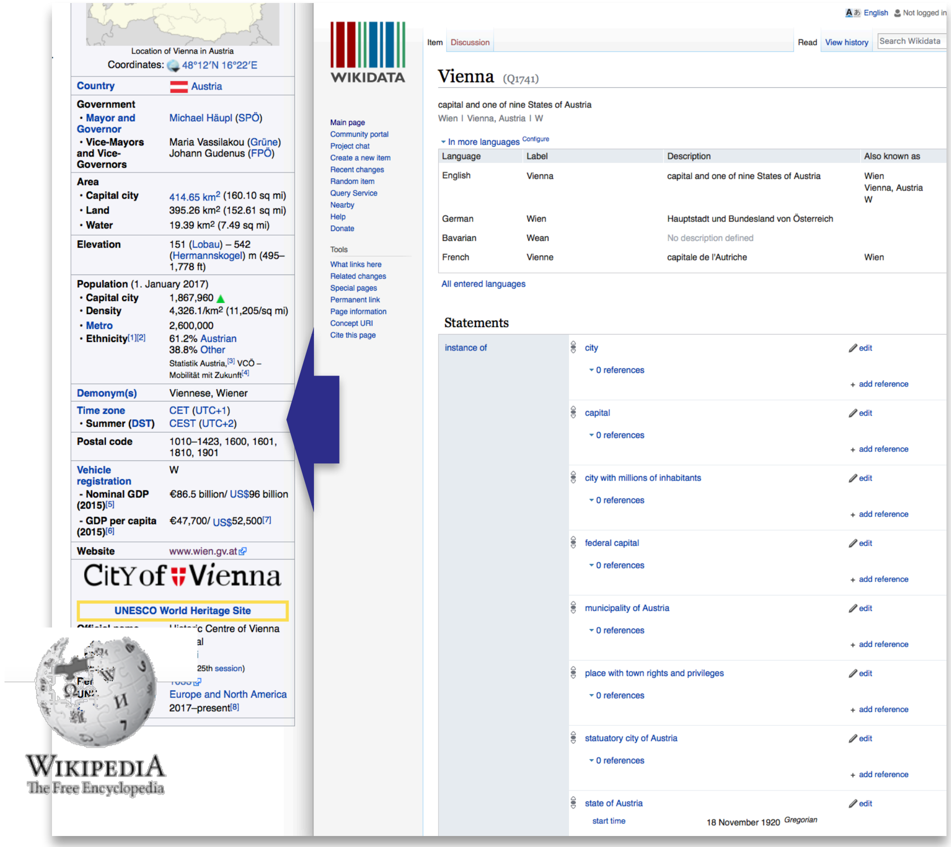
http://dbpedia-spotlight.github.io/demo/

LODlive and various other visualization tools can be used to visualize and explore the DBPedia (and other linked) data

http://en.lodlive.it/?http://dbpedia.org/resource/Wolfgang_Amadeus_Mozart

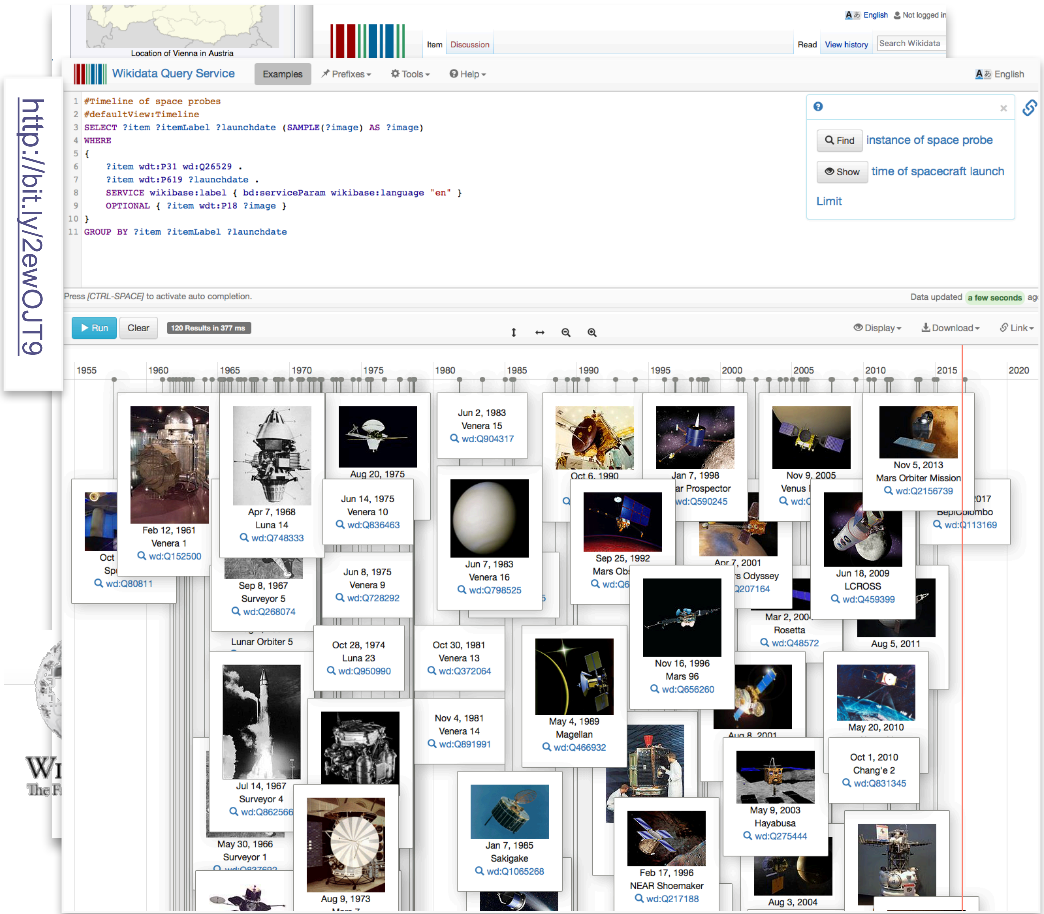
Linked Data Applications:

WIKIDATA



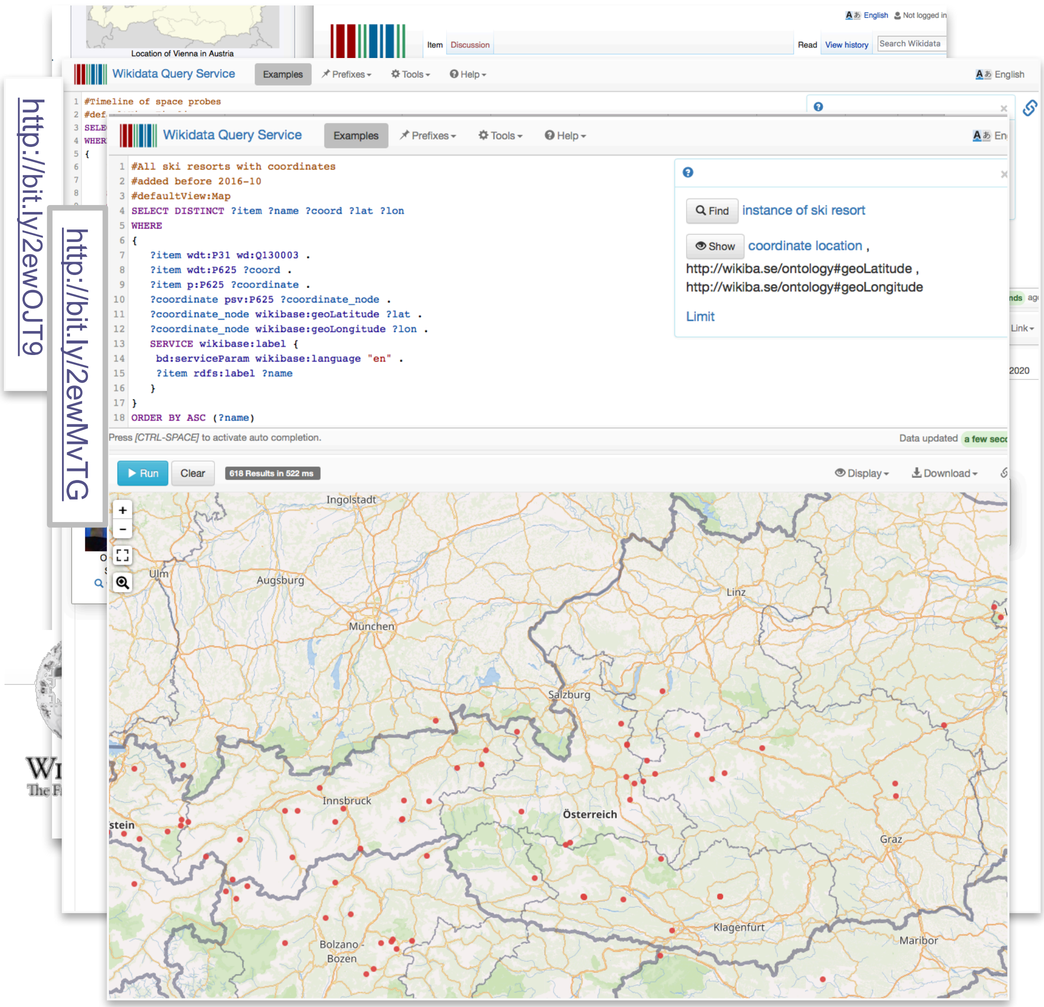
- User-curated source for structured information in Wikipedia
- Follows Linked Data principles (concept URIs)
- Provides a Linked data interface and SPARQL query service

Linked Data Applications: WIKIDATA



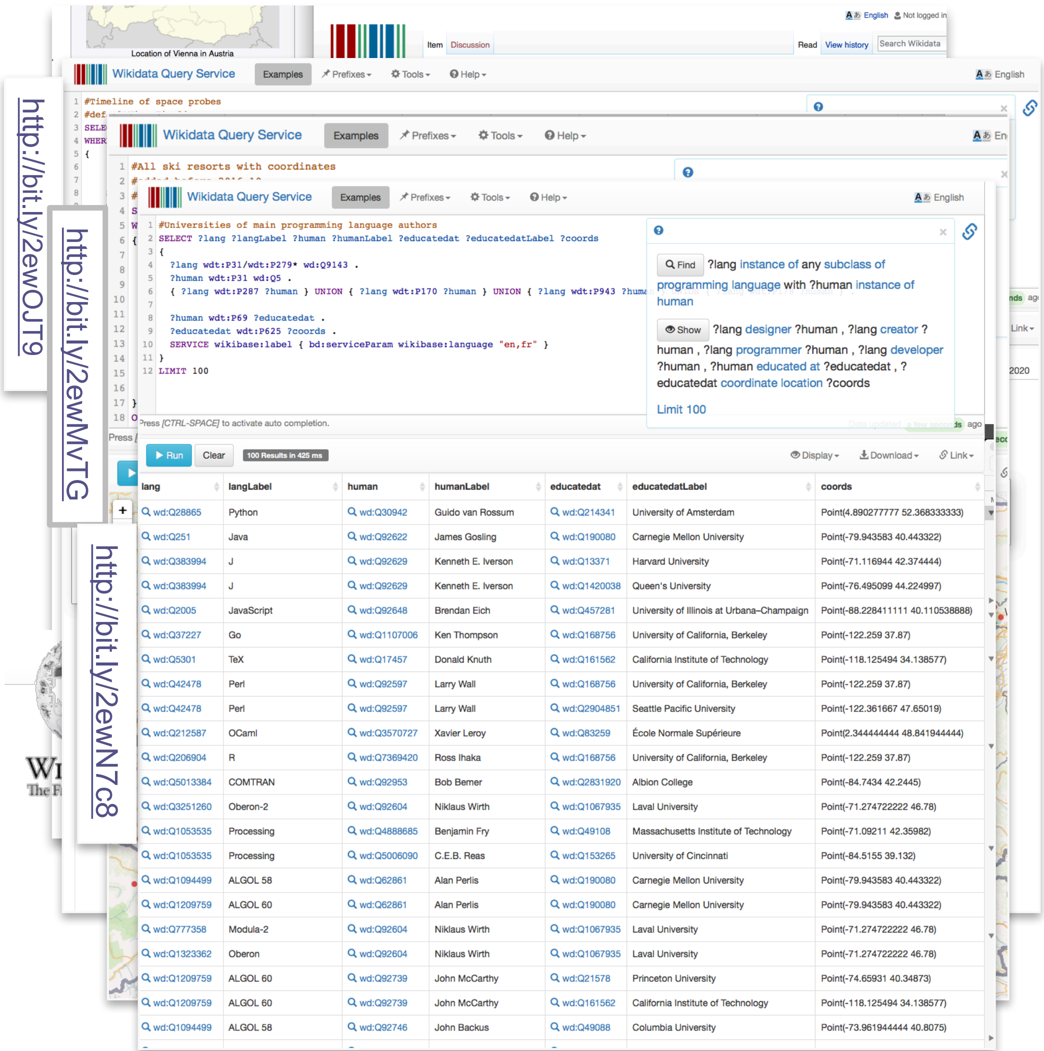
- User-curated source for structured information in Wikipedia
- Follows Linked Data principles (concept URIs)
- Provides a Linked data interface and SPARQL query service

Linked Data Applications: WIKIDATA



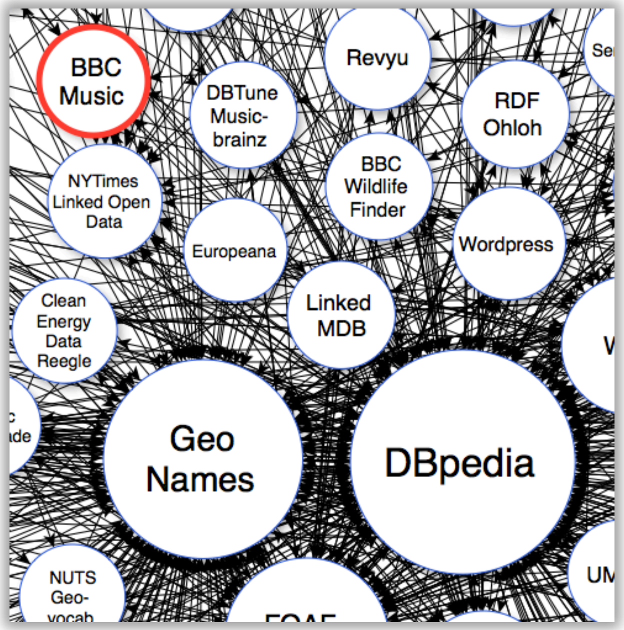
- User-curated source for structured information in Wikipedia
- Follows Linked Data principles (concept URIs)
- Provides a Linked data interface and SPARQL query service

Linked Data Applications:



- User-curated source for structured information in Wikipedia
- Follows Linked Data principles (concept URIs)
- Provides a Linked data interface and SPARQL query service

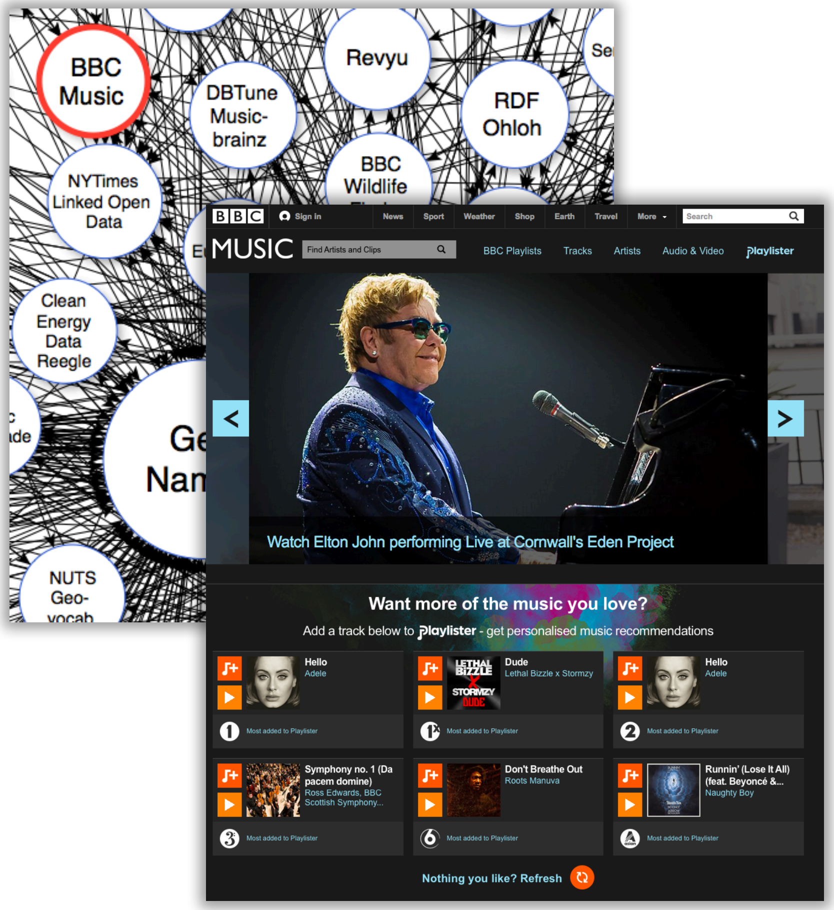
Linked Data Applications: BBC MUSIC



- Information is dynamically aggregated from external, publicly available data (Wikipedia, MusicBrainz, Discogs,..)
- No screen scraping
- No specialized API
- Data access via simple HTTP request
- Data is always up-to-date without manual interaction
- All data available as Linked Open Data

Adapted from Lecture Slides Semantic Web Technologies, Dr. Harald Sack, Hasso-Plattner-Institut, Potsdam

Linked Data Applications: BBC MUSIC



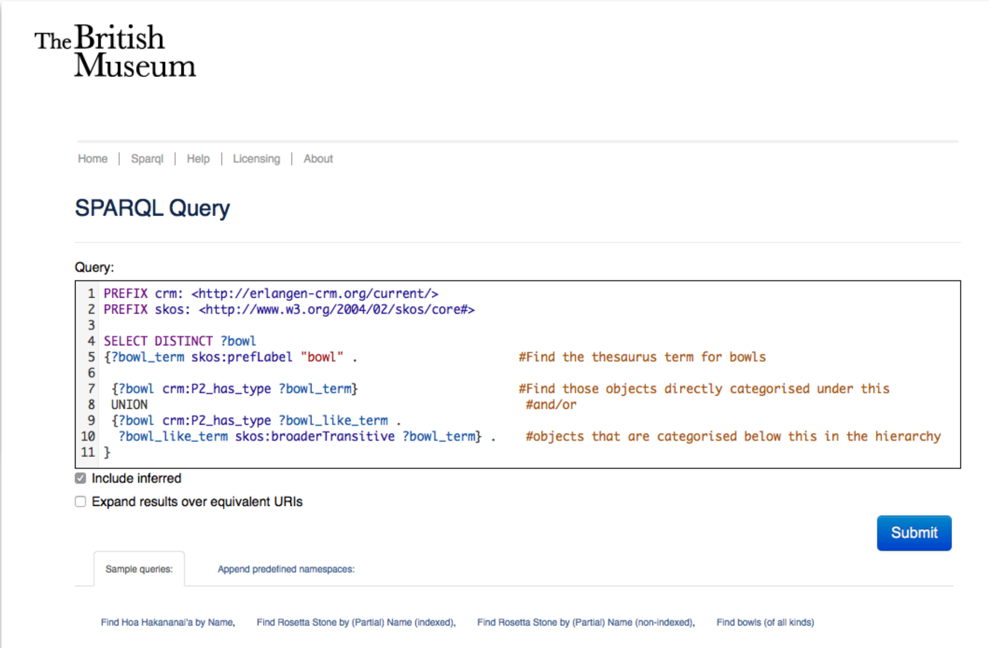
http://bbc.co.uk/music

- Information is dynamically aggregated from external, publicly available data (Wikipedia, MusicBrainz, Discogs,..)
- No screen scraping
- No specialized API
- Data access via simple HTTP request
- Data is always up-to-date without manual interaction
- All data available as Linked Open Data

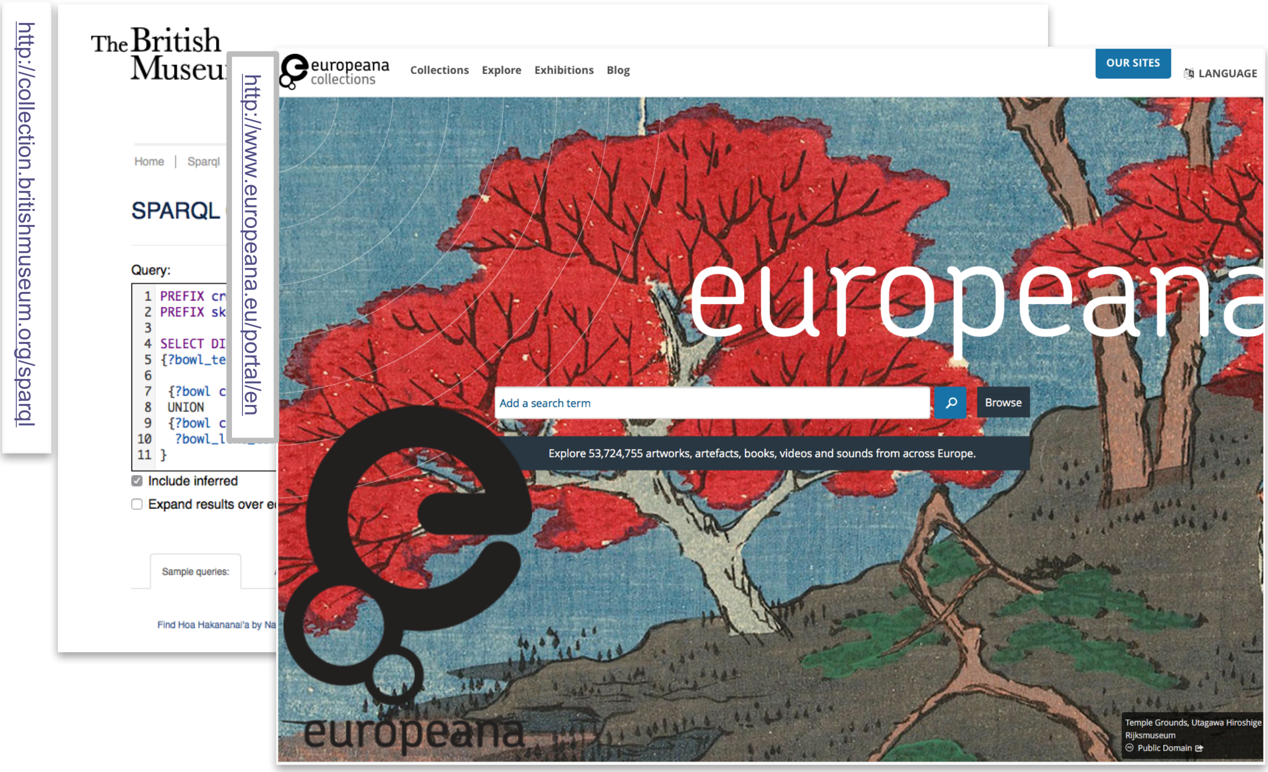
Linked Data Applications: Libraries and Museums

Linked Data Applications: Libraries and Museums

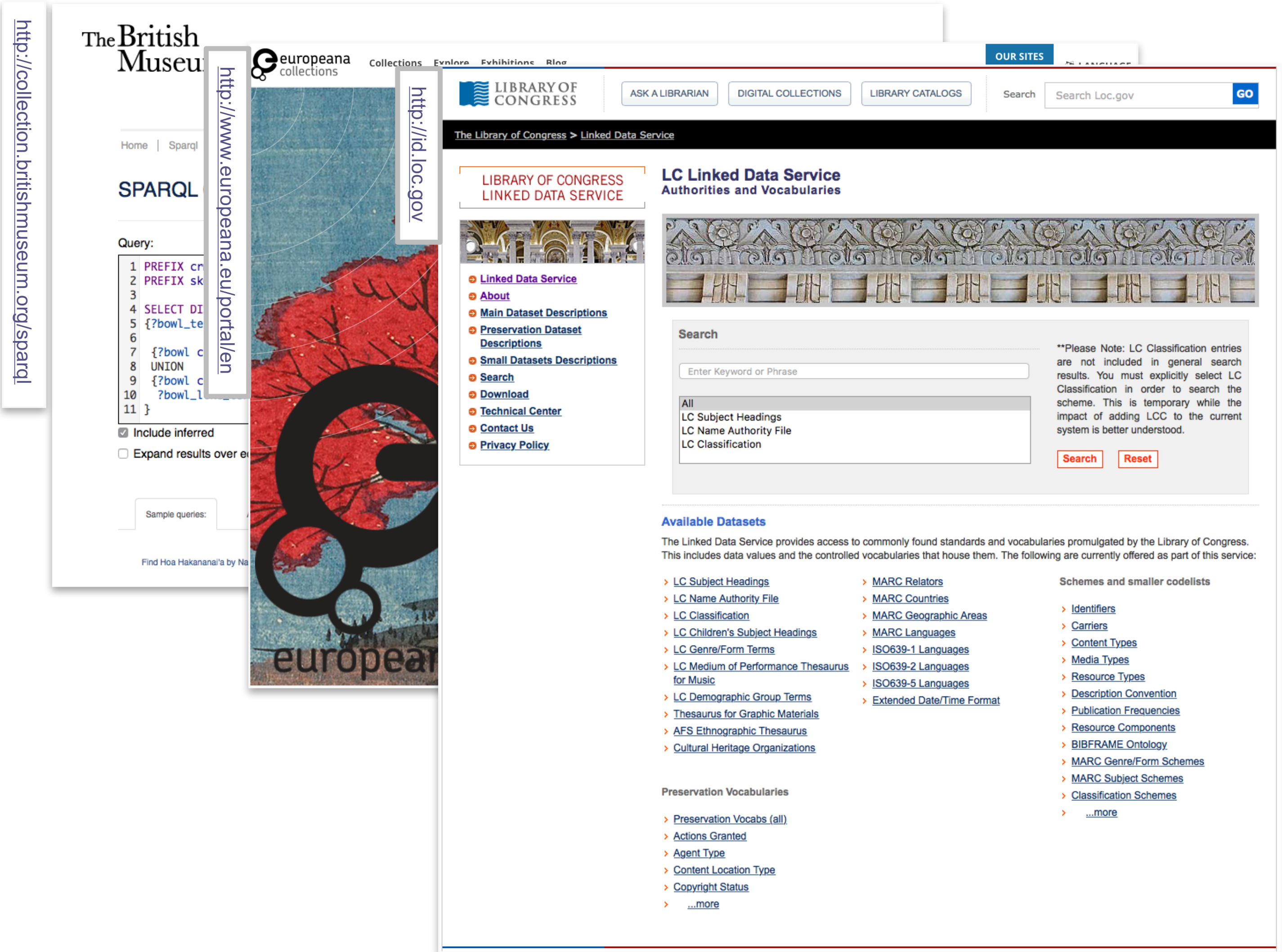
http://collection.britishmuseum.org/sparql



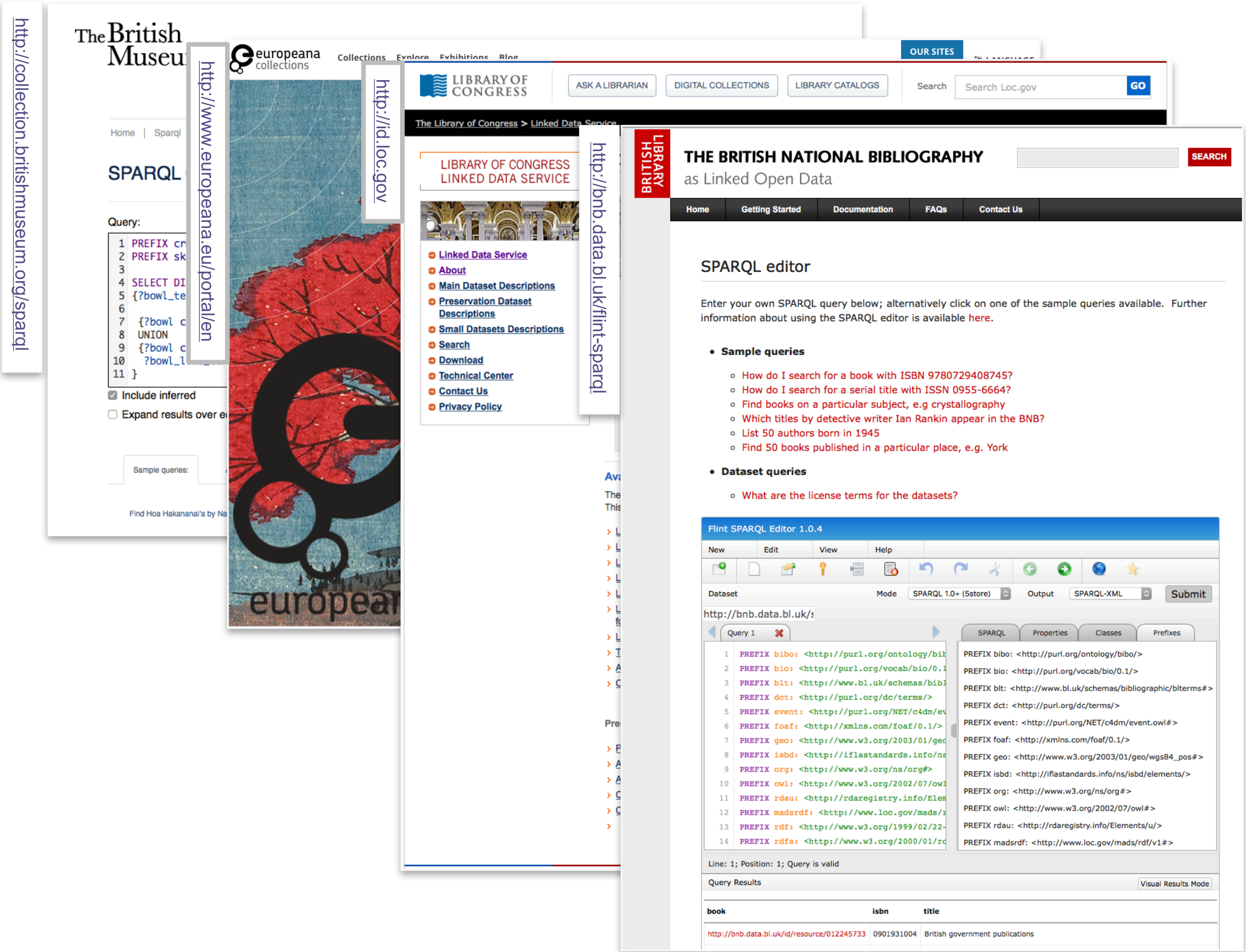
Linked Data Applications: Libraries and Museums



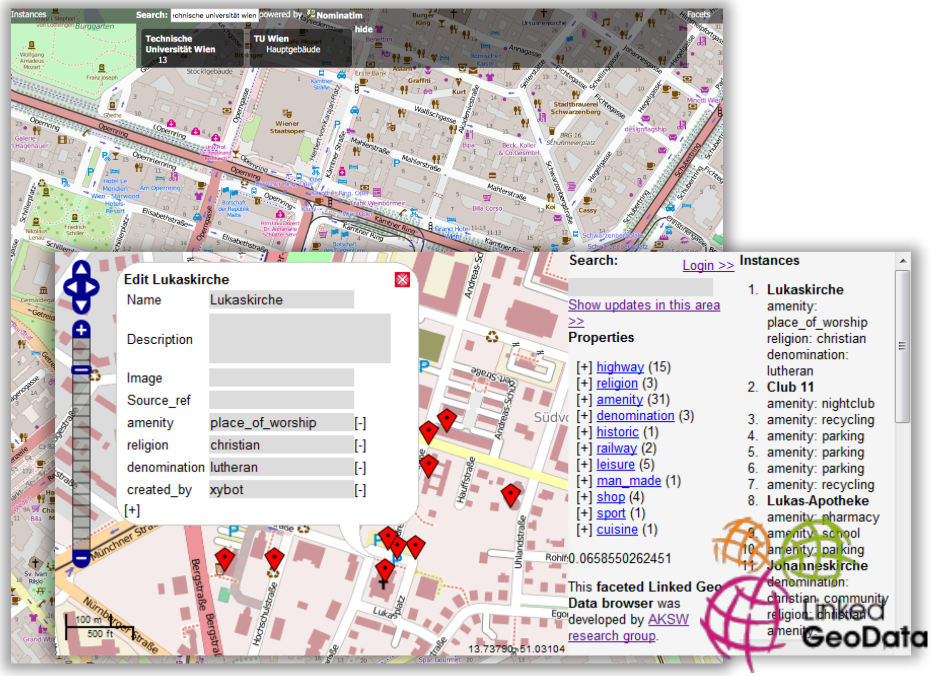
Linked Data Applications: Libraries and Museums



Linked Data Applications: Libraries and Museums

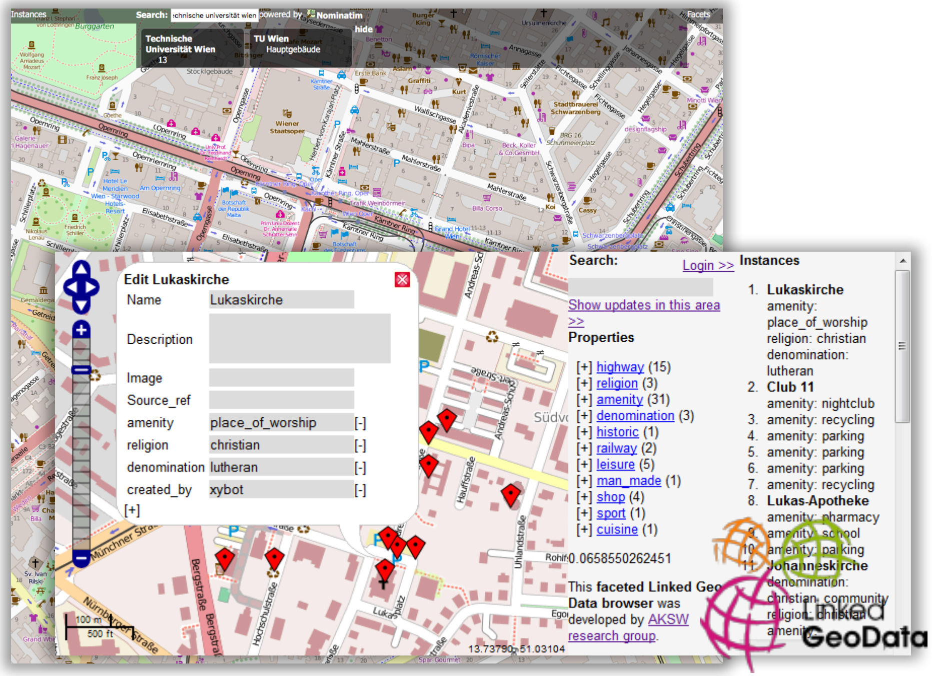


Linked Data Applications: Geospatial Data

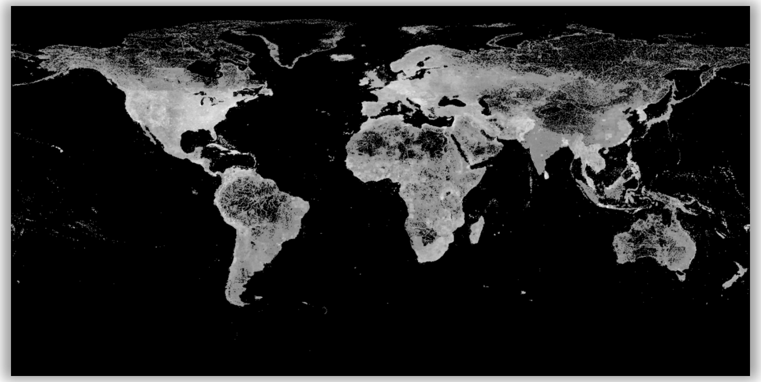


- Spatial knowledge base built from OpenStreetMap data
- 3+ billion nodes and 300 mio. ways → 20 bio triples
- Interlinked with DBpedia and GeoNames

Linked Data Applications: Geospatial Data



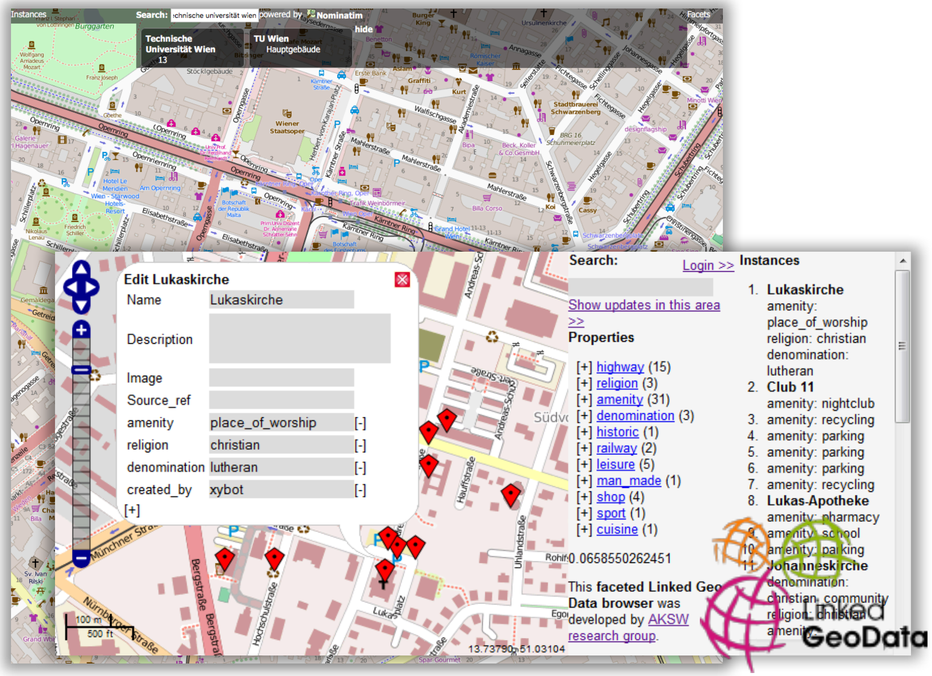
- Spatial knowledge base built from OpenStreetMap data
- 3+ billion nodes and 300 mio. ways → 20 bio triples
- Interlinked with DBpedia and GeoNames



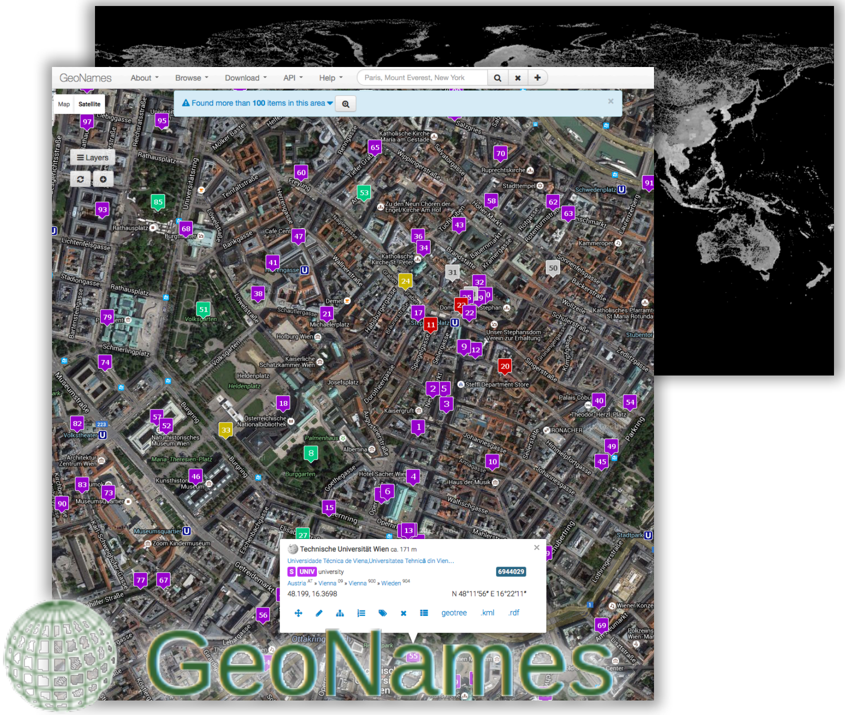
- Database of 10+ Mio geographical names in various languages
- Categorized into feature classes and codes
- Organized in Ontology
- Interlinked with DBPedia and others

GeoNames

Linked Data Applications: Geospatial Data

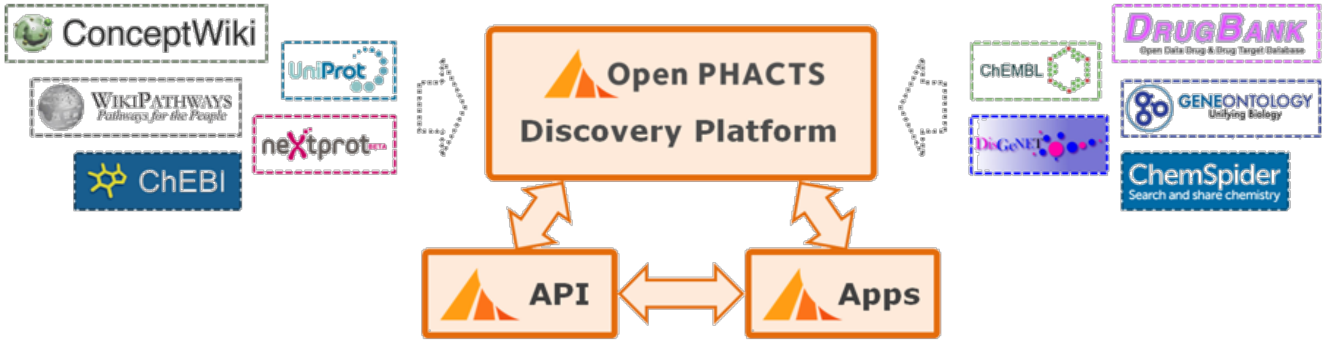


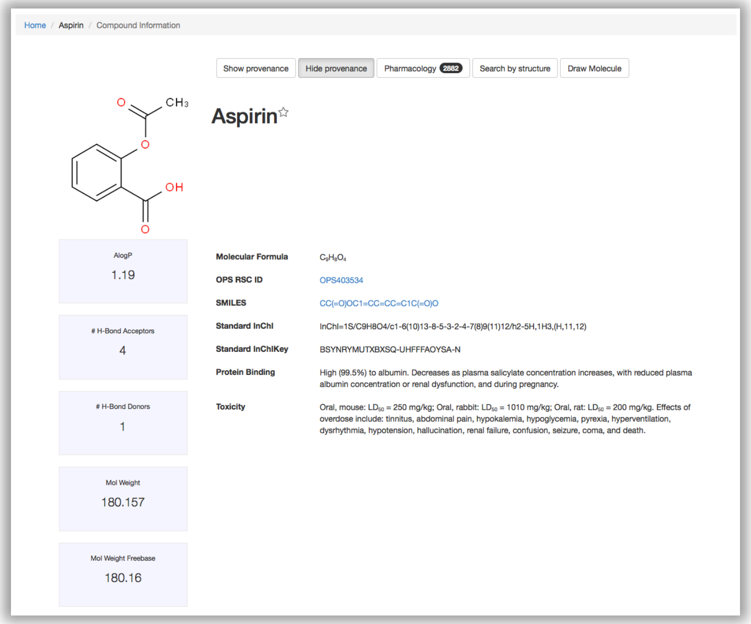
- Spatial knowledge base built from OpenStreetMap data
- 3+ billion nodes and 300 mio. ways → 20 bio triples
- Interlinked with DBpedia and GeoNames



- Database of 10+ Mio geographical names in various languages
- Categorized into feature classes and codes
- Organized in Ontology
- Interlinked with DBPedia and others

Linked Data Applications: OpenPhacts

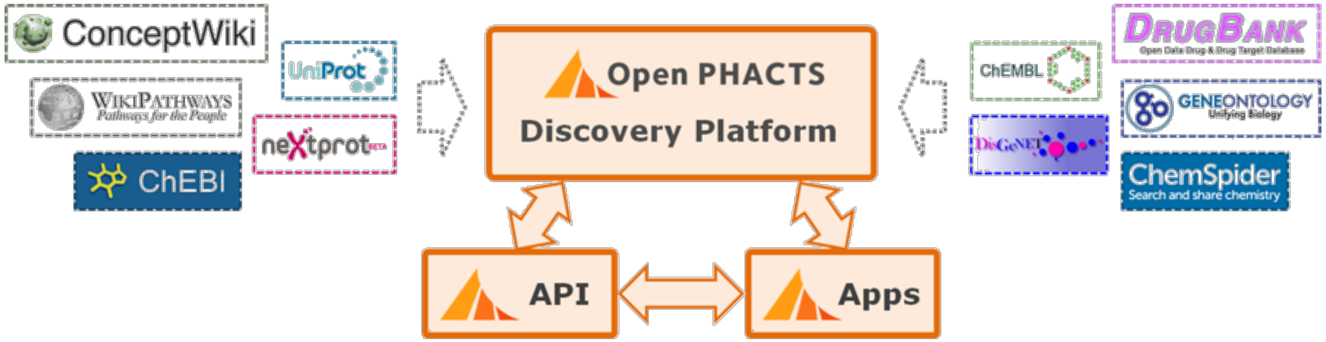


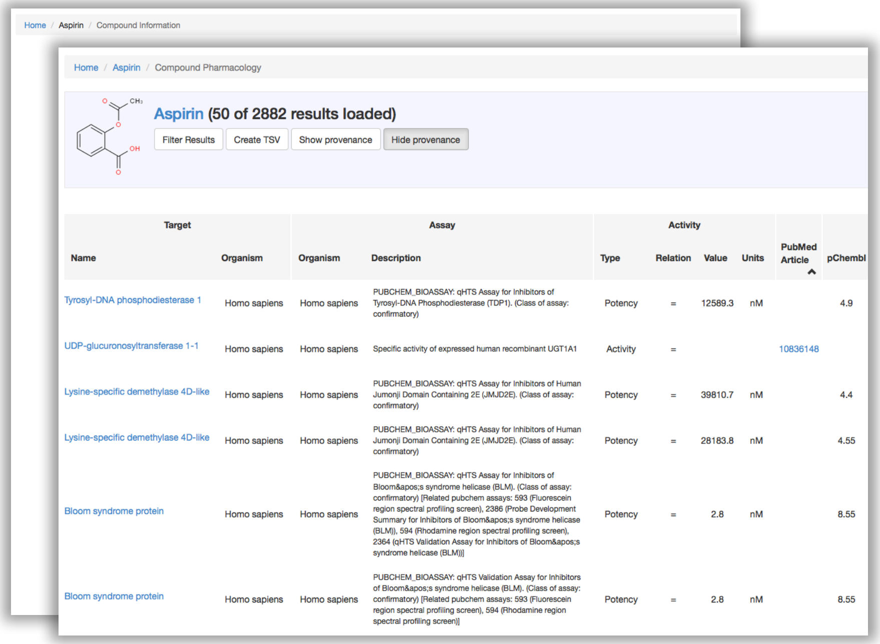


http://www.openphacts.org

- ‘Open pharmacological space’ using semantic web standards and technologies
- Goal: reduce barriers to drug discovery
- Semantic integration of multiple sources of publicly-available pharmacological and physicochemical data
- Intuitive querying interface to browse relationships between *compounds*, *targets*, *pathways*, *diseases* and *tissues*

Linked Data Applications: OpenPhacts





Home / Aspirin / Compound Information

Home / Aspirin / Compound Pharmacology

Aspirin (50 of 2882 results loaded)

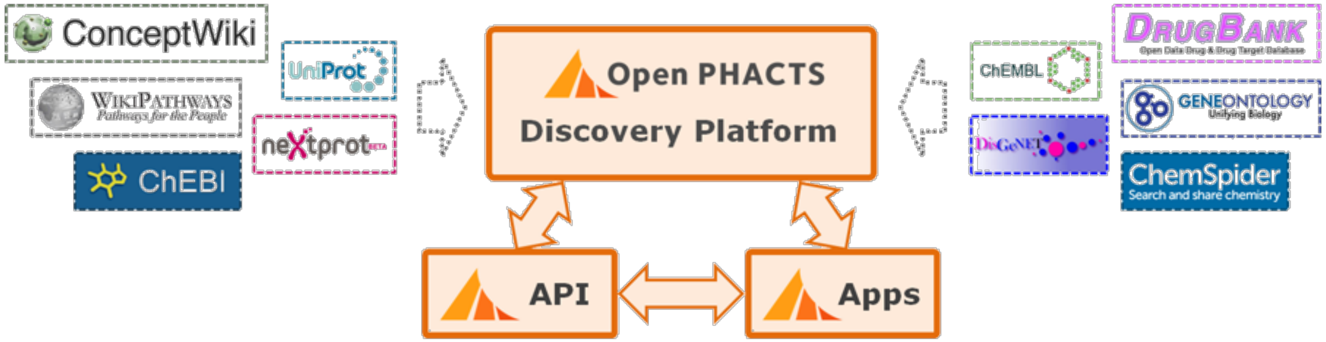
Filter Results | Create TSV | Show provenance | Hide provenance

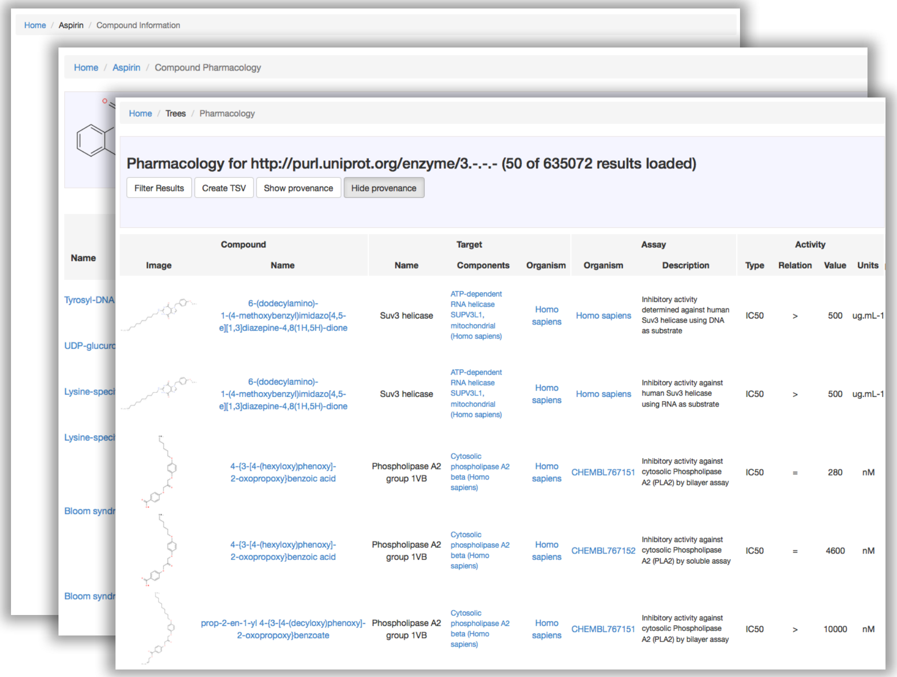
Target		Assay		Activity					
Name	Organism	Organism	Description	Type	Relation	Value	Units	PubMed Article	pChembl
Tyrosyl-DNA phosphodiesterase 1	Homo sapiens	Homo sapiens	PUBCHEM_BIOASSAY: qHTS Assay for Inhibitors of Tyrosyl-DNA Phosphodiesterase (TDP1). (Class of assay: confirmatory)	Potency	=	12589.3	nM		4.9
UDP-glucuronosyltransferase 1-1	Homo sapiens	Homo sapiens	Specific activity of expressed human recombinant UGT1A1	Activity	=			10836148	
Lysine-specific demethylase 4D-like	Homo sapiens	Homo sapiens	PUBCHEM_BIOASSAY: qHTS Assay for Inhibitors of Human Jumonji Domain Containing 2E (JMJD2E). (Class of assay: confirmatory)	Potency	=	39810.7	nM		4.4
Lysine-specific demethylase 4D-like	Homo sapiens	Homo sapiens	PUBCHEM_BIOASSAY: qHTS Assay for Inhibitors of Human Jumonji Domain Containing 2E (JMJD2E). (Class of assay: confirmatory)	Potency	=	28183.8	nM		4.55
Bloom syndrome protein	Homo sapiens	Homo sapiens	PUBCHEM_BIOASSAY: qHTS Assay for Inhibitors of Bloom's syndrome helicase (BLM). (Class of assay: confirmatory) [Related pubchem assays: 593 (Fluorescein region spectral profiling screen), 2386 (Probe Development Summary for Inhibitors of Bloom's syndrome helicase (BLM)), 594 (Rhodamine region spectral profiling screen), 2364 (qHTS Validation Assay for Inhibitors of Bloom's syndrome helicase (BLM))]	Potency	=	2.8	nM		8.55
Bloom syndrome protein	Homo sapiens	Homo sapiens	PUBCHEM_BIOASSAY: qHTS Validation Assay for Inhibitors of Bloom's syndrome helicase (BLM). (Class of assay: confirmatory) [Related pubchem assays: 593 (Fluorescein region spectral profiling screen), 594 (Rhodamine region spectral profiling screen)]	Potency	=	2.8	nM		8.55

http://www.openphacts.org

- ‘Open pharmacological space’ using semantic web standards and technologies
- Goal: reduce barriers to drug discovery
- Semantic integration of multiple sources of publicly-available pharmacological and physicochemical data
- Intuitive querying interface to browse relationships between *compounds*, *targets*, *pathways*, *diseases* and *tissues*

Linked Data Applications: OpenPhacts

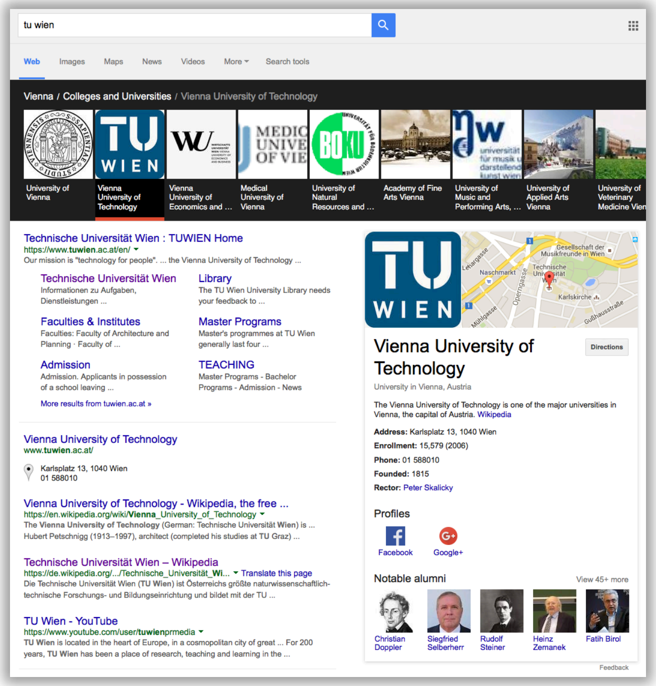




http://www.openphacts.org

- 'Open pharmacological space' using semantic web standards and technologies
- Goal: reduce barriers to drug discovery
- Semantic integration of multiple sources of publicly-available pharmacological and physicochemical data
- Intuitive querying interface to browse relationships between *compounds*, *targets*, *pathways*, *diseases* and *tissues*

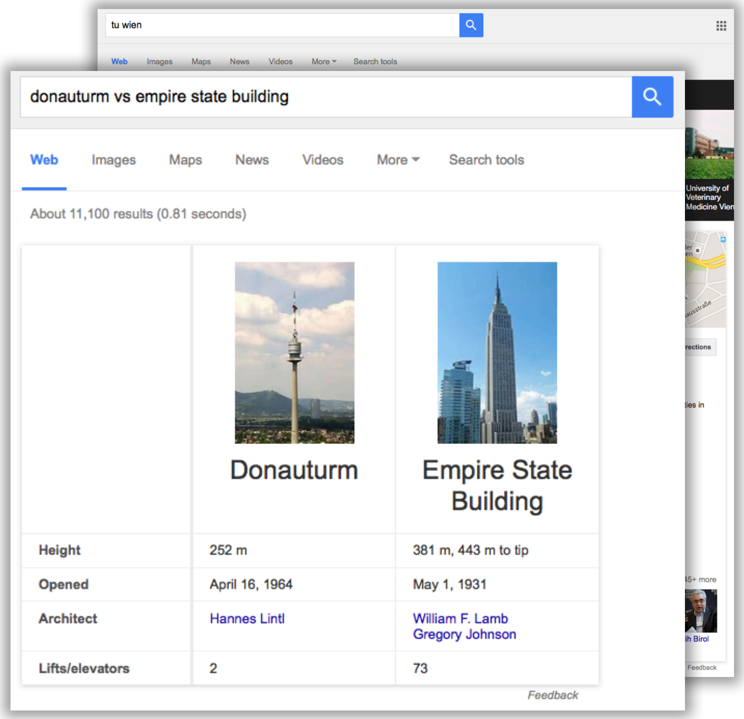
Industry Applications: Knowledge graphs



Google

- Entity search results
- Understand and answer queries
- Schema.org (markup on > 2.5 billion pages as of 2016)

Industry Applications: Knowledge graphs



Google

- Entity search results
- Understand and answer queries
- Schema.org (markup on > 2.5 billion pages as of 2016)

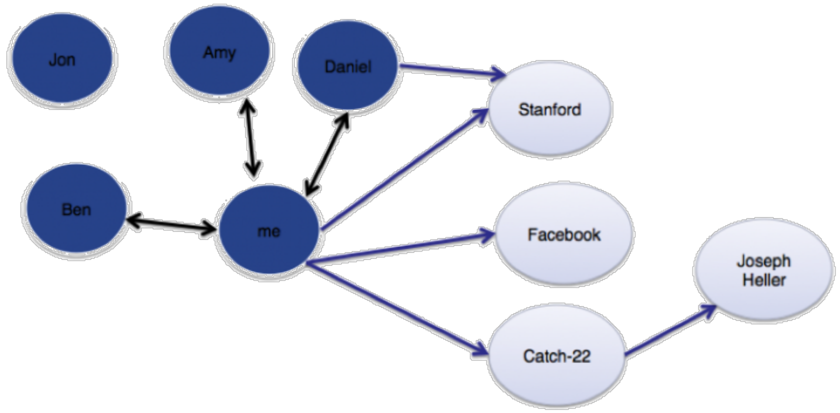
Industry Applications: Knowledge graphs

Google

- Entity search results
- Understand and answer queries
- Schema.org (markup on > 2.5 billion pages as of 2016)

Industry Applications: Knowledge graphs

- Entity graph: 100+Bill. connections between people, places and interests
- Initially extracted from Wikipedia for “fallback pages” (e.g., unclaimed community sites)
- Now serves as a semantic backbone to capture users’ interests
- Facebook Open Graph protocol (RDFa)



Google

- Entity search results
- Understand and answer queries
- Schema.org (markup on > 2.5 billion pages as of 2016)

Linked Data vs Knowledge Graphs

Overlaps:

- Knowledge Graphs also represent explicit semantics in a graph-based data model
- Both are frequently used to facilitate semantic search
- Knowledge graphs can use open standards (e.g., RDFa)

Key differences:

- Proprietary (data and technologies), closed "ecosystem"
- Tightly integrated with services
- Typically not published externally → no way to link to

Linked Data vs. “The” Semantic Web

Overlaps:

- *"LD is the Semantic Web done right"* (Tim Berners-Lee)
- Semantic web is made up of Linked Data
- Linked Data is based on Semantic web standards

Key Differences:

- Semantic Web was all about "semantifying" the web, Linked Data is based on web standards (URIs, http), but doesn't center around web *pages*.
- LD is a more pragmatic "bottom-up" approach.
- *"Linked Data is mainly about publishing structured data in RDF using URIs rather than focusing on the ontological level or inference."*

M. Hausenblas "Exploiting Linked Data For Building Web Applications"
IEEE Internet Computing, 2009

Linked Data vs. Open Data

Overlaps:

- Openness is a core principle in the design of LD
- Many Linked Data sets published under an open license
  → Linked *Open* Data (LOD) and LD often used synonymously

Key differences:

- Linked Data technologies can be used without publishing data – e.g., for internal and external data integration.
- Not all open data will ever be linked (the majority will remain in formats such as csv, txt etc.)

Linked Data vs Big Data

Overlaps:

- LD as a whole is big (38.606.408.854 triples and counting! *)
- No rigid up-front (e.g., relational) data model
- Big Data technologies (e.g., Hadoop) are used to handle LD
- LD can represent knowledge extracted from big unstructured data

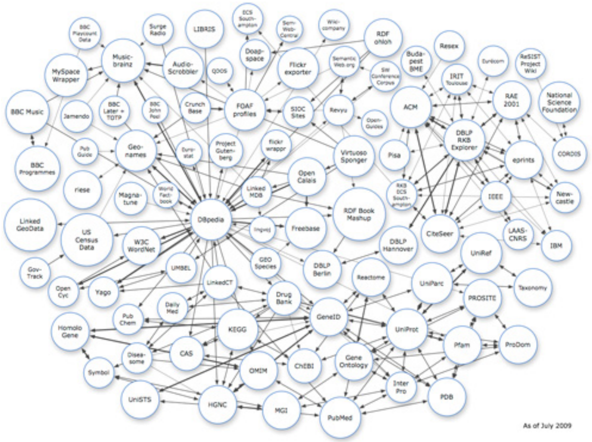
Key Differences:

- Individual linked data sets are typically not "big" per se
  (e.g., English DBPedia dump currently < 5 GB)
- LD is structured and semantically explicit,
  "big data lakes" are typically neither
- Big data based on distributed data infrastructures within an organization
  (e.g., Hadoop clusters), LD creates a decentralized, globally distributed data infrastructure

*) http://lodlaundromat.org as per 2016-05-10

Summary of key ideas

Summary of key ideas



Web of data

Linking

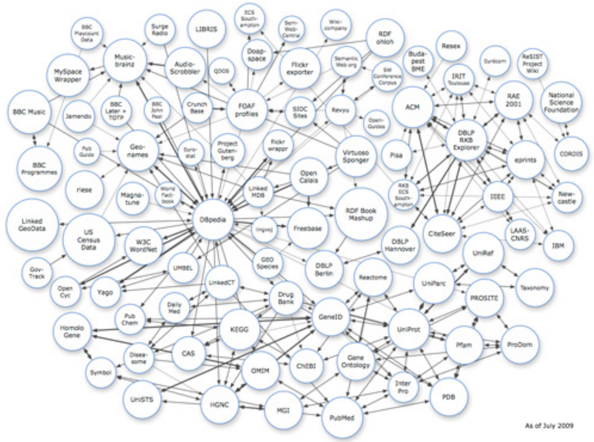
Open

Global data space

Network effects

Decentralized

Summary of key ideas



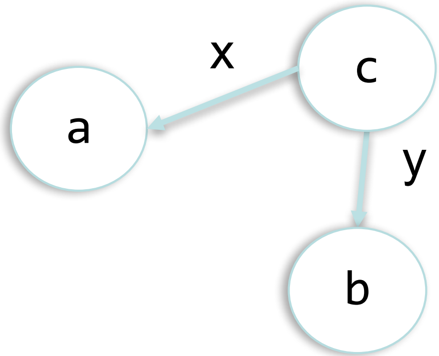
Web of data

Linking Open

Global data space

Network effects

Decentralized



Graph-based

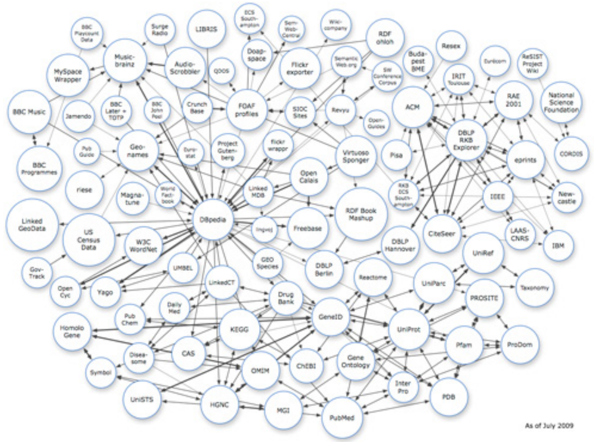
Flexible

Bottom-up

Emergent

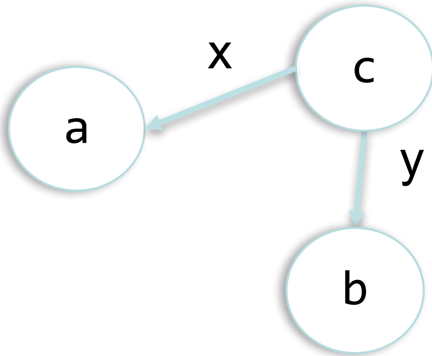
Agile

Summary of key ideas



Web of data

Linking
Open
Global data space
Network effects
Decentralized



Graph-based

Flexible
Bottom-up
Emergent
Agile

Explicit Semantics

Machine readable
Interoperable
Ad-hoc integration
Discovery
Inference

SO WHAT DOES ALL THAT HAVE TO DO WITH KEYWORD SEARCH?

Information Retrieval on the Web today

Keyword-based search is the dominant IR paradigm on the web today (Google, Yahoo, Bing,..)

→ Typically relatively high recall, but not all results
[due to synonyms and metaphors, missing context,..]

→ Typically low precision (i.e., many irrelevant results)

Problems:

- Highly sensitive to vocabulary
- Polysemy (ambiguity)
- Lack of context

Information extraction on the Web Today

- Typically no integration over multiple sites
  (search results list single pages)

- Software lacks
  - knowledge of contexts
  - world knowledge
  - experience

  ..to correctly perform many information extraction and integration tasks

Most of the web is difficult to interpret for machines because they

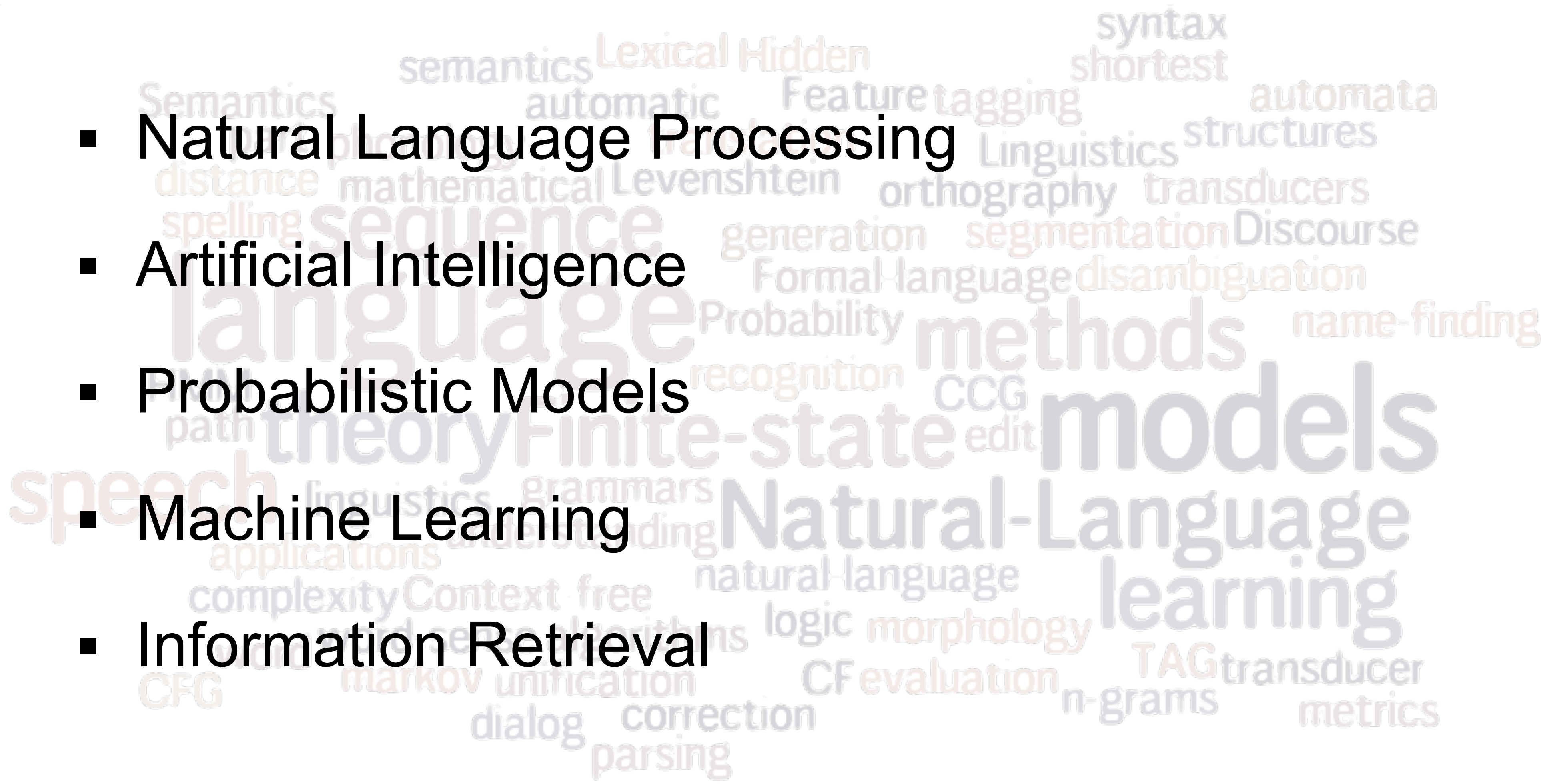
- cannot infer facts from partial information
- cannot leverage mental associations
- have limited tolerance for inconsistent terminology
- cannot process sensory information (e.g., visual, sound)
- …

Semantics?

- Semantics = “study of meaning” (from Greek σεμαντικος)
- Web content traditionally only understandable for humans
- Goal of the Semantic Web:
  Make the web machine-understandable

Fundamental problem:
understanding human language

“Understanding” language: Approach 1



- Natural Language Processing
- Artificial Intelligence
- Probabilistic Models
- Machine Learning
- Information Retrieval

Understanding language is difficult…

Understanding language is difficult…

Non-standard language

ph342 m9 1337 h4xX0r 5k!11Zz!!

Understanding language is difficult…

Non-standard language

ph342 m9 1337 h4xX0r 5k!11Zz!!

Neologisms

binge-watch, Obamacare, Brexit, Trumpism..

Adapted from Lecture Slides Semantic Web Technologies, Dr. Harald Sack, Hasso-Plattner-Institut, Potsdam

Understanding language is difficult…

Non-standard language

ph342 m9 1337 h4xX0r 5k!11Zz!!

Neologisms

binge-watch, Obamacare, Brexit, Trumpism..

Segmentation problems

Last night ***I shot*** *an elephant* ***in my pajamas****.*

Last night I shot ***an elephant in my pajamas****.*

Adapted from Lecture Slides Semantic Web Technologies, Dr. Harald Sack, Hasso-Plattner-Institut, Potsdam

Understanding language is difficult…

Non-standard language

ph342 m9 1337 h4xX0r 5k!11Zz!!

Neologisms

binge-watch, Obamacare, Brexit, Trumpism..

Segmentation problems

*Last night **I shot** an elephant **in my pajamas**.*

*Last night I shot **an elephant in my pajamas**.*

Idioms

- *kick the bucket*
- *spill the beans*
- *an arm and a leg*
- *get cold feet*
- *lose face*
- *piece of cake*

Understanding language is difficult…

Non-standard language

ph342 m9 1337 h4xX0r 5k!11Zz!!

Neologisms

binge-watch, Obamacare, Brexit, Trumpism..

Tricky entity names

The New York John F. Kennedy airport

Segmentation problems

Last night ***I shot*** *an elephant* ***in my pajamas****.*

Last night I shot ***an elephant in my pajamas****.*

Idioms

- *kick the bucket*
- *spill the beans*
- *an arm and a leg*
- *get cold feet*
- *lose face*
- *piece of cake*

Understanding language is difficult…

Non-standard language

ph342 m9 1337 h4xX0r 5k!11Zz!!

Neologisms

binge-watch, Obamacare, Brexit, Trumpism..

Segmentation problems

Last night ***I shot*** *an elephant* ***in my pajamas****.*

Last night I shot ***an elephant in my pajamas****.*

Idioms

- *kick the bucket*
- *spill the beans*
- *an arm and a leg*
- *get cold feet*
- *lose face*
- *piece of cake*

Tricky entity names

The New York John F. Kennedy airport

Ambiguity

We saw her duck.

Adapted from Lecture Slides Semantic Web Technologies, Dr. Harald Sack, Hasso-Plattner-Institut, Potsdam

Understanding language is difficult…

Non-standard language

ph342 m9 1337 h4xX0r 5k!11Zz!!

Neologisms

binge-watch, Obamacare, Brexit, Trumpism..

Segmentation problems

Last night ***I shot*** *an elephant* ***in my pajamas****.*

Last night I shot ***an elephant in my pajamas****.*

Idioms

- *kick the bucket*
- *spill the beans*
- *an arm and a leg*
- *get cold feet*
- *lose face*
- *piece of cake*

Tricky entity names

The New York John F. Kennedy airport

Ambiguity

We saw her duck.

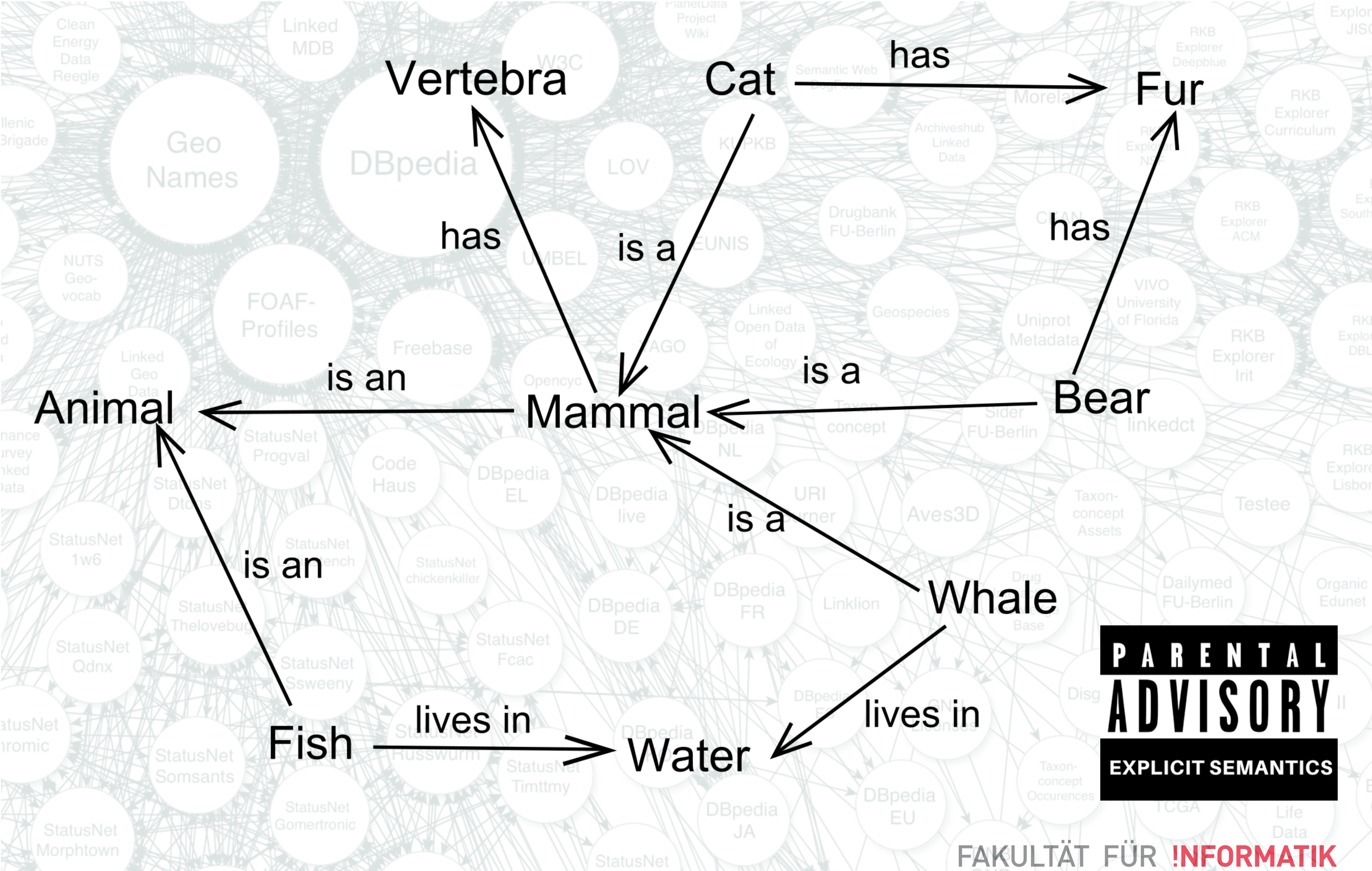
World knowledge

Mary and Sue are sisters.

Mary and Sue are mothers.

Adapted from Lecture Slides Semantic Web Technologies, Dr. Harald Sack, Hasso-Plattner-Institut, Potsdam

Approach 2: Explicit Semantics



Adapted from Lecture Slides Semantic Web Technologies, Dr. Harald Sack, Hasso-Plattner-Institut, Potsdam

Something to tinker with during the break..

You can browse DBpedia resources via the Linked Data frontend available at

`http://dbpedia.org/page/[resourceName]`

e.g. `dbr:Vienna` → [`http://dbpedia.org/page/Vienna`](http://dbpedia.org/page/Vienna)

Try to find out:

- *Where is TU Wien (`dbr:TU_Wien`) located?*
- *What is Marie Curie known for?*
- *Is the Danube river longer than the Volga river?*
- *Find a connection between `dbr:Vienna` and `dbr:Berlin`*
- *Find a connection between `dbr:The_Lord_of_the_Rings` and `dbr:Aldous_Huxley`*

References

Videos:

- Tim Berners-Lee: *The next Web of open, linked data* (TED talk)
  https://www.youtube.com/watch?v=OM6XIICm_qo
- Annenberg Networks Theory Seminar with Tim-Berners-Lee
  http://www.yovisto.com/video/10017
- Metaweb (now defunct): *Words vs entities*
  https://www.youtube.com/watch?v=TJfrNo3Z-DU
- Manu Sporny: Intro to the Semantic Web
  https://www.youtube.com/watch?v=OGg8A2zfWKg
- Manu Sporny: *What is linked data?*
  https://www.youtube.com/watch?v=4x_xzT5eF5Q

Tutorial:

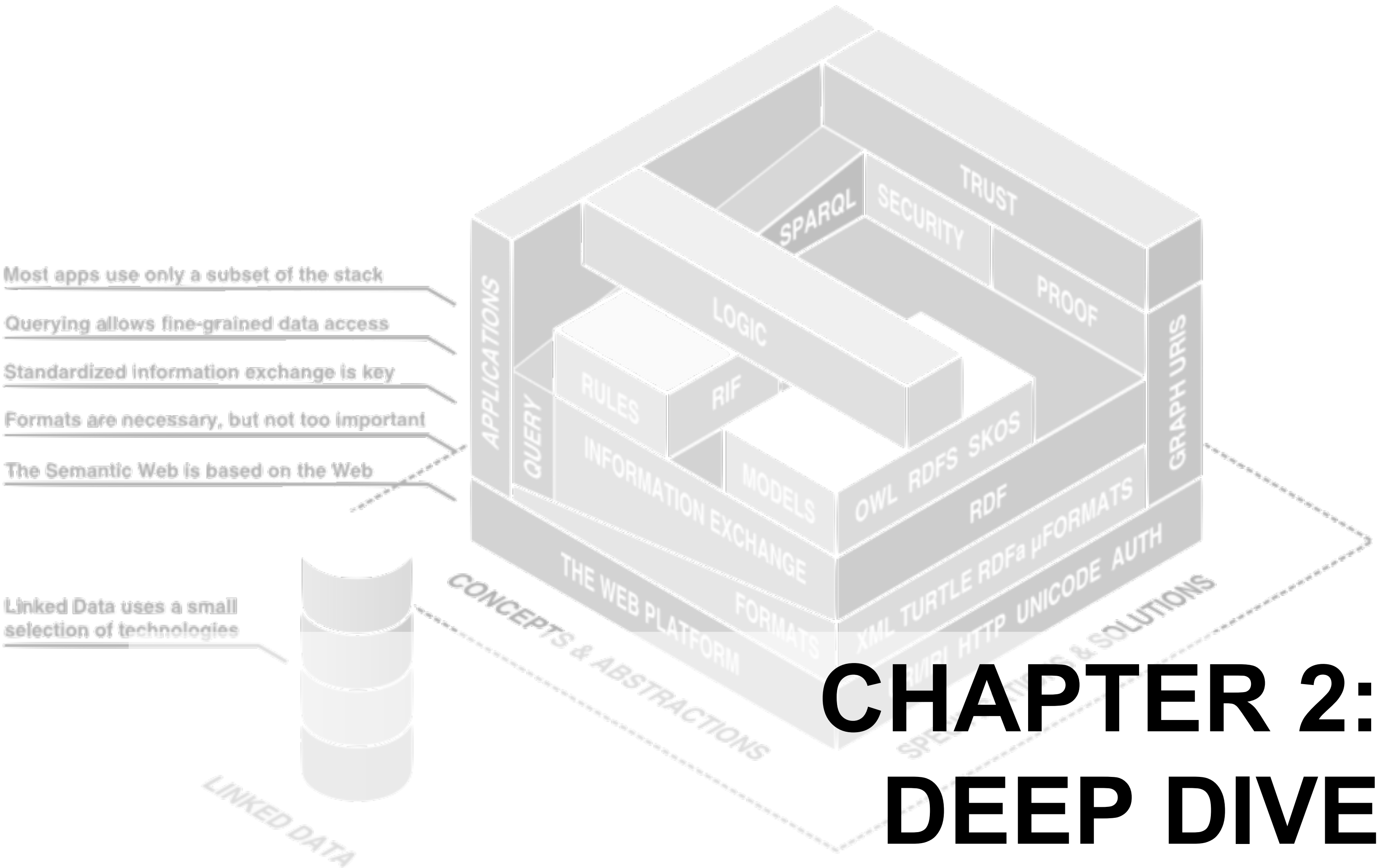
- Linda Project: Linked Data Primer

Articles:

- C. Bizer, T. Heath, and T. Berners-Lee. **Linked Data - The Story So Far**. International Journal on Semantic Web and Information Systems, 5(3):1 – 22, 2009.

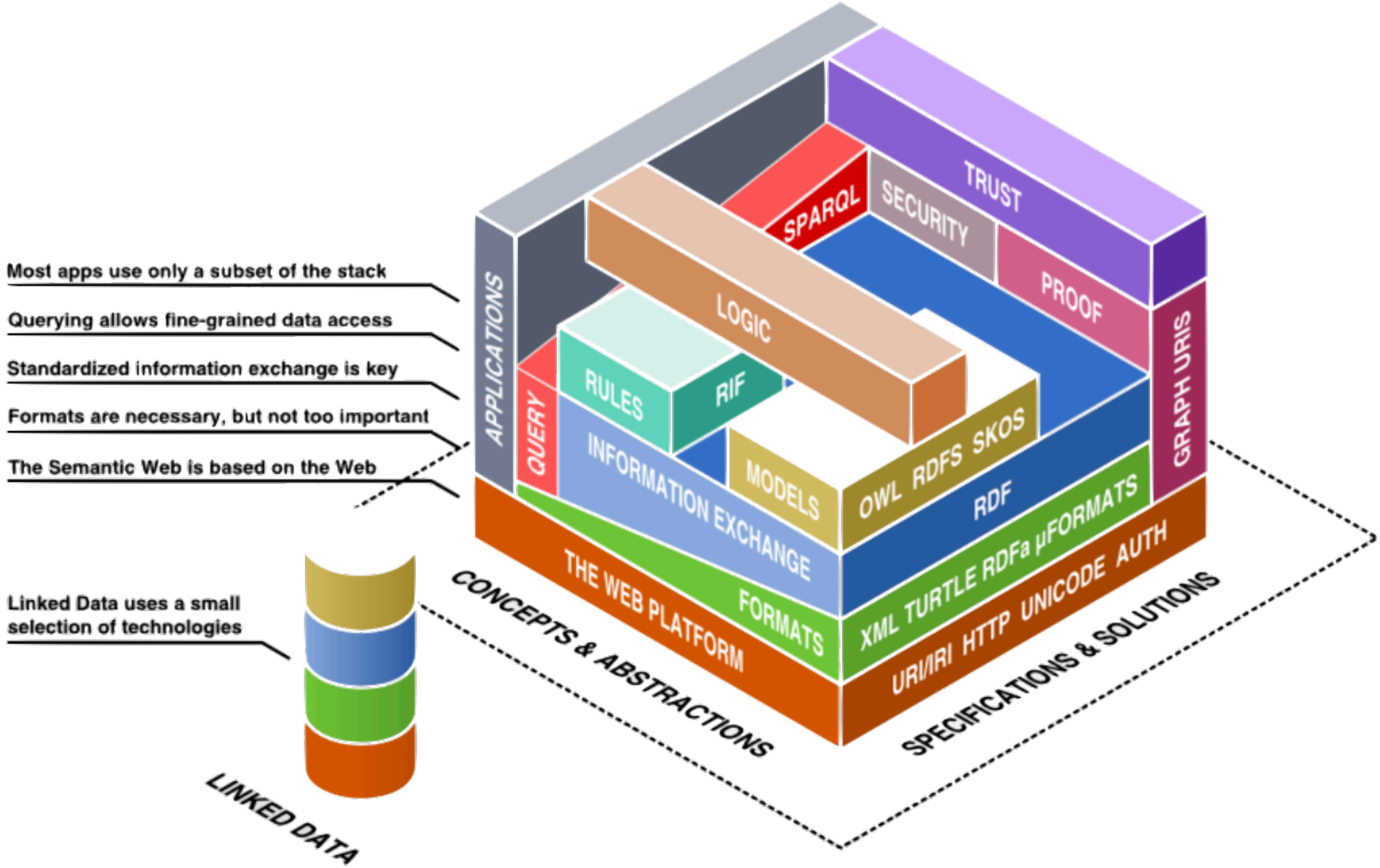
Books:

- T. Pellegrini, H. Sack, and S. Auer, Eds., *Linked Enterprise Data*. Heidelberg: Springer Berlin, 2014.
- Tom Heath, Christian Bizer (2011). *Linked Data - Evolving the Web into a Global Data Space*. Morgan & Claypool, 2011.
- EUCLID Project Consortium (2014). *Using Linked Data Effectively*.
- Hitzler, Rudolph, Krötzsch (2009). *Foundations of Semantic Web Technologies*. Chapman & Hall/CRC



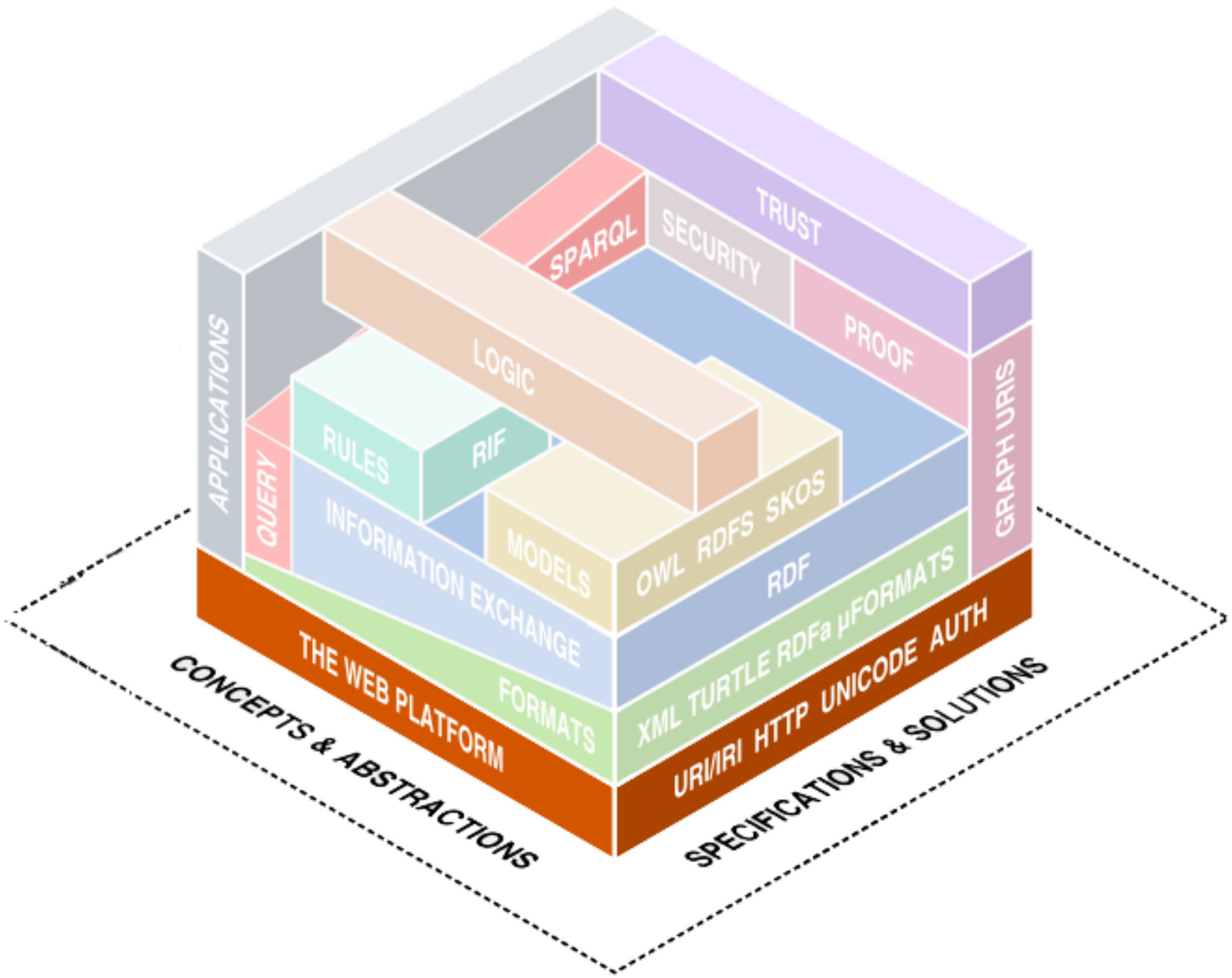
CHAPTER 2: DEEP DIVE

The Semantic Web Technology Stack



http://bnode.org/blog/2009/07/08/the-semantic-web-not-a-piece-of-cake

Web Technology Foundations



HTTP Content negotiation

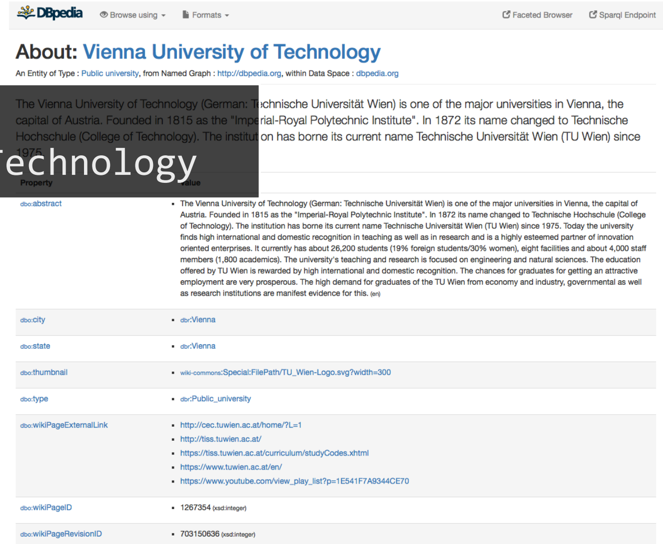
Provides different representations of a resource at the same URI.

Speciefied in RFC 7231 HTTP 1.1

HTTP Content negotiation

Provides different representations of a resource at the same URI.

```
curl -L -H "Accept: text/html"
http://dbpedia.org/resource/Vienna_University_of_Technology
```

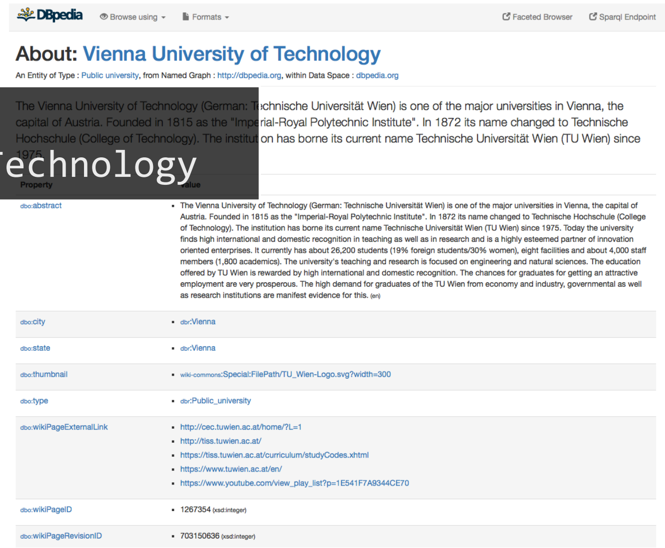


Speciefied in RFC 7231 HTTP 1.1

HTTP Content negotiation

Provides different representations of a resource at the same URI.

```
curl -L -H "Accept: text/html"
http://dbpedia.org/resource/Vienna_University_of_Technology
```




```
curl -L -H "Accept: application/rdf+xml"
http://dbpedia.org/resource/Vienna_University_of_Technology
```

Speecified in RFC 7231 HTTP 1.1

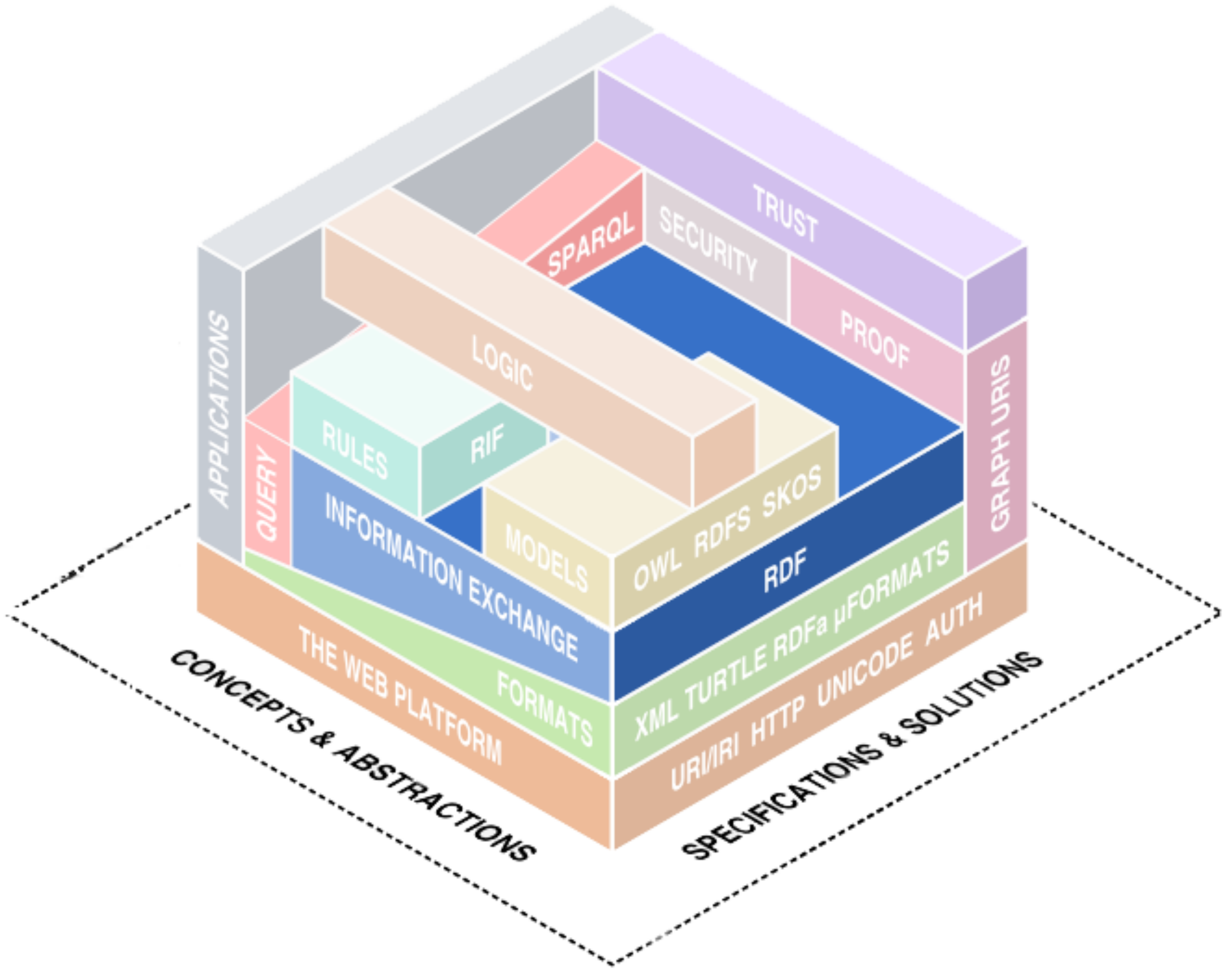


Image: http://bnode.org/blog/2009/07/08/the-semantic-web-not-a-piece-of-cake

RDF (Resource Description Framework)

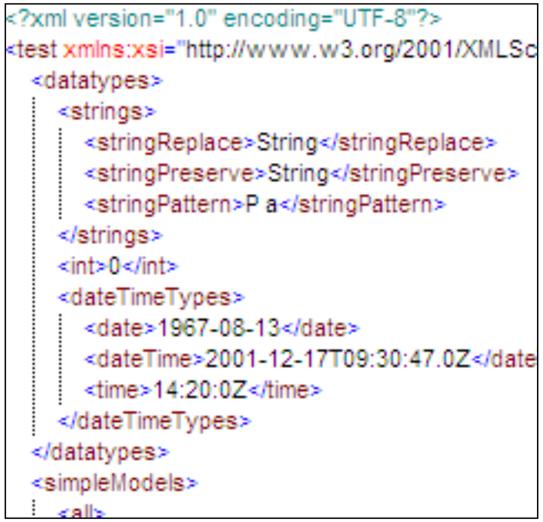
- Graph-based model
- Formal semantics → machine-readable
- Knowledge expressed as a list of statements
- All statements follow the same simple schema (RDF Triple)
- Resource can be anything, must be uniquely identified and referenceable via URI

- Developed in the late 1990s
- Final W3C recommendation in 2004 (RDF 1.1)
- Abstract model with various syntax notations and serialization formats:
  - Graph notation
  - RDF/XML (historically first serialization format)
  - N3/Turtle
  - N-Triples
  - JSON-LD

XML vs RDF

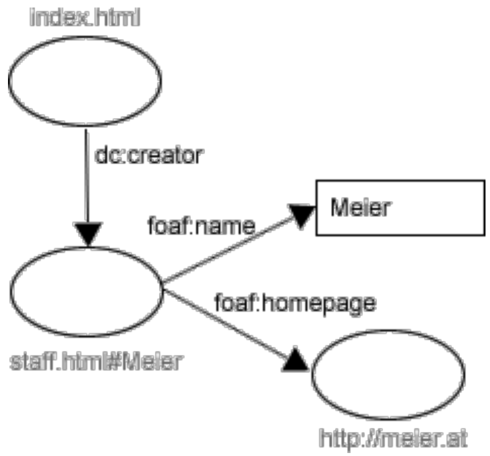
Inherently different data models

- XML is hierarchical
  - One root element + nested sub elements
  - Organizes data in a structured way



```
<?xml version="1.0" encoding="UTF-8"?>
<test xmlns:xsi="http://www.w3.org/2001/XMLSc
  <datatypes>
    <strings>
      <stringReplace>String</stringReplace>
      <stringPreserve>String</stringPreserve>
      <stringPattern>P a</stringPattern>
    </strings>
    <int>0</int>
    <dateTimeTypes>
      <date>1967-08-13</date>
      <dateTime>2001-12-17T09:30:47.0Z</date
      <time>14:20:0Z</time>
    </dateTimeTypes>
  </datatypes>
  <simpleModels>
    <all>
```

- RDF is based on associations between resources
  - More flexible with the use of links (URIs)
  - More difficult for people to read
    - The structure cannot be grasped easily
    - Alternative representation as RDF-Graph



RDF Triples

Basic RDF building block

- **Subject:** a resource, which may be identified with a URI
- **Predicate:** a URI-identified reused specification of the relationship
- **Object:** a resource or literal to which the subject is related

Set of RDF assertions in triple-form form a labelled, directed graph:

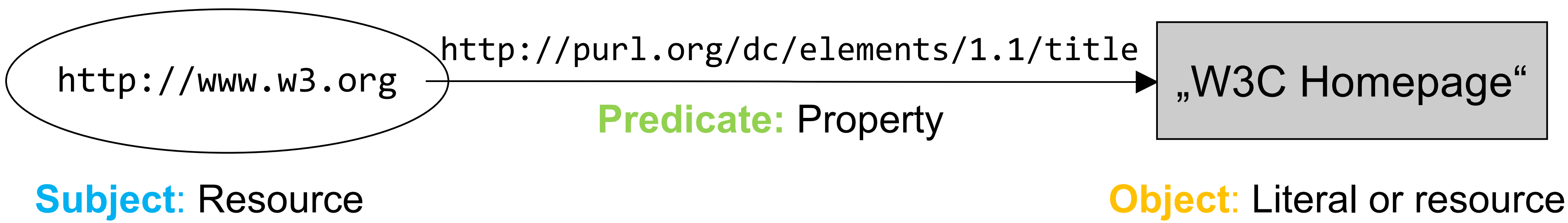
- **Resources:** the subjects and objects are nodes of the graph
- **Predicates:** each predicate use becomes a label for an arc connecting the subject to the object.

RDF Triples

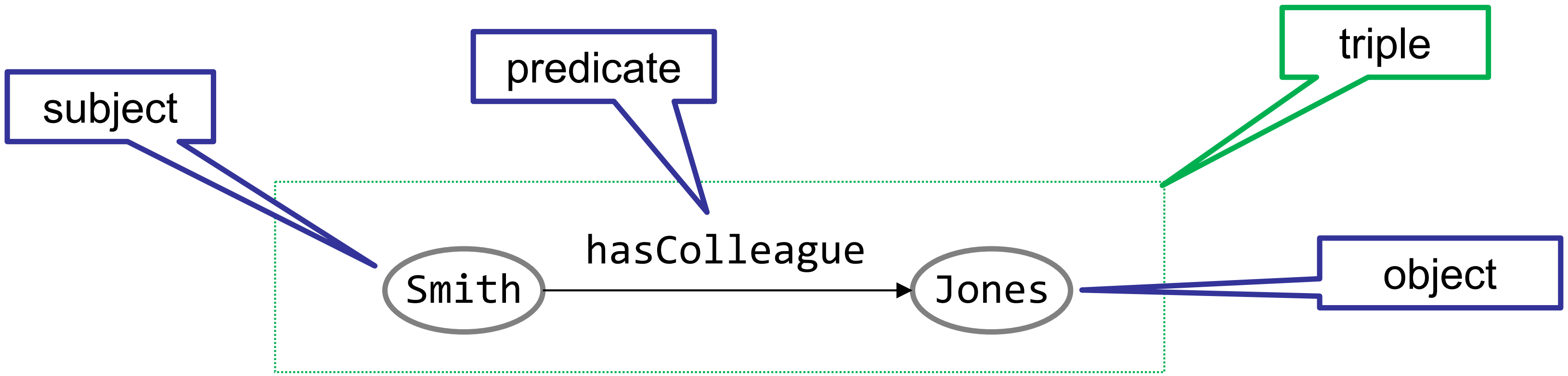
Basic RDF building block
- **Subject:** a resource, which may be identified with a URI
- **Predicate:** a URI-identified reused specification of the relationship
- **Object:** a resource or literal to which the subject is related

Set of RDF assertions in triple-form form a labelled, directed graph:

- **Resources:** the subjects and objects are nodes of the graph
- **Predicates:** each predicate use becomes a label for an arc connecting the subject to the object.

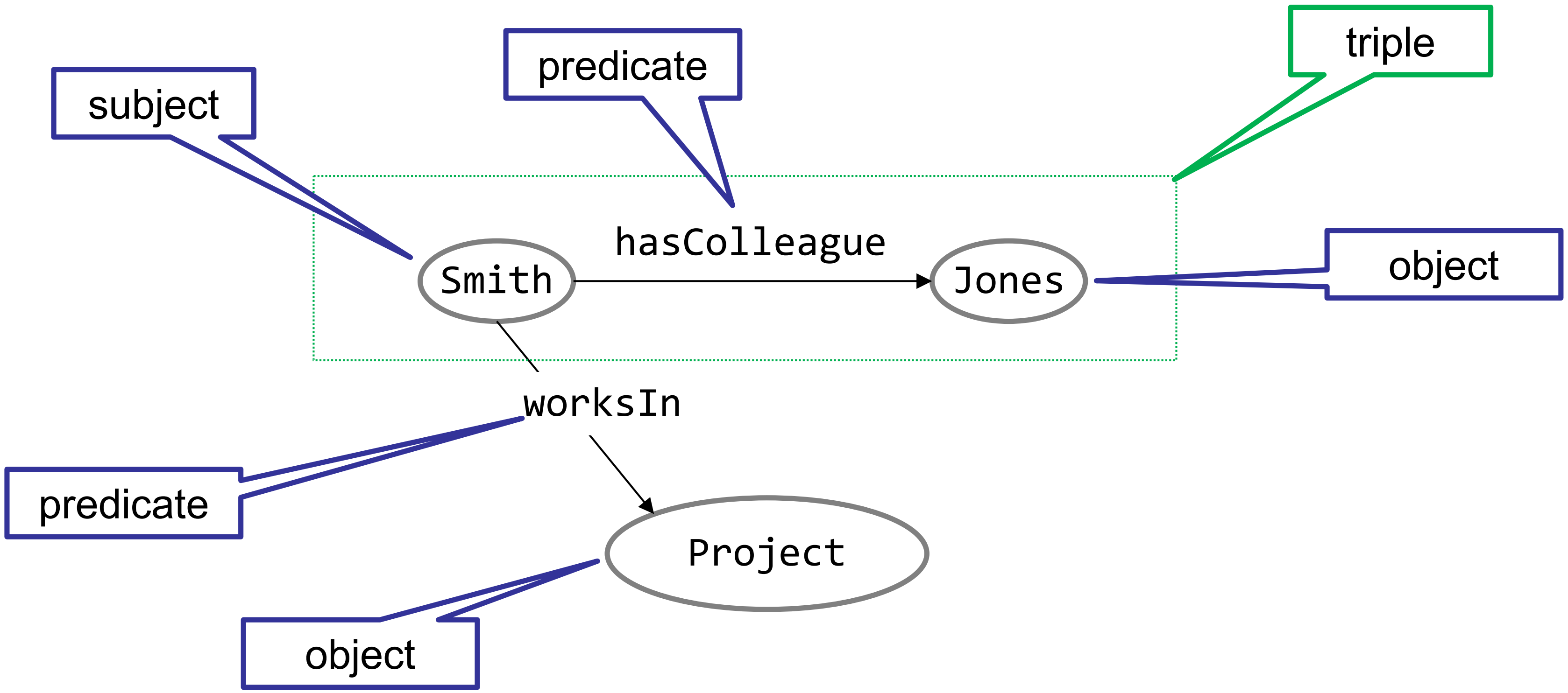


- Object of one statement may be the subject of another statement
- The result is a directed labelled (multi-)graph
- The object of a triple is a resource or a literal



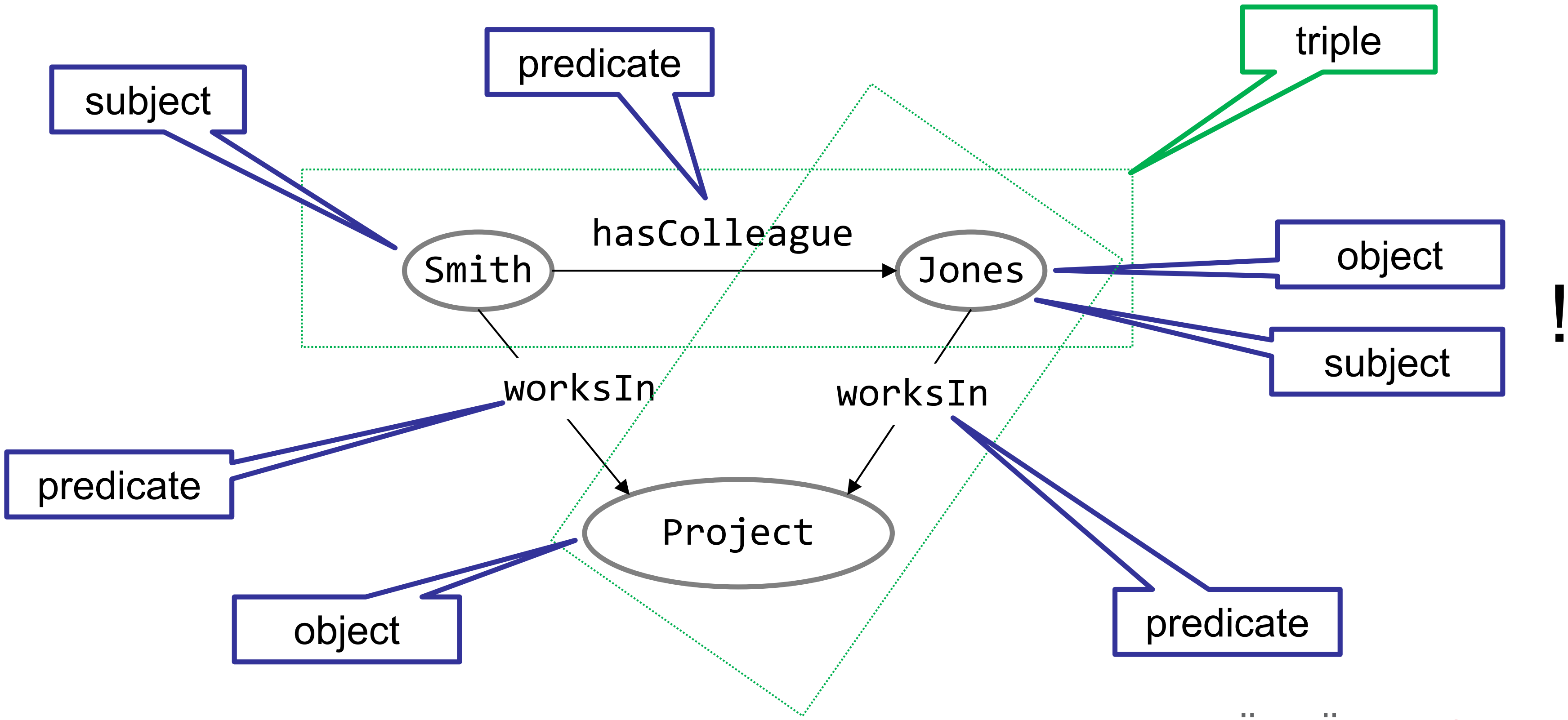
RDF Graphs

- Object of one statement may be the subject of another statement
- The result is a directed labelled (multi-)graph
- The object of a triple is a resource or a literal



RDF Graphs

- Object of one statement may be the subject of another statement
- The result is a directed labelled (multi-)graph
- The object of a triple is a resource or a literal



Plain literals have a **lexical form** (their lexical value) and optionally a **language tag**.

```
“27” ,  “Hello world”@en
```

Typed Literals

- Formed by pairing a string with a URIref that identifies a particular datatype
- Indicates what datatype should be used to interpret a given literal
- RDF has no built-in set of datatypes of its own (except `rdf:XMLLiteral`)
- RDF datatype concepts are based on a conceptual framework from **XML Schema datatypes** that defines
  - the value space,
  - the lexical space and
  - the lexical-to-value mapping for a datatype (see RDF specifications)

```
"27"^^http://www.w3.org/2001/XMLSchema#integer
```

XML Schema datatypes

	Datatype
Core types	xsd:string
	xsd:boolean
	xsd:decimal
	xsd:integer
IEEE floating-point numbers	xsd:double
	xsd:float
Time and date	xsd:date
	xsd:time
	xsd:dateTime
	xsd:dateTimeStamp
Recurring and partial dates	xsd:gYear
	xsd:gMonth
	xsd:gDay
	xsd:gYearMonth
	xsd:gMonthDay
	xsd:duration
	xsd:yearMonthDuration
	xsd:dayTimeDuration

Limited-range integer numbers	xsd:byte
	xsd:short
	xsd:int
	xsd:long
	xsd:unsignedByte
	xsd:unsignedShort
	xsd:unsignedInt
	xsd:unsignedLong
	xsd:positiveInteger
	xsd:nonNegativeInteger
	xsd:negativeInteger
	xsd:nonPositiveInteger
Encoded binary data	xsd:hexBinary
	xsd:base64Binary
Miscellaneous XSD types	xsd:anyURI
	xsd:language
	xsd:normalizedString
	xsd:token
	xsd:NMTOKEN
	xsd:Name
	xsd:NCName

https://www.w3.org/TR/2013/WD-rdf11-concepts-20130115/#xsd-datatypes

Excursus: RDF Serialization Formats

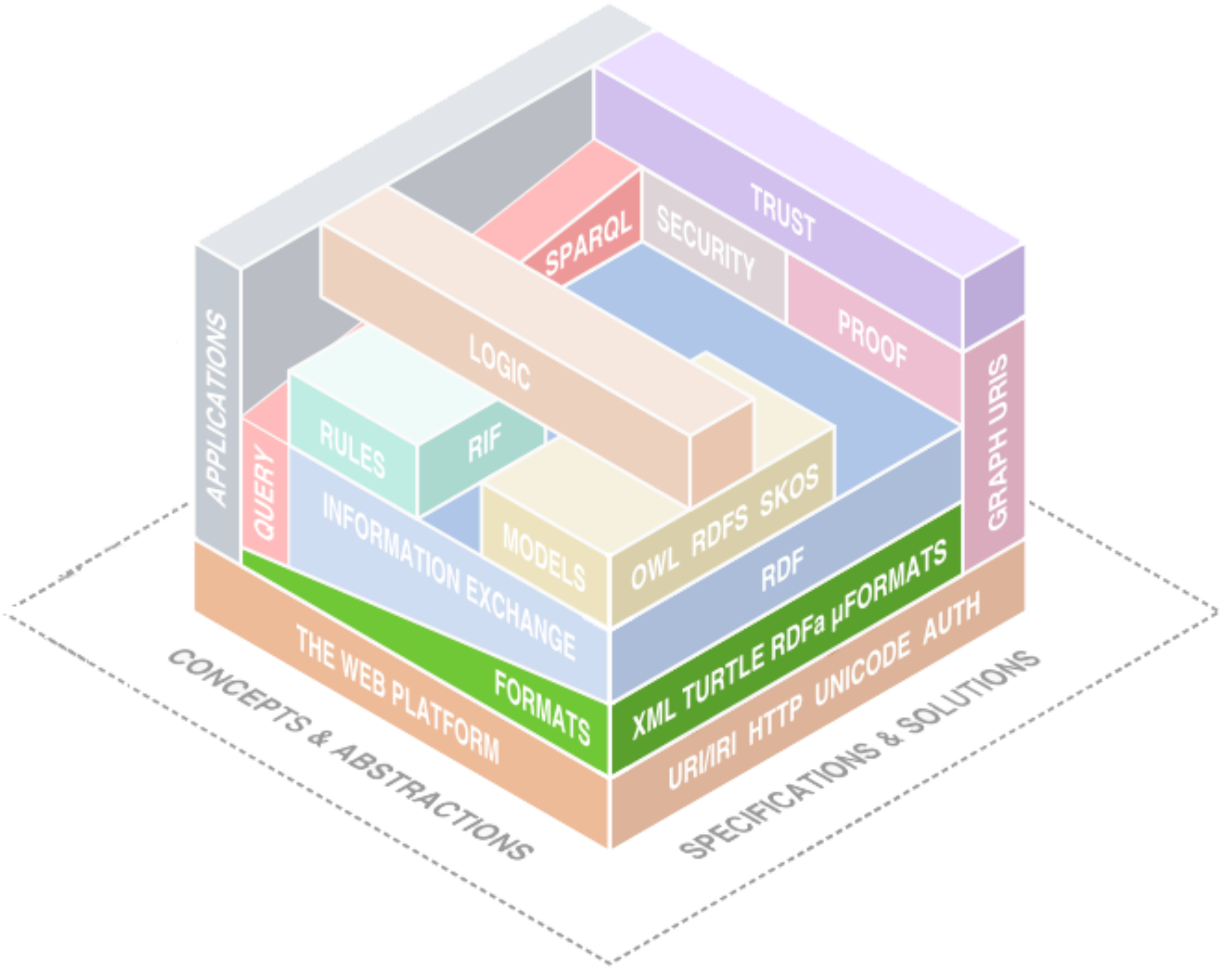
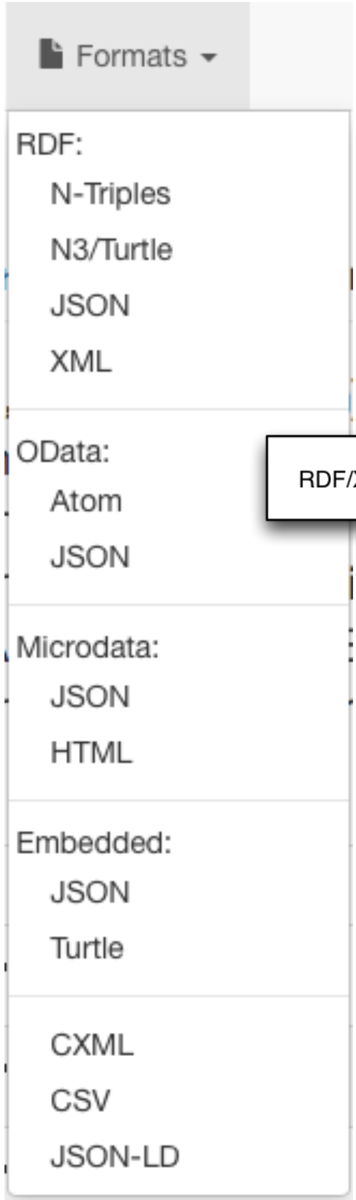
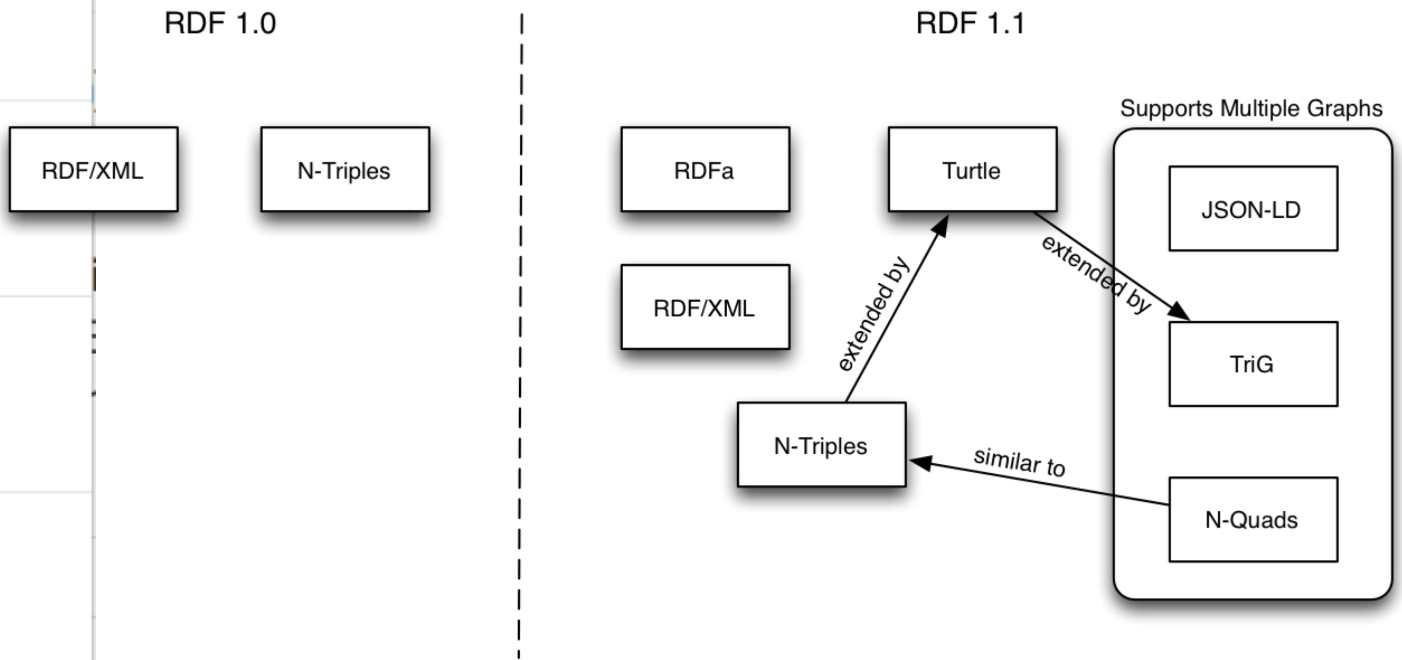


Image: http://bnode.org/blog/2009/07/08/the-semantic-web-not-a-piece-of-cake

RDF Serialization Formats





http://www.w3.org/TR/rdf11-new/

RDF Notations: RDF/XML

- Useful for inter-machine communication
- Primary (recurring) element in encoding assertions (i.e., triples) is `rdf:Description`, e.g.:

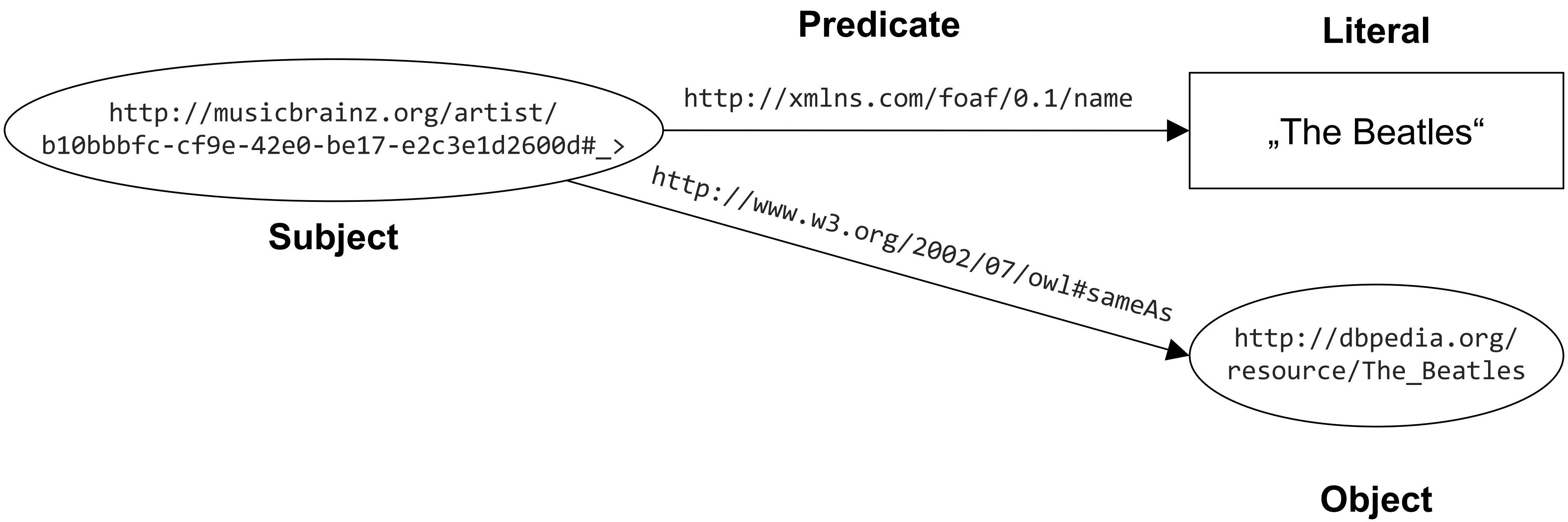
```
<rdf:Description rdf:about="http://musicbrainz.org/artist/b10bbbfc-cf9e-42e0-be17-e2c3e1d2600d#_">
  <foaf:name>The Beatles</foaf:name>
  <owl:sameAs rdf:resource="http://dbpedia.org/resource/The_Beatles">
</rdf:Description>
```

RDF/XML

```
<rdf:Description rdf:about="http://musicbrainz.org/artist/4d5447d7-c61c-4120-ba1b-d7f471d385b9#">
  <foaf:name>John Lennon</foaf:name>
</rdf:Description>
```

RDF/XML

RDF Notations: N-Triples




```
<http://musicbrainz.org/artist/b10bbbfc-cf9e-42e0-be17-e2c3e1d2600d#_>
      <http://xmlns.com/foaf/0.1/name>
      "The Beatles”.

<http://musicbrainz.org/artist/b10bbbfc-cf9e-42e0-be17-e2c3e1d2600d#_>
      <http://www.w3.org/2002/07/owl#sameAs>
      <http://dbpedia.org/resource/The_Beatles>.
```

N-Triples

RDF Notations: N3

- Easy to read and write
- Superset of Turtle
- <subject> <predicate> <object>
- Every subject, predicate, object is identified with a URI
- Two shortcuts for several statements about the same subject
  - ";" introduce another predicate of the same subject
  - "," introduce another object with the same predicate and subject

```
<#pat> <#child> <#al>, <#chaz>, <#mo>;
       <#age>    24;
       <#eyecolor>      "blue".
                                               N3
```

Interpretation:
Pat has the children al, chaz, mo; Pat's age is 24; Pat's eyecolor is blue

- "Terse RDF Triple Language"
- Many URIs share the same basis
  → use prefixes for namespace declarations:

```
@prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>.
@prefix owl:<http://www.w3.org/2002/07/owl#>.
@prefix mo:<http://purl.org/ontology/mo/>.
@prefix dbpedia:<http://dbpedia.org/resouce/>.
```
Turtle

Note: You can find commonly used prefixes here: http://prefix.cc

- Unique base:

```
@base <http://musicbrainz.org/>.
```
Turtle

Has a simple shorthand for class membership:

```
@base <http://musicbrainz.org/>.
@prefix mo:<http://purl.org/ontology/mo/>.
<artist/b10bbbfc-cf9e-42e0-be17-e2c3e1d2600d#_>
               a mo:MusicGroup.
```

Turtle

Has a simple shorthand for class membership:

```
@base <http://musicbrainz.org/>.
@prefix mo:<http://purl.org/ontology/mo/>.
<artist/b10bbbfc-cf9e-42e0-be17-e2c3e1d2600d#_>
               a mo:MusicGroup.
```

Turtle

is equivalent to

```
<http://musicbrainz.org/artist/b10bbbfc-cf9e-42e0-be17-
e2c3e1d2600d#_>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
  <http://purl.org/ontology/mo/MusicGroup>.
```

Turtle

When multiple statements apply to same subject they can be abbreviated as follows:

```
<artist/b10bbbfc-cf9e-42e0-be17-e2c3e1d2600d#_>
        rdfs:label "The Beatles";
        owl:sameAs dbpedia:The_Beatles ,
                <http://www.bbc.co.uk/music/artists/
                b10bbbfc-cf9e-42e0-be17-e2c3e1d2600d#artist> .
```

Turtle

When multiple statements apply to same subject they can be abbreviated as follows:

```
<artist/b10bbbfc-cf9e-42e0-be17-e2c3e1d2600d#_>
      rdfs:label "The Beatles" ;
      owl:sameAs dbpedia:The_Beatles ,
              <http://www.bbc.co.uk/music/artists/
              b10bbbfc-cf9e-42e0-be17-e2c3e1d2600d#artist> .
```

Turtle

Same subject

When multiple statements apply to same subject they can be abbreviated as follows:

```
<artist/b10bbbfc-cf9e-42e0-be17-e2c3e1d2600d#_>
       rdfs:label "The Beatles";
       owl:sameAs dbpedia:The_Beatles,
               <http://www.bbc.co.uk/music/artists/
               b10bbbfc-cf9e-42e0-be17-e2c3e1d2600d#artist> .
```

Turtle

Same subject

Same subject and object

Turtle also provides a simple syntax for data types and language tags for literals, respectively:

```
<recording/5098d0a8-d3c3-424e-9367-1f2610724410#_>
       a mo:Signal;
       rdfs:label "All You Need Is Love" ;
       mo:duration "PT3M48S"^^xsd:duration .

dbpedia:The_Beatles dbpedia-owl:abstract
       "The Beatles were an English rock band formed (…) "@en,
       "The Beatles waren eine britische Rockband in den(…) "@de .
```

Turtle

RDF Notations: JSON-LD

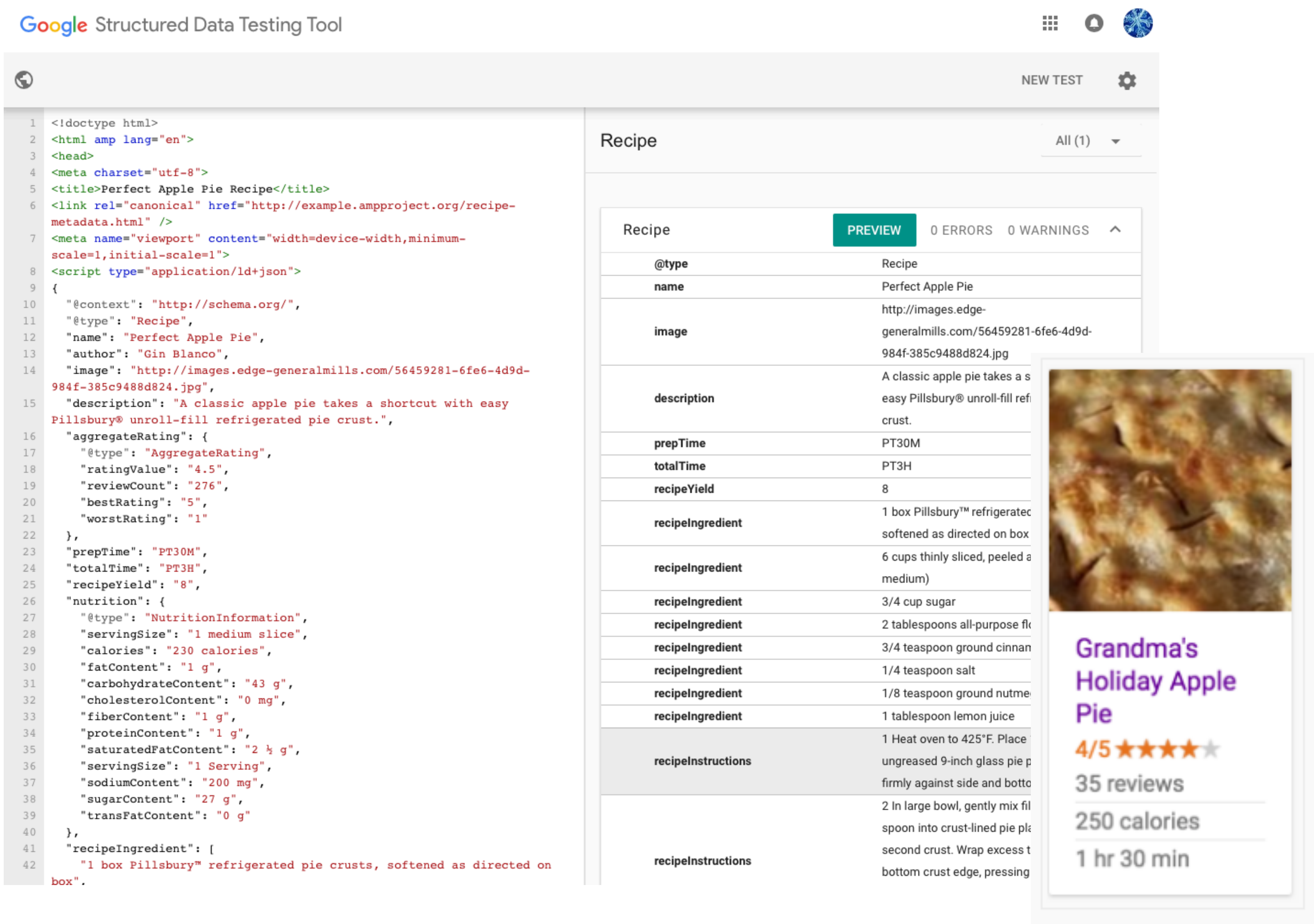
- **J**ava**S**cript **O**bject **N**otation for **L**inked **D**ata
- W3C recommendation
- Imposes structure on JSON with @context annotations (often external vocabularies)

```
{"@context": {
    "name": "http://xmlns.com/foaf/0.1/name",
    "homepage": {
      "@id": "http://xmlns.com/foaf/0.1/workplaceHomepage",
      "@type": "@id"
    },
    "Person": "http://xmlns.com/foaf/0.1/Person"
  },
  "@id": "http://me.example.com",
  "@type": "Person",
  "name": "John Smith",
  "homepage": "http://www.example.com/"
}
```

JSON-LD

Specification: http://json-ld.org/spec/latest/json-ld/
Example from https://en.wikipedia.org/wiki/JSON-LD

JSON-LD in the wild: Google/Schema.org



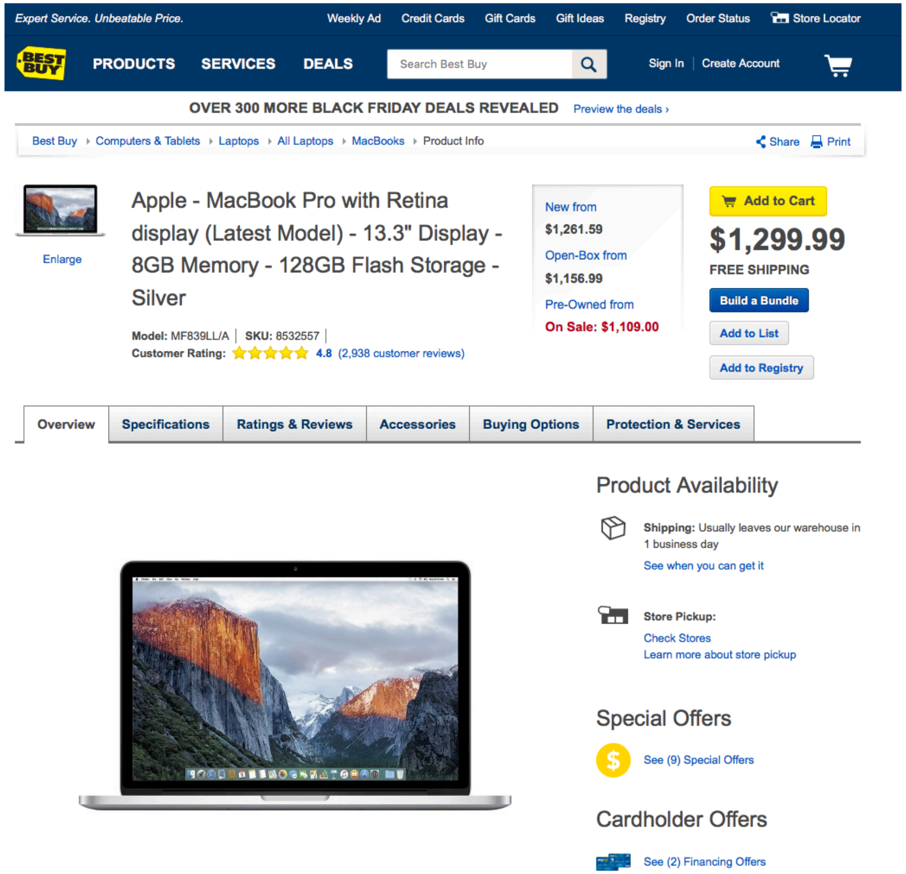
Embedding RDF on the Web

1. Direct RDF embedding
2. Microformats
3. RDFa (lite)
4. HTML5 Microdata (schema.org)

Adoption of Semantic Web Annotations

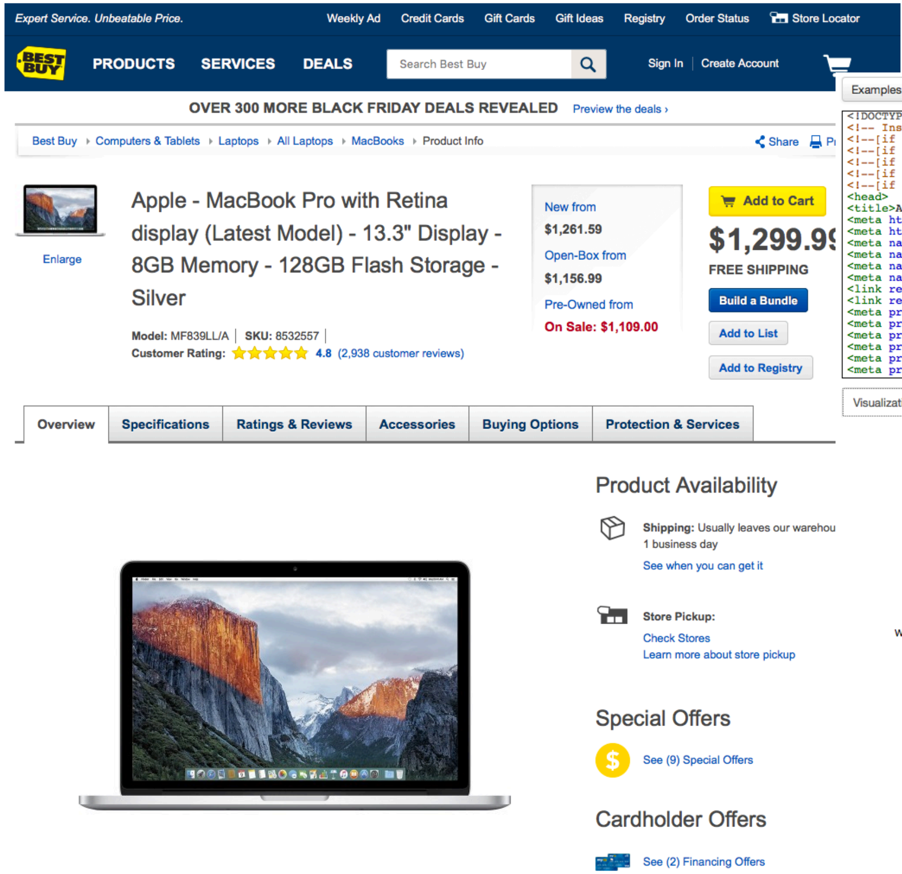
- Search engines: Google (rich snippets), Yahoo, Bing
- Social networks: Facebook (Open Graph protocol), LinkedIn
- Retailers: BestBuy, Tesco,..
- Media and publishers: O'Reilly, Newsweek, BBC..
- Slideshare, Digg
- Whitehouse.gov, Library of Congress, UK government
- etc.

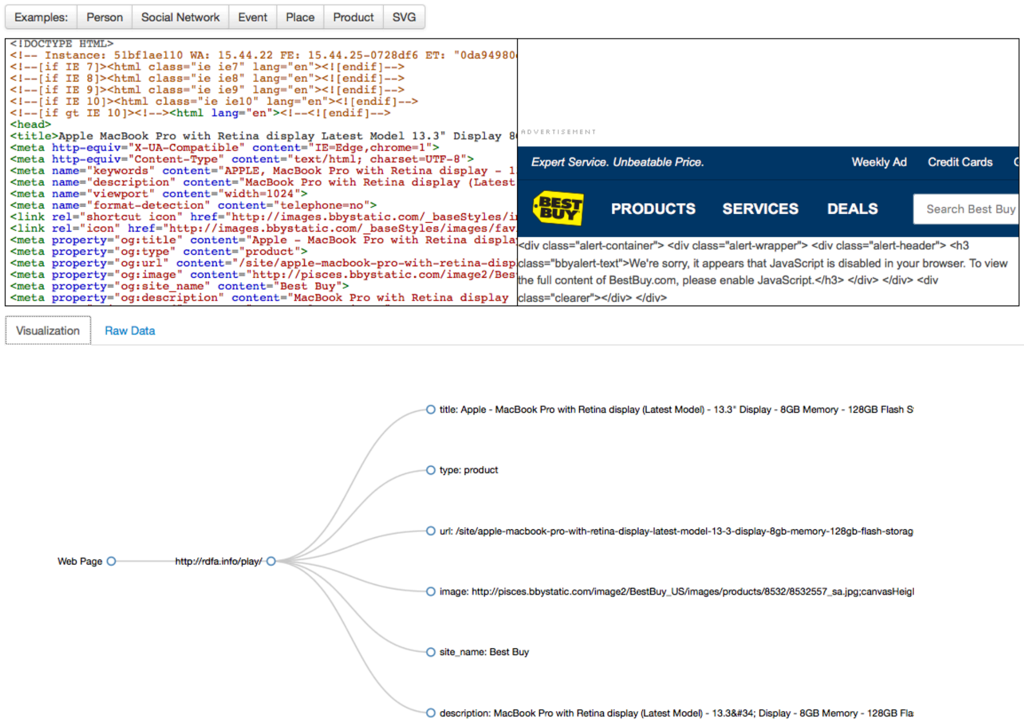
RDFa example: BestBuy



http://rdfa.info/play/

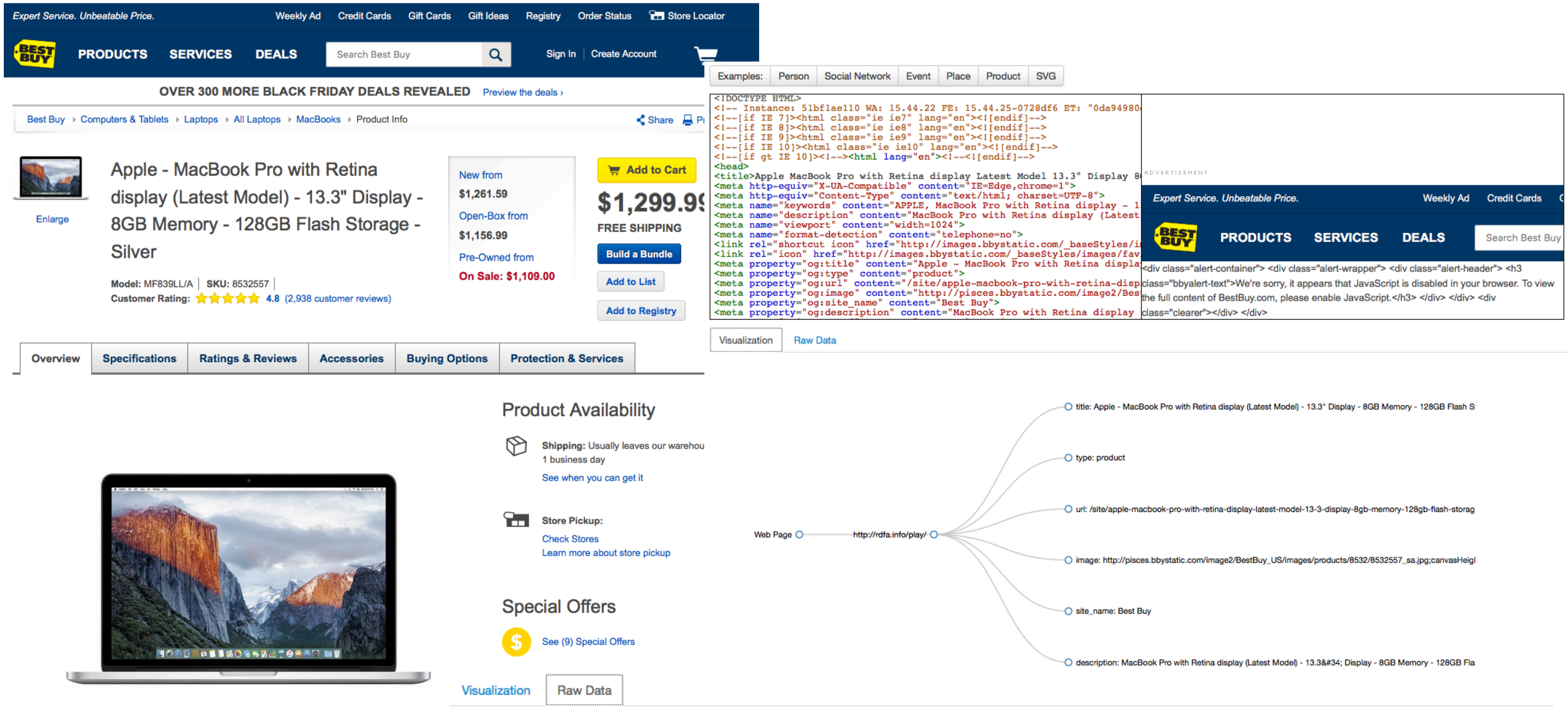
RDFa example: BestBuy





http://rdfa.info/play/

RDFa example: BestBuy

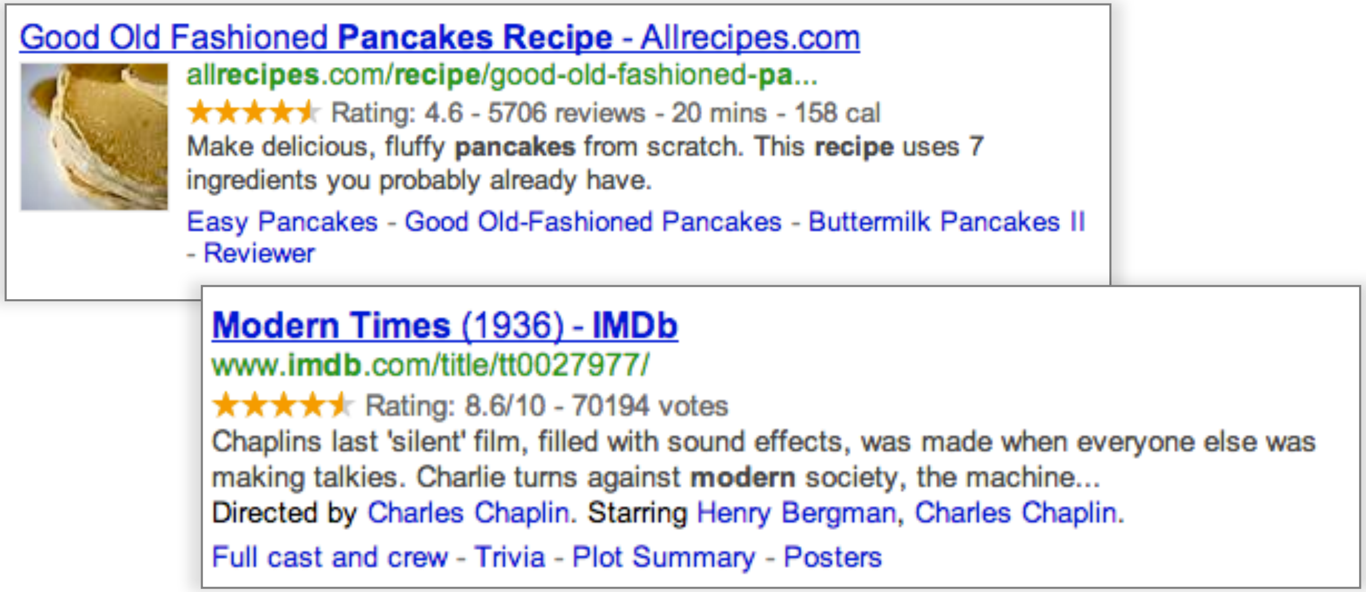



```
@prefix og: <http://ogp.me/ns#> .

<http://rdfa.info/play/>
   og:title "Apple - MacBook Pro with Retina display (Latest Model) - 13.3\" Display - 8GB Memory - 128GB Flash Storage - Silver"@en;
   og:type "product"@en;
   og:url "/site/apple-macbook-pro-with-retina-display-latest-model-13-3-display-8gb-memory-128gb-flash-storage-silver/8532557.p?id=1219343246941&s
   og:image "http://pisces.bbystatic.com/image2/BestBuy_US/images/products/8532/8532557_sa.jpg;canvasHeight=210;canvasWidth=210"@en;
   og:site_name "Best Buy"@en;
   og:description "MacBook Pro with Retina display (Latest Model) - 13.3&#34; Display - 8GB Memory - 128GB Flash Storage,"@en .
```

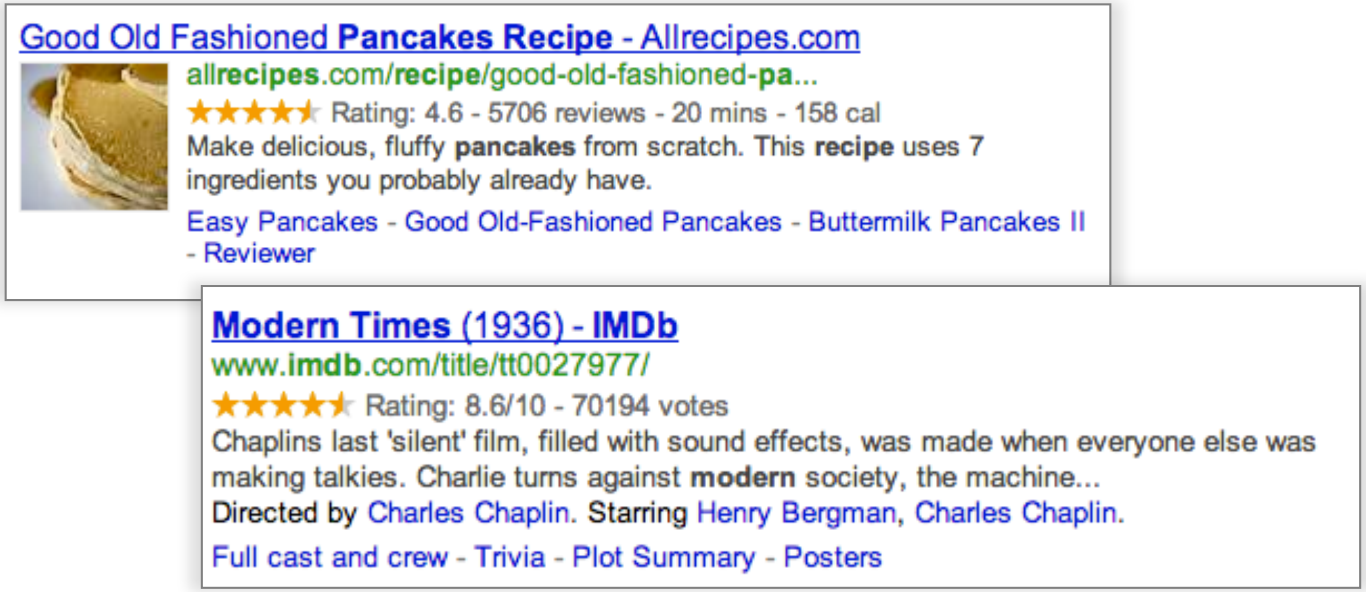
http://rdfa.info/play/

RDFa Example: Google Rich Snippets

RDFa Example: Google Rich Snippets



RDFa Example: Google Rich Snippets



Enhanced search results created from RDFa Lite or Microdata

Currently supported types:
`Product, Recipe, Review, Event, SoftwareApplication`

RDF-S

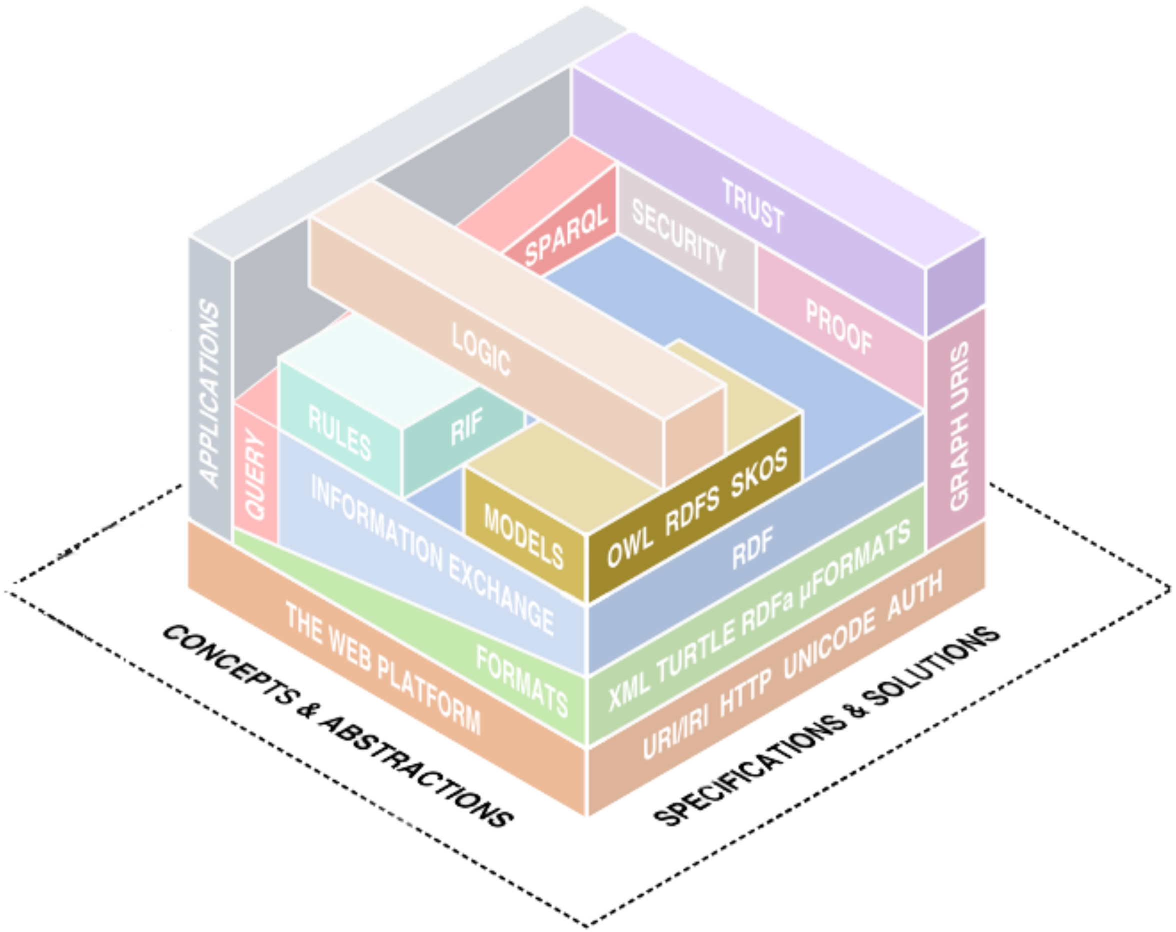


Image: http://bnode.org/blog/2009/07/08/the-semantic-web-not-a-piece-of-cake

RDF-S (RDF Schema)

- Provides primitive lightweight schemas for RDF triples:
  - Vocabulary (class hierarchies, properties)
  - Elementary inferences (multiple inheritance for classes and properties)

- First version in April 1998, W3C recommendation in Feb. 2004
- Every RDF Schema document is an RDF document
- Facilitates entailment regimes
- RDF-S vs. OWL:
  - RDF-S is less expressive than OWL
  - Many RDF-S components included in OWL

1. **Expectation:**
   Nominate
   - the 'types' (i.e., classes) of things we might make assertions about, and
   - the properties we might apply, as predicates in these assertions, to capture their relationships

2. **Inference:**
   Given a set of assertions, using these classes and properties, specify what should be inferred about assertions that are implicitly made
   - Type and property propagation
   - subClass and subProp transitivity
   - Domain and range inference

RDF-S Example

Schema

We *expect* to use this vocabulary to make assertions about music groups...

```
mo:MusicGroup rdfs:subClassOf foaf:Group .
```

Turtle

Existing fact

Having made such an assertation...

```
<artist/b10bbbfc-cf9e-42e0-be17-e2c3e1d2600d#_>
       rdf:type
       mo:MusicGroup .
```

Turtle

RDF-S Example

Schema

We *expect* to use this vocabulary to make assertions about music groups...

```
mo:MusicGroup rdfs:subClassOf foaf:Group .
```
Turtle

Existing fact

Having made such an assertation...

```
<artist/b10bbbfc-cf9e-42e0-be17-e2c3e1d2600d#_>
       rdf:type
       mo:MusicGroup .
```
Turtle

→ Inferred fact

..Inferences can be drawn that we did not explicitly make.

```
<artist/b10bbbfc-cf9e-42e0-be17-e2c3e1d2600d#_>
       rdf:type
       foaf:Group .
```
Turtle

Signatures: Domain and Range

`rdfs:domain`

- Instance of `rdf:property`
- Restricts the type of values of a property

```turtle
mo:member rdfs:domain mo:MusicGroup .
```
Turtle

`rdfs:range`

- Instance of `rdf:property`
- Used to restrict the type of values of a property

```turtle
mo:member rdfs:range foaf:Agent .
```
Turtle

RDF and RDF-S Vocabularies

RDF Classes:
`rdf:Property`, `rdf:Statement`, `rdf:XMLLiteral`, `rdf:Seq`, `rdf:Bag`, `rdf:Alt`, `rdf:List`

RDF Properties:
`rdf:type`, `rdf:subject`, `rdf:predicate`, `rdf:object`, `rdf:first`, `rdf:rest`, `rdf:_n`, `rdf:value`

RDF Resources:
`rdf:nil`

RDFS Classes
- `rdfs:Resource`
- `rdfs:Class`
- `rdfs:Literal`
- `rdfs:Datatype`
- `rdfs:Container`
- `rdfs:ContainerMembershipProperty`

RDFS Properties
- `rdfs:domain`
- `rdfs:range`
- `rdfs:subPropertyOf`
- `rdfs:subClassOf`
- `rdfs:member`
- `rdfs:seeAlso`
- `rdfs:isDefinedBy`
- `rdfs:comment`
- `rdfs:label`

http://www.w3.org/TR/2014/NOTE-rdf11-primer-20140624/
http://www.w3.org/TR/rdf-schema/

Looks like object-oriented programming, but…

- Properties are first-class citizens
- Multi-instantiation:
  - resource can have several types
  - can be the instance of several classes
  - like multiple perspectives on/facets of a resource

- OO: design decisions based on *operational* properties of a class
- RDF-S/OWL: decisions based on *structural* properties of a class
  - → RDF-S/OWL are property-focused; OO is class-focused
  - → a class structure and relations among classes in an ontology are different from the structure for a similar domain in an OO program
  - → Impedance mismatch: one-to-one class mapping between Objects and RDF-S/OWL classes is problematic (e.g. multiple inheritance)

Main RDF-S constructs

Classes:

- **`rdfs:Resource`** all resources within RDF are implicitly members of this class
- **`rdfs:Class`** declares a resource as a class (type or category) for other resources; instantiation of classes with `rdf:Type`
- **`rdfs:Literal`** literal values such as strings and integers (plain or typed)
- **`rdfs:Datatype`** class of datatypes (both an instance of and a subclass of `rdfs:Class`; each instance of `rdfs:Datatype` is a subclass of `rdfs:Literal`)

Properties:

- **`rdfs:subClassOf`** allows to declare hierarchies of classes; properties of super class are inherited
- **`rdfs:subPropertyOf`** all resources related by one property are also related by another
- **`rdfs:domain`** defines the scope of a property (see slide)
- **`rdfs:range`** defines the range of values of a property (see slide)

- **`rdfs:comment`** used to provide a human-readable description of a resource
- **`rdfs:seeAlso`** used to indicate a resource that might provide additional information
- **`rdfs:isDefinedBy`** indicates a resource defining the subject source

(no inference)

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

RDF(S) does not allow for

- **Localized range and domain constraints**

  e.g., can't express that the range of `publishedBy` is `Publisher` when applied to `Journal` and `Institution` when applied to `TechnicalReport`

- **Existence and cardinality constraints**

  e.g., can't express that all instances of `Paper` have an author that is also a `Person`, or that papers must have at least 3 reviewers

- **Transitive, inverse or symmetrical properties**

  e.g., can't express that `isSubEventOf` is a transitive property, or that `hasRole` is the inverse of `isRoleAt`

OWL

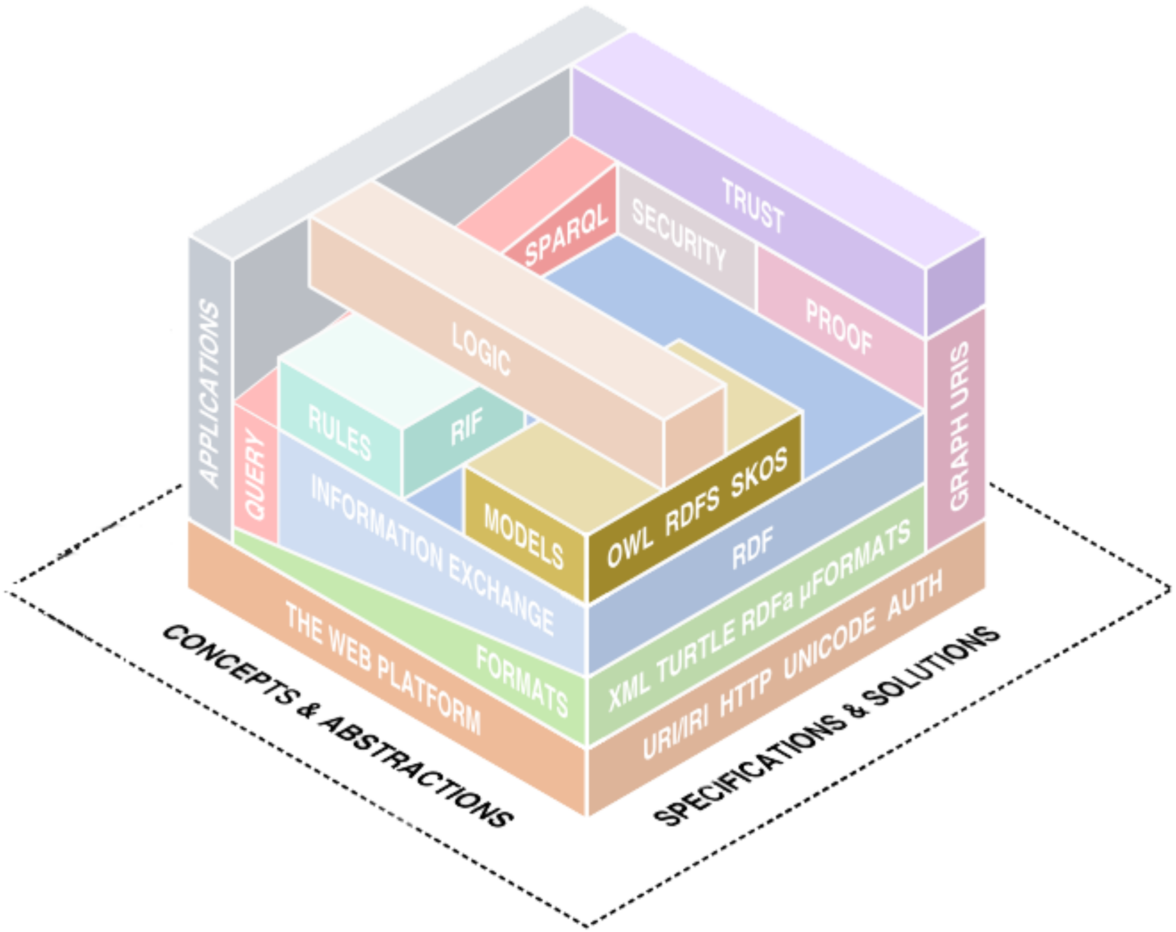


Image: http://bnode.org/blog/2009/07/08/the-semantic-web-not-a-piece-of-cake

Web Ontology Language (OWL)

- W3C Recommendation for the representation of Ontologies
- Full-fledged knowledge representation language for the web
- Provides wider range of ontological constructs and avoids some of the potential confusion in RDF-S
- Formal foundations: Description Logics (DL)
  - well-defined semantics
  - well understood properties (e.g., computational complexity)
  - allows to verify consistency of defined knowledge
  - inferencing allows to make implicit knowledge explicit
  - many tools available
- OWL models can be exchanged as RDF documents

OWL is made up of **terms** which provide for:

- **Class construction:** forming new classes from membership of existing ones (e.g., `unionOf`, `intersectionOf`, etc.).
- **Property construction:** distinction between
  - `ObjectProperties` (resources as values) and
  - `DatatypeProperties` (literals as values).
- **Class axioms:**
  sub-class, equivalence and disjointness relationships.
- **Property axioms:** sub-property relationship, equivalence and disjointness, and relationships between properties.
- **Individual axioms:**
  statements about individuals (sameIndividual, differentIndividuals).

Querying with SPARQL

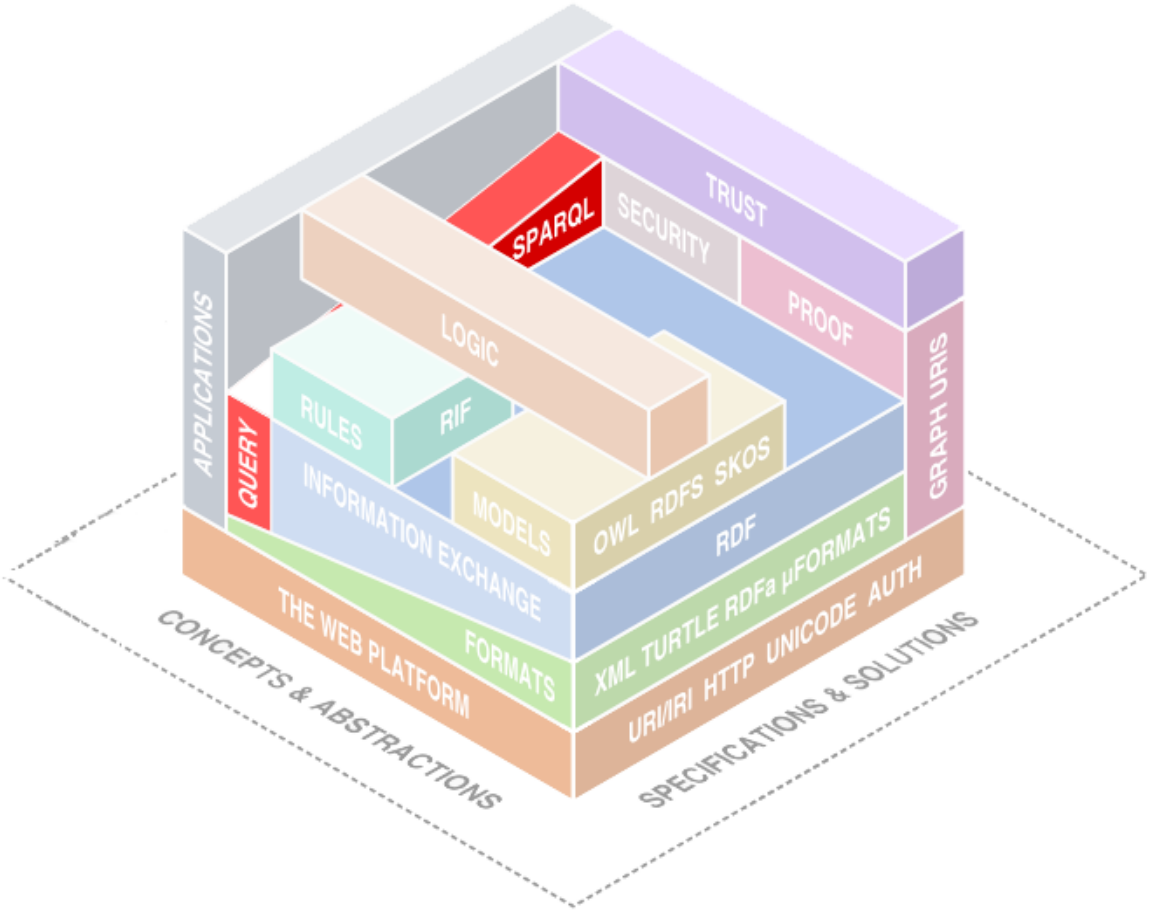


Image: http://bnode.org/blog/2009/07/08/the-semantic-web-not-a-piece-of-cake

- SPARQL = **S**PARQL **P**rotocol **a**nd **R**DF **Q**uery **L**anguage
- Query language designed to use a syntax similar to SQL
- Lets us:
  - Retrieve and manipulate data stored in RDF
  - Explore data by querying unknown relationships
  - Perform complex joins of disparate databases in a single, simple query
  - Transform RDF data from one vocabulary to another

- History:
  - Developed by W3C RDF Data Access Working group (DAWG)
  - January 2008: SPARQL 1.0
  - March 2013: SPARQL 1.1

- Implementations:
  - OpenLink Virtuoso (https://virtuoso.openlinksw.com)
  - Apache Jena ARQ (http://jena.apache.org)
  - Many more: https://www.w3.org/wiki/SparqlImplementations

SPARQL vs. SQL

SQL	SPARQL
Based on relations (tables)	Based on labelled directed graphs
The relations (tables) to be matched over should be indicated	Assumes a default graph (The `FROM` clause populates this with specific identified subgraphs).
(Retrieval) queries produce a relation from a relation	`SELECT` queries produce a relation from a graph. `CONSTRUCT` queries produce a graph from a graph

SPARQL Standards

- SPARQL Query
  - Declarative query language for RDF data
  - http://www.w3.org/TR/rdf-sparql-query/
- SPARQL Algebra
  - Defines the semantics of a SPARQL query execution
  - http://www.w3.org/2001/sw/DataAccess/rq23/rq24-algebra.html
- SPARQL Update
  - Declarative manipulation language for RDF data
  - http://www.w3.org/TR/sparql11-update/
- SPARQL Protocol
  - Standard for communication between SPARQL services and clients
  - http://www.w3.org/TR/sparql11-protocol/

SPARQL Query 1.1

- SPARQL 1.0 only allows data access (querying)
- SPARQL 1.1 introduces:

Query extensions

Aggregates, Sub-queries, Negation, Expressions in the `SELECT` clause, Property paths, assignment, short form for `CONSTRUCT`, expanded set of functions and operators

Updates

Data management:
`Insert, Delete, Delete/Insert`

Graph management:
`Create, Load, Clear, Drop`
`Copy, Move, Add`

Federation extension

Service, values, service variables (informative)

SPARQL Fundamentals

- **RDF graph:**
  Set of RDF assertions, manipulated as a labeled directed graph

- **RDF data set**: set of RDF triples; comprised of
  – one default graph
  – zero or more named graphs

- **SPARQL protocol client:** HTTP client that sends requests for SPARQL Protocol operations (queries or updates)

- **SPARQL protocol service:**
  HTTP server that services requests for SPARQL Protocol operations

- **SPARQL endpoint:** The URI at which a SPARQL Protocol service listens for requests from SPARQL clients

Red text means:

"This is a core part of the SPARQL syntax or language."

Blue/Black text means:

"This is an example of query-specific text or values that might go into a SPARQL query."

SPARQL Syntax Basics

RDF triple:

```
dbpedia:The_Beatles foaf:name "The Beatles" .
```

RDF triple pattern: Contains one or more variables

```
dbpedia:The_Beatles foaf:made ?album .
?album mo:track ?track .
?album ?p ?o .
```

RDF quad pattern: includes graph name (URI or variable)

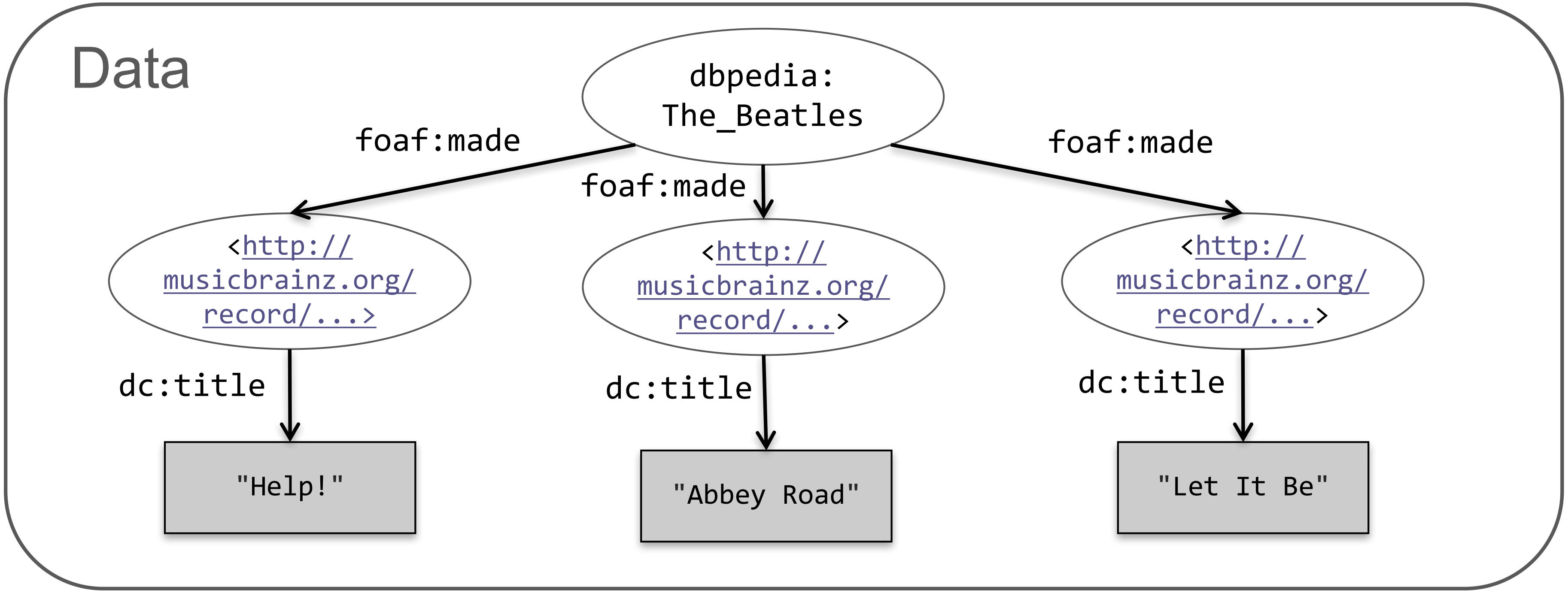
```
GRAPH <:g> {:s :p :o .}
GRAPH ?g {dbpedia:The_Beatles foaf:name ?o.}
```

- Key idea: Pattern matching
- Queries describe sub-graphs of the queried graph
- Graph patterns are RDF graphs specified in Turtle syntax which contain variables (prefixed by either ? or $)

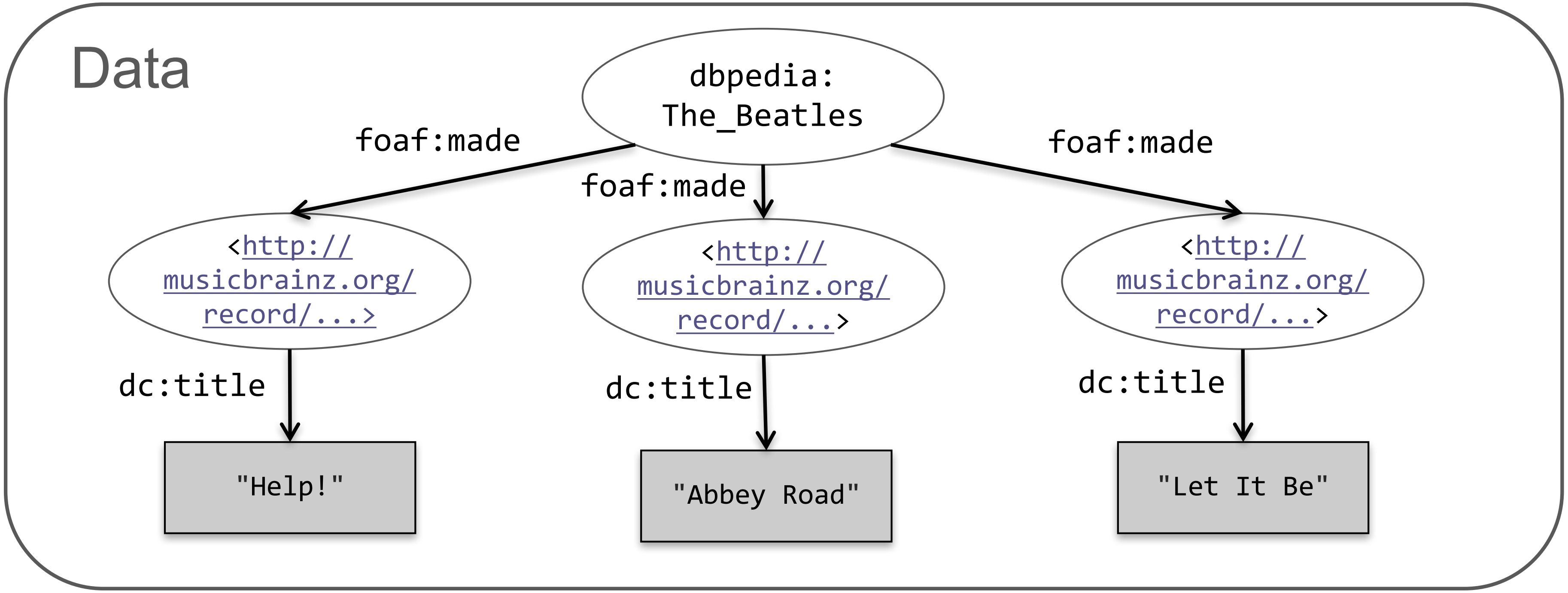
Sub-graphs that match the graph patterns yield a result

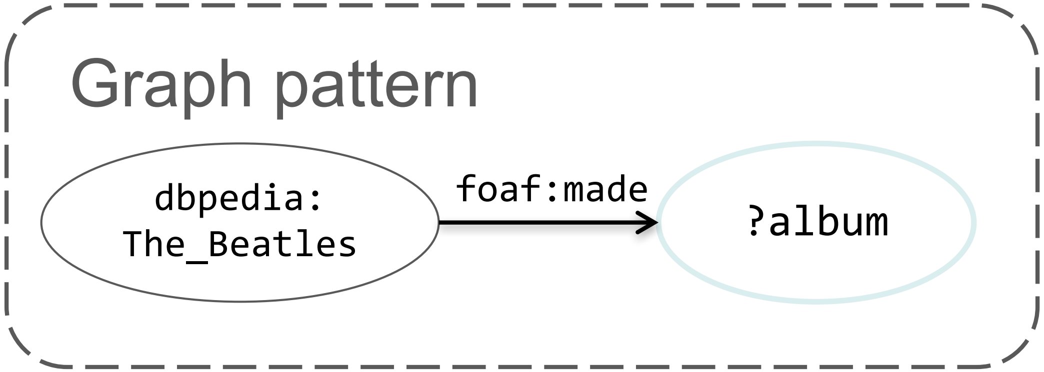
SPARQL Query Example (1)

SPARQL Query Example (1)

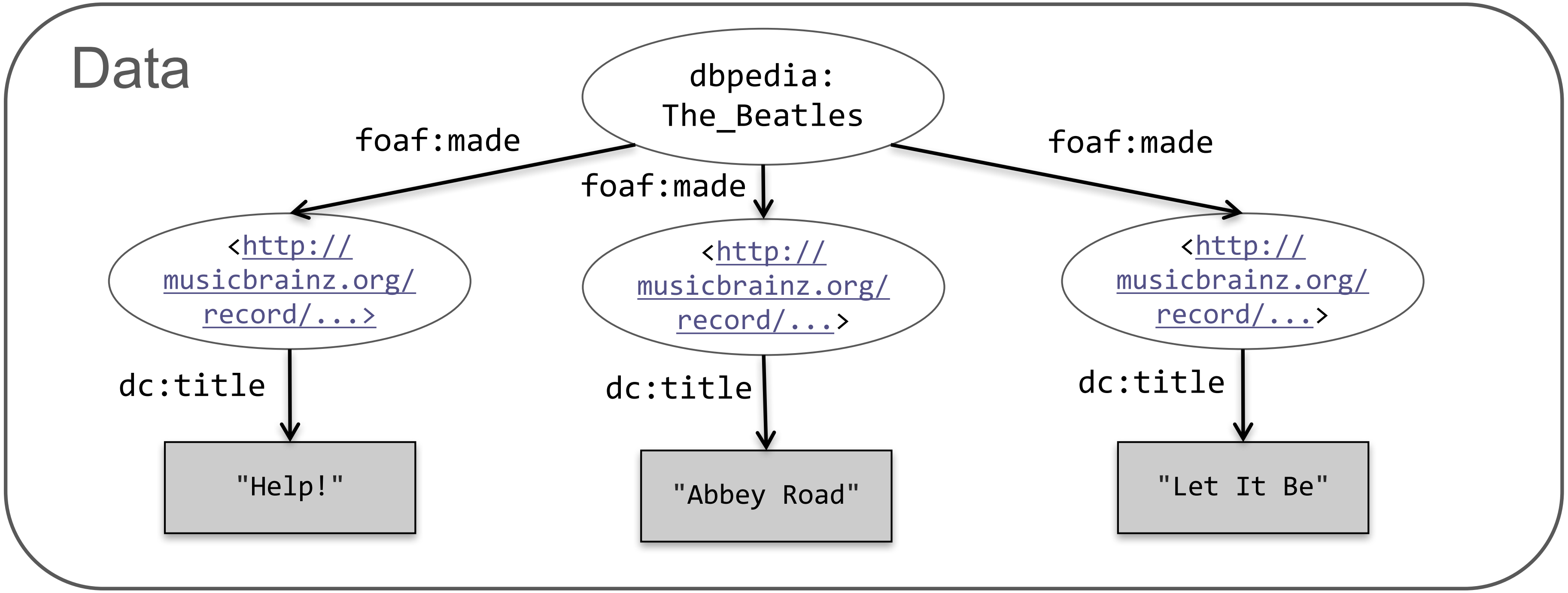


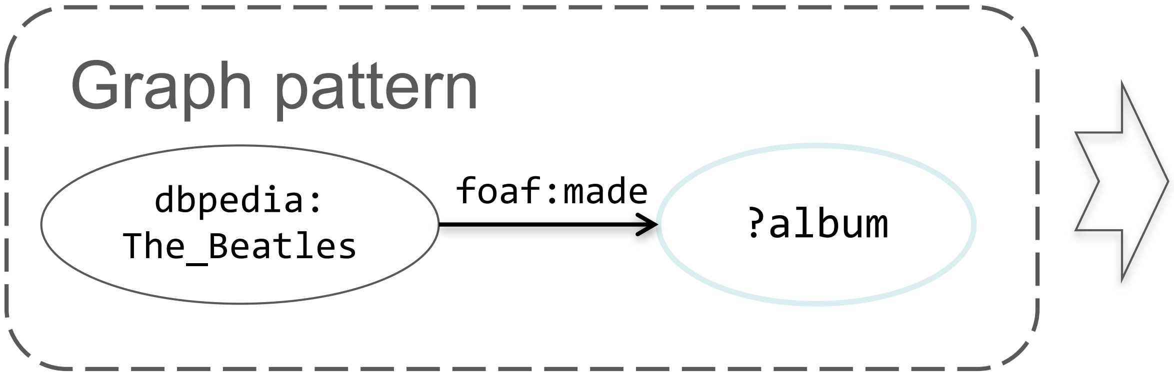
SPARQL Query Example (1)



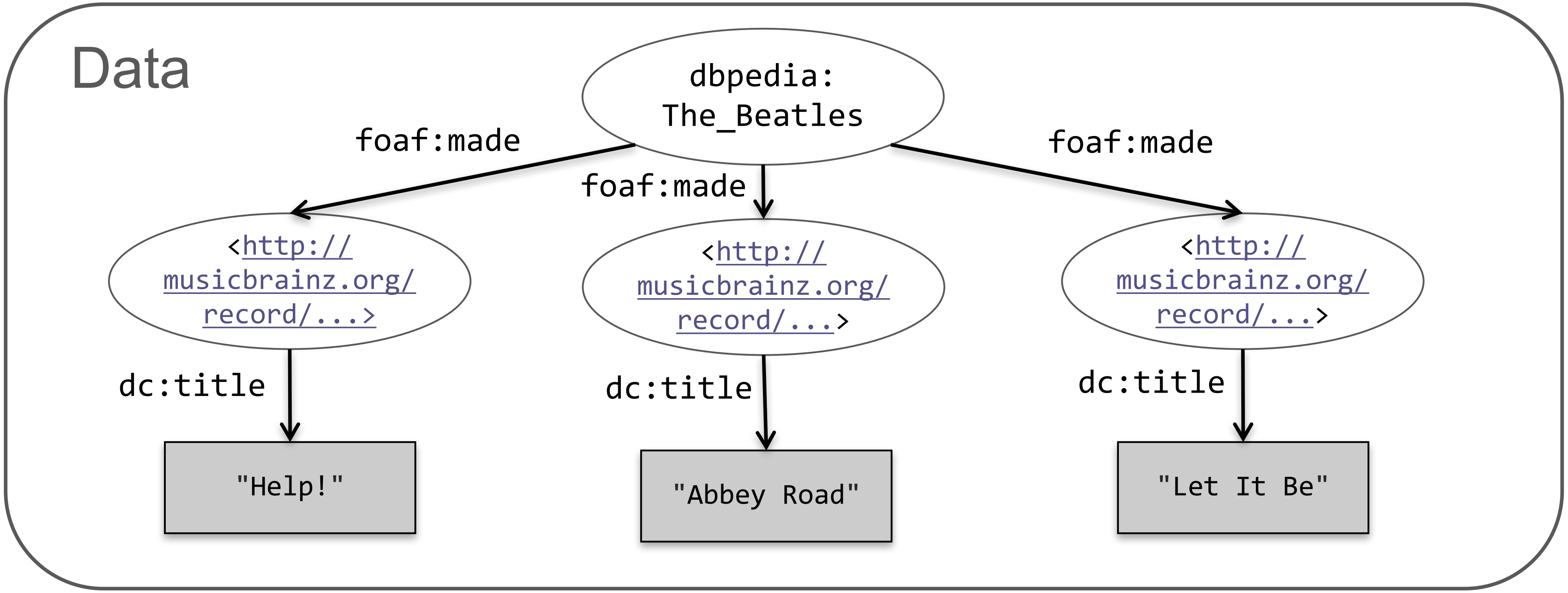


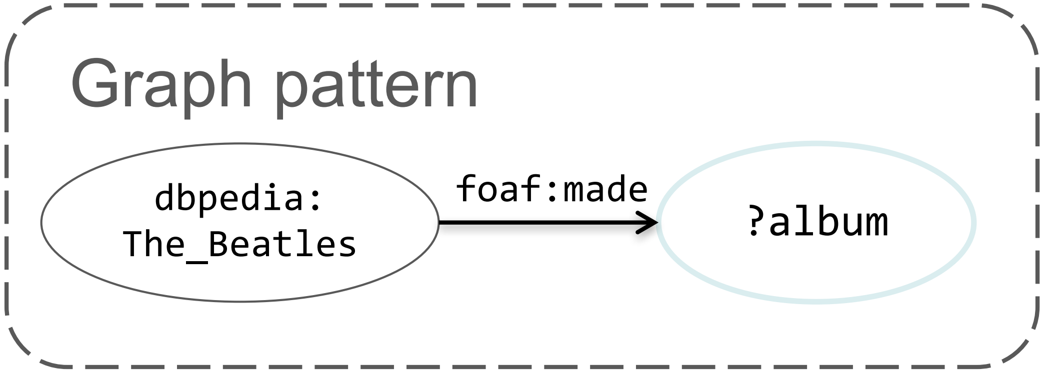
SPARQL Query Example (1)





SPARQL Query Example (1)

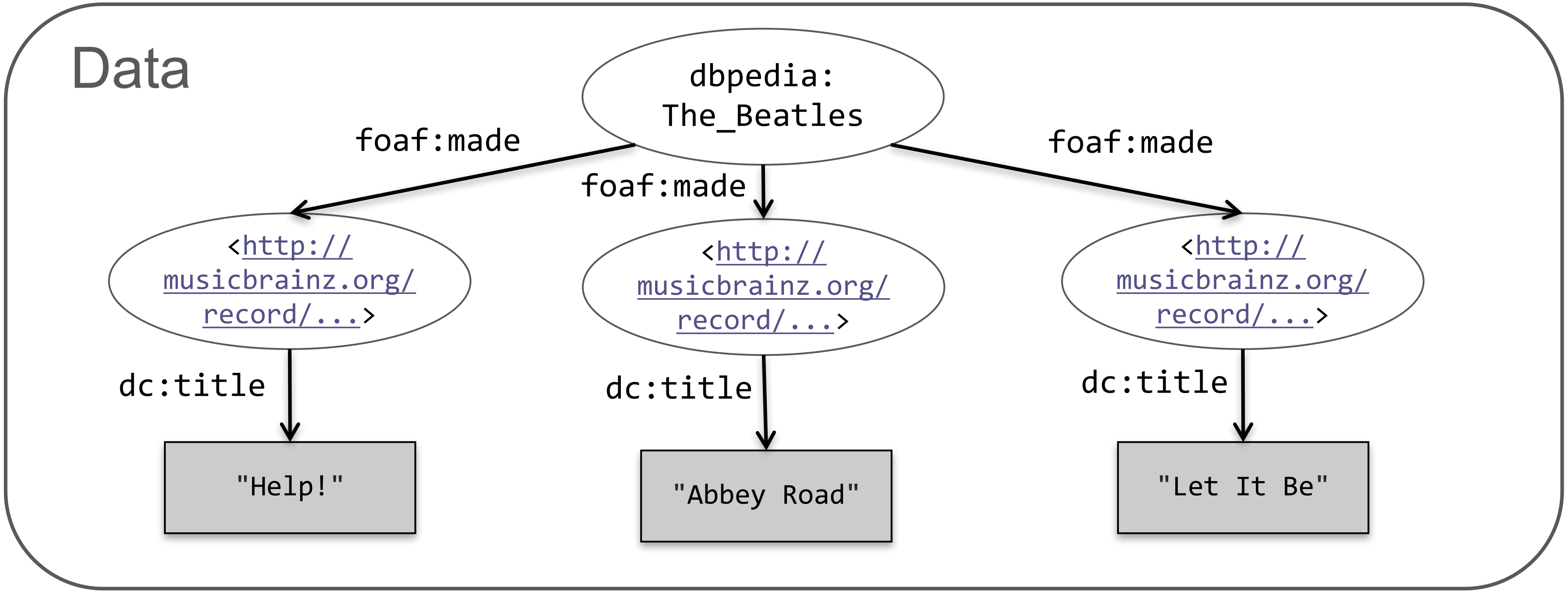




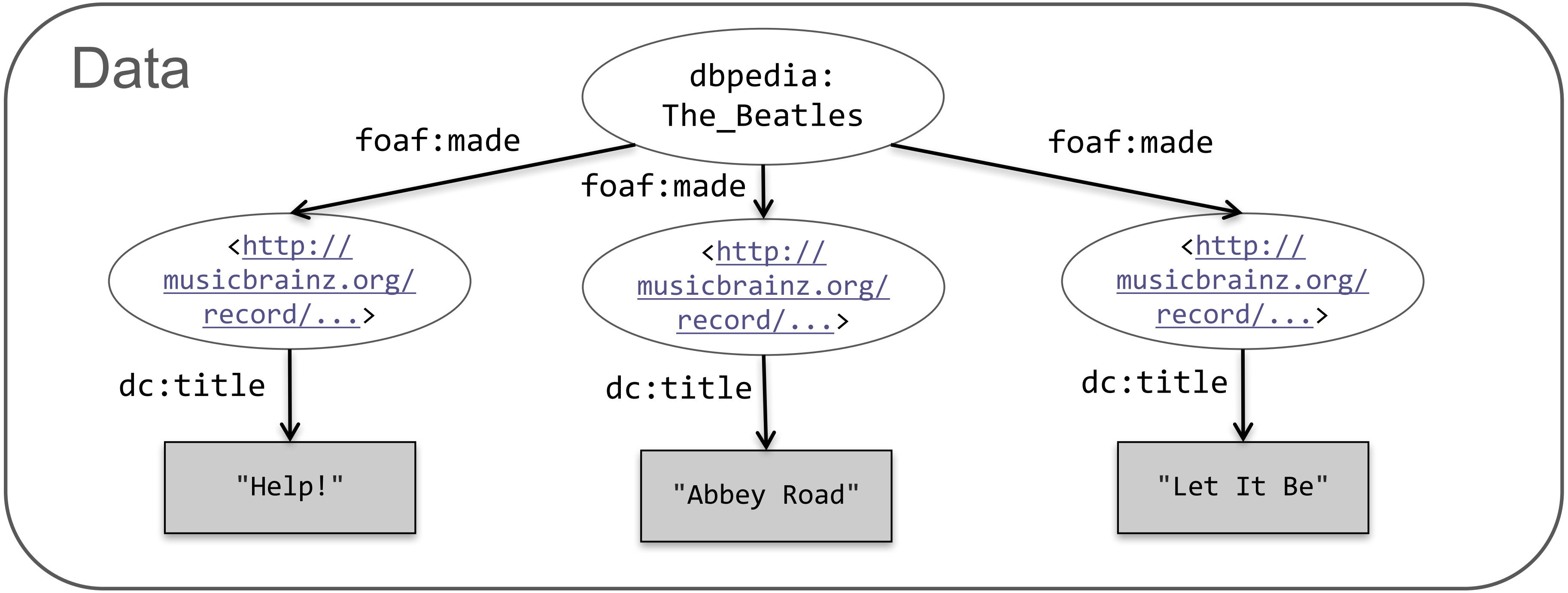
Results

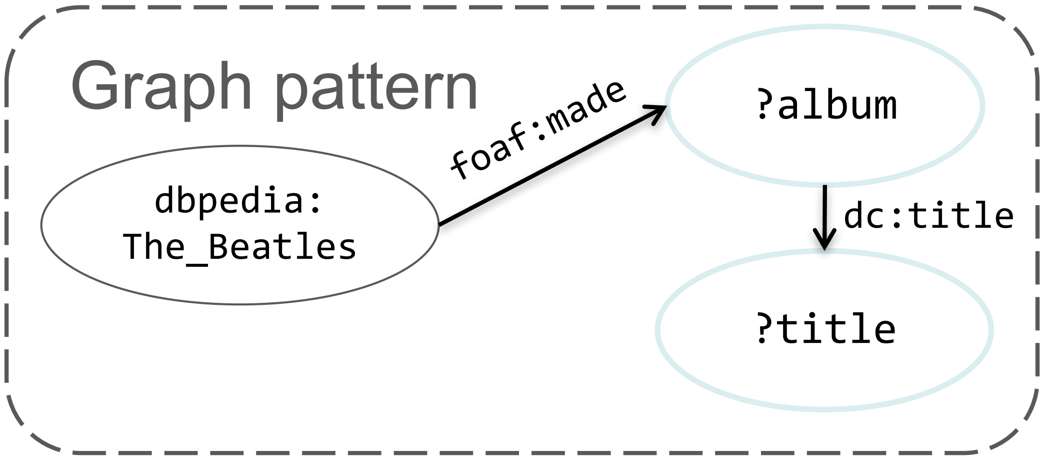
?album
<http://musicbrainz.org...>
<http://musicbrainz.org...>
<http://musicbrainz.org...>

SPARQL Query Example (2)

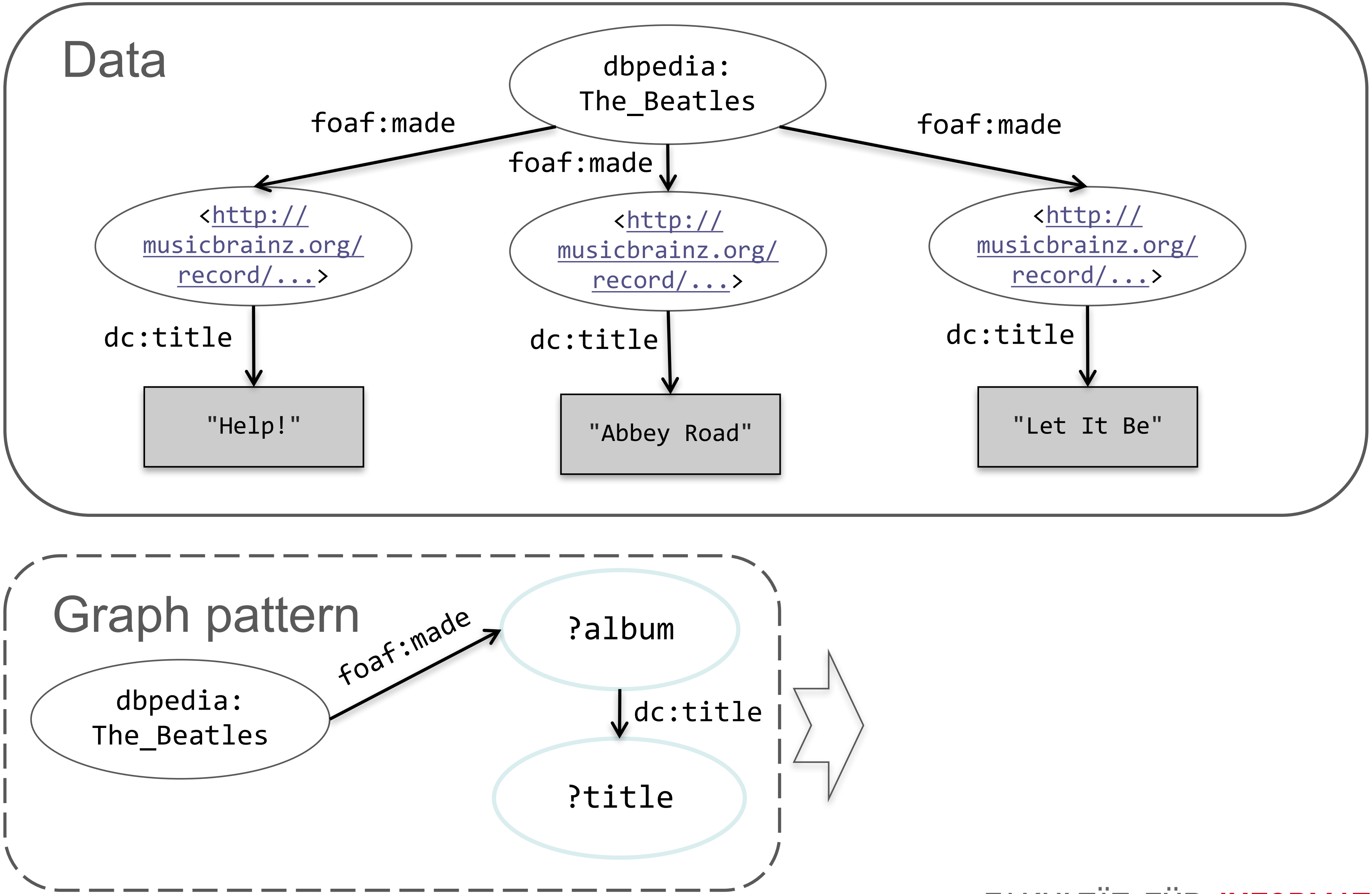


SPARQL Query Example (2)



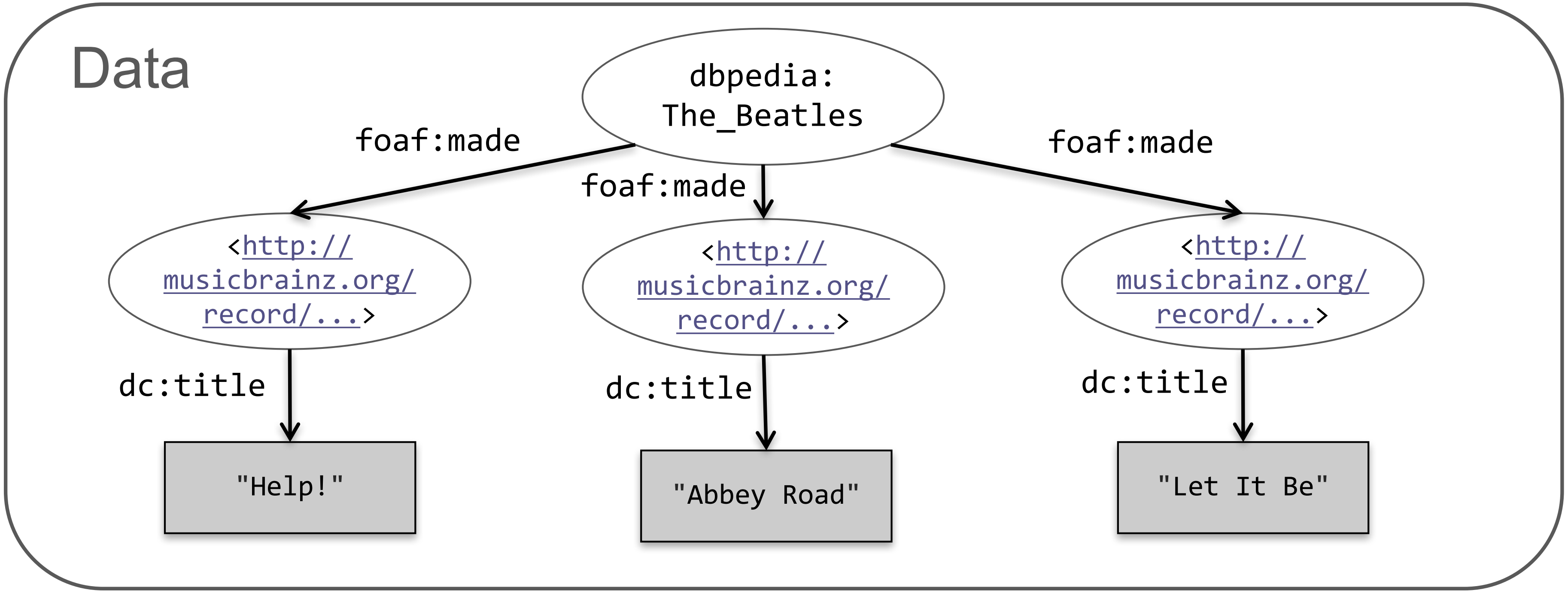


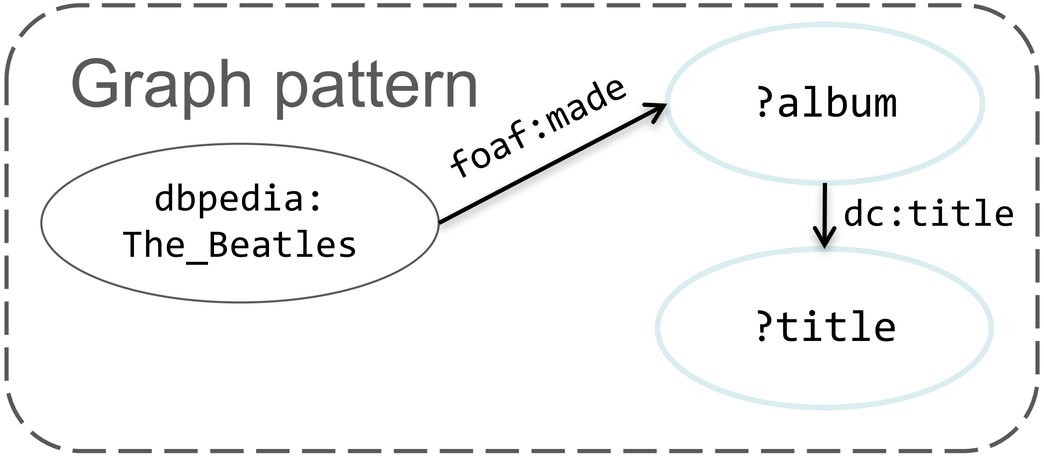
SPARQL Query Example (2)



FAKULTÄT FÜR !INFORMATIK

SPARQL Query Example (2)





Results

?album	?title
<http://...>	"Help!"
<http://...>	"Abbey Road"
<http://...>	"Let It Be"

Query Forms

SPARQL supports the following query forms:

- **ASK**
  tests whether or not a query pattern has a solution. Returns yes/no
- **SELECT**
  returns variables and their bindings directly
- **CONSTRUCT**
  returns a single RDF graph specified by a graph template
- **DESCRIBE**
  returns a single RDF graph containing RDF data about resource

Anatomy of a **SELECT** query

Declare prefix shortcuts (*optional*)

```
PREFIX foo: <…>
PREFIX bar: <…>
…
SELECT …
FROM <…>
FROM NAMED <…>
WHERE {
    …
}
GROUP BY …
HAVING …
ORDER BY …
LIMIT …
OFFSET …
VALUES …
```

Query result clause

Define the dataset (*optional*)

Query pattern

Query modifiers (*optional*)

http://www.slideshare.net/LeeFeigenbaum/sparql-cheat-sheet

Anatomy of a SELECT query

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX mo: <http://purl.org/ontology/mo/>

SELECT ?album
    FROM  <http://musicbrainz.org/20130302>
    WHERE {
        dbpedia:The_Beatles foaf:made ?album .
        ?album a mo:Record ;
               dc:title ?title
    }
        ORDER BY ?title
```

Prologue:

- Namespaces are added with the `PREFIX` directive
- Subtly different from Turtle syntax - the final period is not used

Anatomy of a SELECT query

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX mo: <http://purl.org/ontology/mo/>

SELECT ?album
    FROM  <http://musicbrainz.org/20130302>
    WHERE {
        dbpedia:The_Beatles foaf:made ?album .
        ?album a mo:Record ;
               dc:title ?title
    }
        ORDER BY ?title
```

Query form:

- ASK, SELECT, DESCRIBE or CONSTRUCT
- SELECT retrieves variables and their bindings as a table

Anatomy of a SELECT query

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX mo: <http://purl.org/ontology/mo/>

SELECT ?album
    FROM  <http://musicbrainz.org/20130302>
    WHERE {
        dbpedia:The_Beatles foaf:made ?album .
        ?album a mo:Record ;
               dc:title ?title
    }
    ORDER BY ?title
```

Data set specification:

- This clause is optional
- `FROM` or `FROM NAMED`
- Indicates the sources for the data against which to find matches

Anatomy of a SELECT query

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX mo: <http://purl.org/ontology/mo/>

SELECT ?album
    FROM  <http://musicbrainz.org/20130302>
    WHERE {
        dbpedia:The_Beatles foaf:made ?album .
        ?album a mo:Record;
               dc:title ?title
    }
       ORDER BY ?title
```

Query pattern:

- Statement patterns to match against data specified between "{}"
- Generalizes Turtle with variables and keywords (final period optional)

Anatomy of a SELECT query

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX mo: <http://purl.org/ontology/mo/>

SELECT ?album
    FROM  <http://musicbrainz.org/20130302>
    WHERE {
        dbpedia:The_Beatles foaf:made ?album .
        ?album a mo:Record;
               dc:title ?title
    }
        ORDER BY ?title
```

Solution modifier:

- Modify the result set
- `ORDER BY`, `LIMIT` or `OFFSET` reorganize rows
- `GROUP BY` combines them

Query Form: ASK

Query: *Is Paul McCartney member a of ‘The Beatles’?*

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX dbpedia-ont: <http://dbpedia.org/ontology/>
PREFIX mo: <http://purl.org/ontology/mo/>

ASK WHERE { dbpedia:The_Beatles mo:member
                               dbpedia:Paul_McCartney. }
```

Query Form: ASK

Query: *Is Paul McCartney member a of 'The Beatles'?*

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX dbpedia-ont: <http://dbpedia.org/ontology/>
PREFIX mo: <http://purl.org/ontology/mo/>

ASK WHERE { dbpedia:The_Beatles mo:member
                                dbpedia:Paul_McCartney. }
```

Results

true

Query: *Is Paul McCartney member a of 'The Beatles'?*

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX dbpedia-ont: <http://dbpedia.org/ontology/>
PREFIX mo: <http://purl.org/ontology/mo/>

ASK WHERE { dbpedia:The_Beatles mo:member
                                   dbpedia:Paul_McCartney. }
```

Results

true

Query: *Is Elvis Presley a member of 'The Beatles'?*

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX dbpedia-ont: <http://dbpedia.org/ontology/>
PREFIX mo: <http://purl.org/ontology/mo/>

ASK WHERE { dbpedia:The_Beatles mo:member
                                  dbpedia:Elvis_Presley. }
```

Query Form: ASK

Query: *Is Paul McCartney member a of 'The Beatles'?*

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX dbpedia-ont: <http://dbpedia.org/ontology/>
PREFIX mo: <http://purl.org/ontology/mo/>

ASK WHERE { dbpedia:The_Beatles mo:member
                                 dbpedia:Paul_McCartney. }
```

Results

true

Query: *Is Elvis Presley a member of 'The Beatles'?*

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX dbpedia-ont: <http://dbpedia.org/ontology/>
PREFIX mo: <http://purl.org/ontology/mo/>

ASK WHERE { dbpedia:The_Beatles mo:member
                              dbpedia:Elvis_Presley. }
```

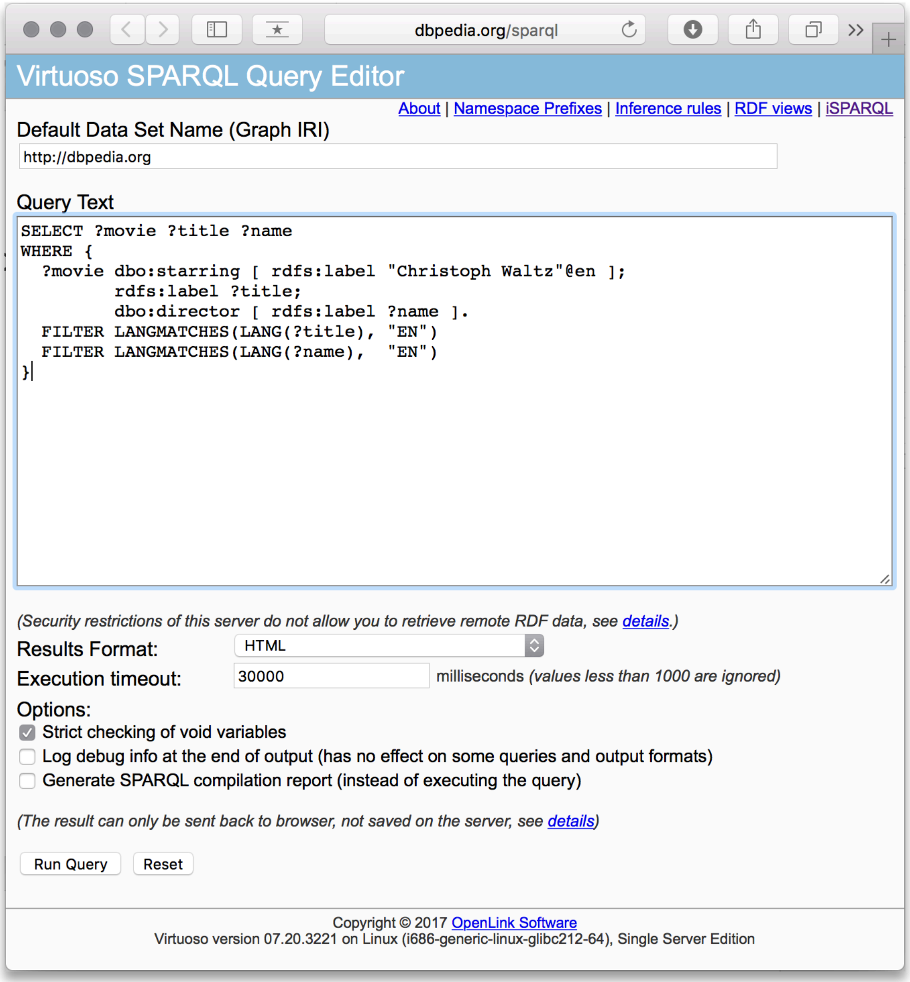
Results

false

Adapted from EUCLID learning materials – "Querying Linked Data"

Example DBPedia SELECT query

Q: Who directed movies starring Christoph Waltz?

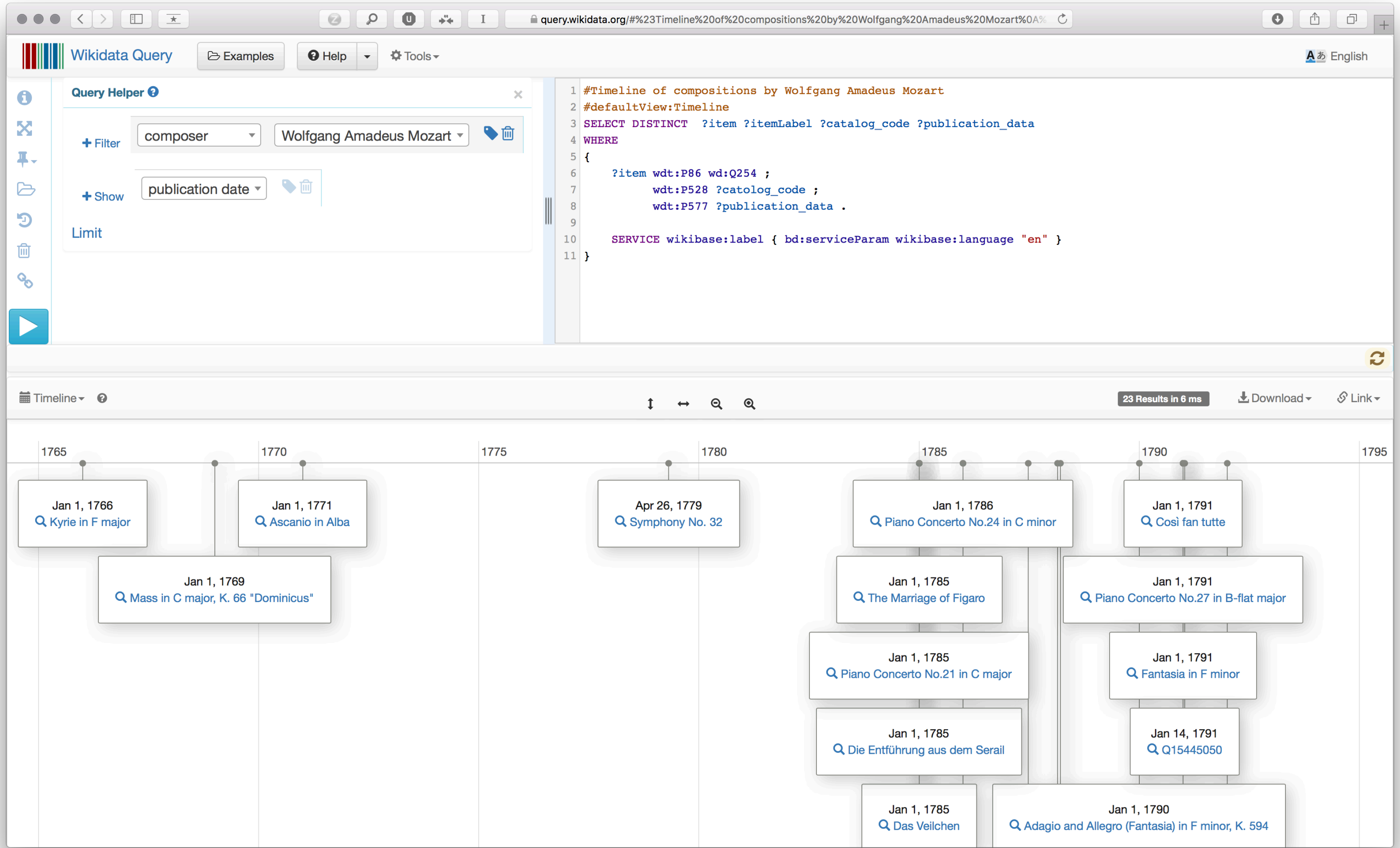


Execute: http://tinyurl.com/ycr2rgsf

Inspired by http://client.linkeddatafragments.org/ sample queries

Example Wikidata SELECT query

Q: Compositions by W.A. Mozart?



Execute: http://tinyurl.com/yc8j92b2

Query Form: DESCRIBE

Takes the resources within the solution, and provides information about them as RDF statements.

They can be identified by:

- Specifying **explicit IRIs**

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
DESCRIBE dbpedia:Paul_McCartney
```

- **Bindings of variables** in the WHERE clause

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX mo: <http://purl.org/ontology/mo/>

DESCRIBE ?member
WHERE {
        dbpedia:The_Beatles mo:member ?member .}
```

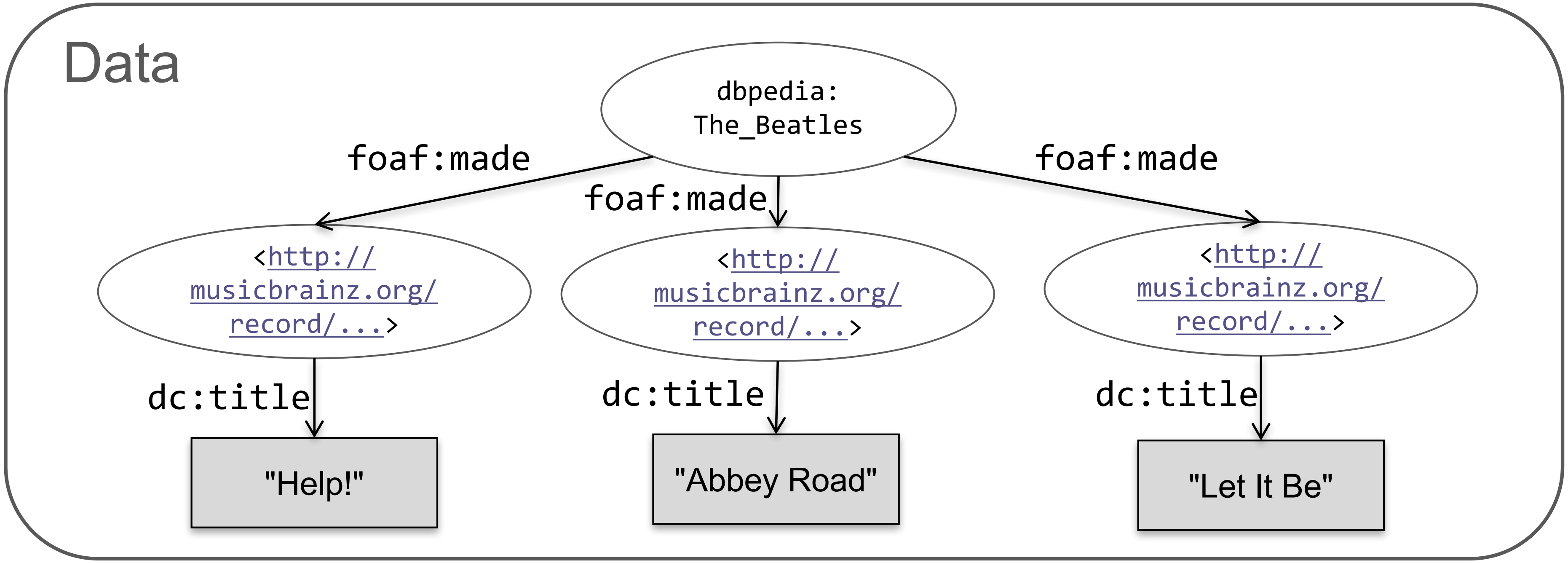
Query Form: CONSTRUCT (1)

- Returns RDF statements created from variable bindings
- Template: graph pattern with variables from the query pattern

Query: *Create the `dc:creator` descriptions for albums and their tracks recorded by 'The Beatles'.*

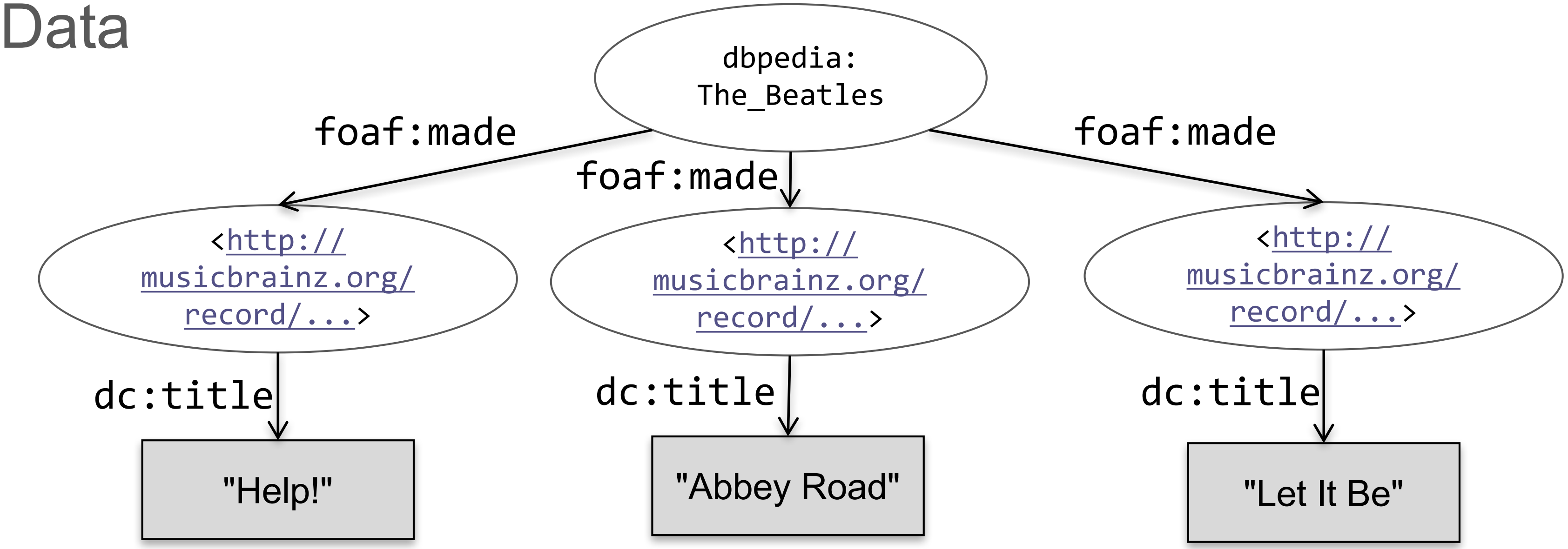
```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX mo: <http://purl.org/ontology/mo/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>

CONSTRUCT {
  ?album dc:creator dbpedia:The_Beatles .
  ?track dc:creator dbpedia:The_Beatles .}
WHERE {
  dbpedia:The_Beatles foaf:made ?album .
  ?album mo:track ?track .}
```

Query Form: CONSTRUCT (2)



Query Form: CONSTRUCT (2)

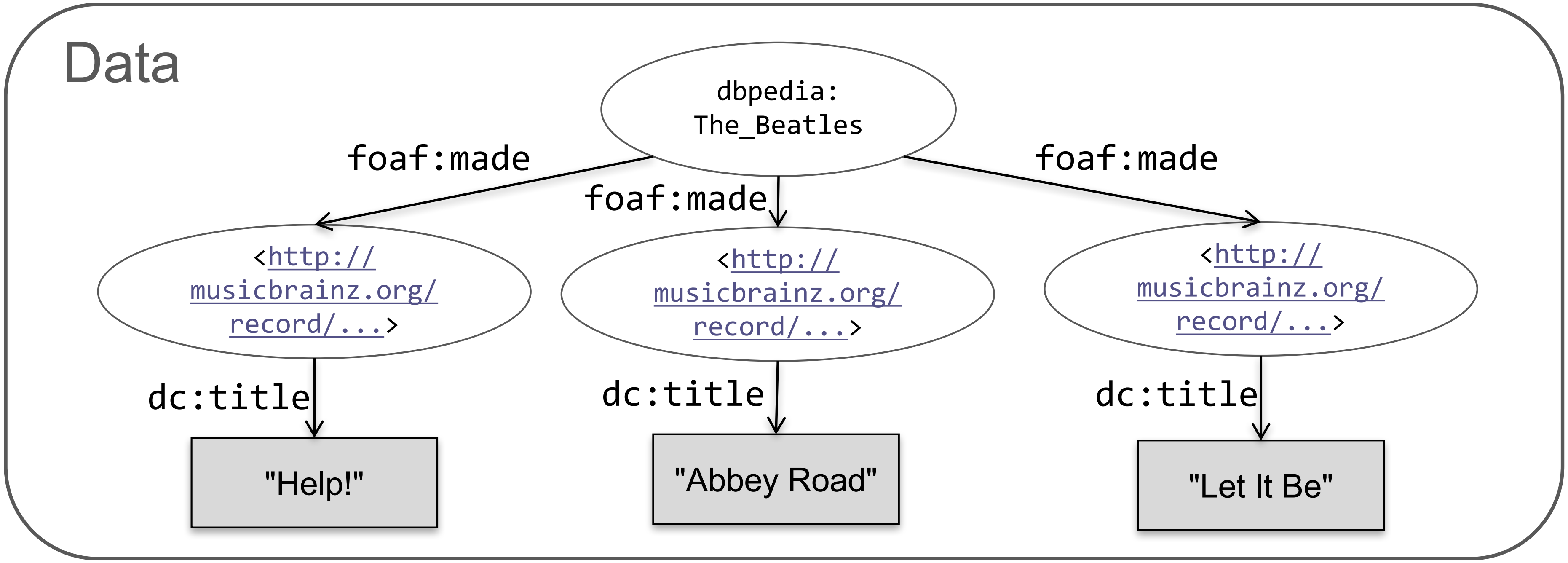
Data



Query

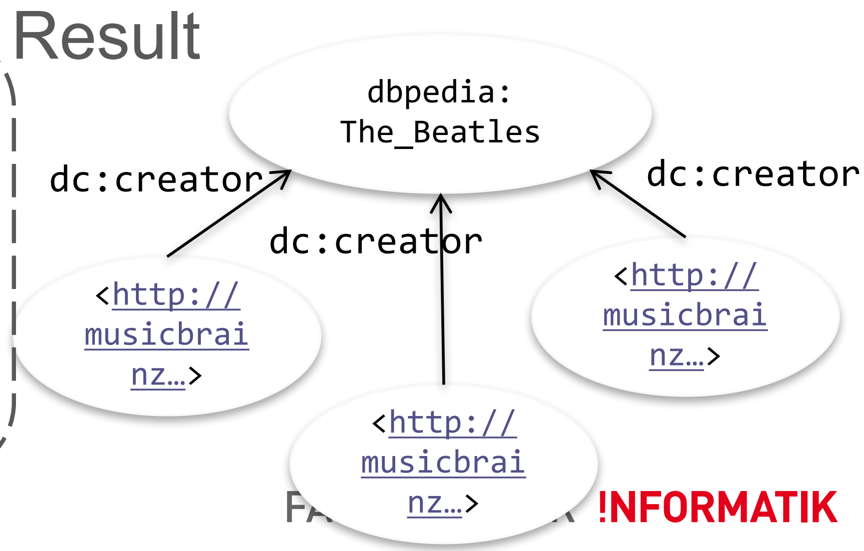
```
CONSTRUCT {
  ?album dc:creator dbpedia:The_Beatles .}
WHERE {
  dbpedia:The_Beatles foaf:made ?album .}
```

Query Form: CONSTRUCT (2)



Query

```
CONSTRUCT {
  ?album dc:creator dbpedia:The_Beatles .}
WHERE {
  dbpedia:The_Beatles foaf:made ?album .}
```



Query Form: CONSTRUCT (3)

It is possible to combine the query with solution modifiers (ORDER BY, LIMIT, OFFSET) to get **subsets of results**

Query *Create the dc:creator descriptions for the 10 most recent albums and their tracks recorded by 'The Beatles'.*

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX mo: <http://purl.org/ontology/mo/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>

CONSTRUCT {
  ?album dc:creator dbpedia:The_Beatles .
  ?track dc:creator dbpedia:The_Beatles .}
WHERE {
  dbpedia:The_Beatles foaf:made ?album .
  ?album mo:track  ?track ;
         dc:date   ?date .
} ORDER BY DESC(?date)
  LIMIT 10
```

Query Form: CONSTRUCT (4)

- **UNION** Graph Pattern
- Allows the specification of alternatives (disjunctions)

Query: *Create the dc:creator descriptions for the albums recorded by 'The Beatles' in 'Abbey Road Studios' or 'Trident Studios'*

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/
PREFIX dc: <http://purl.org/dc/elements/1.1/>

CONSTRUCT {
  ?album dc:creator dbpedia:The_Beatles .}
WHERE {
    dbpedia:The_Beatles foaf:made ?album .
    ?album dc:title ?album_name .
    {?album dbpedia-ont:recordedIn dbpedia:Abbey_Road_Studios .}
         UNION
    {?album dbpedia-ont:recordedIn dbpedia:Trident_Studios .}}
```


Adapted from EUCLID learning materials – "Querying Linked Data"

Query Form: CONSTRUCT (5)

- **Filter expressions:** regular expressions over strings
- Different types of filters and functions may be used

Query: *Create the dc:creator descriptions of the albums recorded by 'The Beatles' whose title contains the word 'love .*

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>

CONSTRUCT {?album dc:creator dbpedia:The_Beatles .}
WHERE {
    dbpedia:The_Beatles foaf:made ?album .
    ?album dc:title ?album_name ;
    FILTER (REGEX(?album_name, ".*love.*", i)) }
```

Query Form: CONSTRUCT (6)

FILTER expressions

Type of function	Function	Result type

Functional Forms	`bound`
	`IF`
	`COALESCE`
	`NOT EXISTS, EXISTS`
	`or, and`
	`RDFTerm-equal (=), sameTerm`
	`IN, NOT IN`
		`xsd:boolean`
		`rdfTerm`
		`rdfTerm`
		`xsd:boolean`
		`xsd:boolean`
		`xsd:boolean`
		`boolean`

Functions on RDF Terms	`isIRI, isBlank, isLiteral`
	`isNumeric`
	`str, lang`
	`datatype`
	`IRI`
	`BNODE`
		`xsd:boolean`
		`simple literal`
		`iri`
		`iri`
		`blank node`

Functions on Numerics	`ABS, ROUND, CEIL, FLOOR` `RAND`	`numeric` `xsd:double`

Source: http://www.w3.org/TR/sparql11-query/#SparqlOps

Query Form: CONSTRUCT (7)

FILTER expressions

Type of function	Function	Result type
Functions on Strings	STRLEN SUBSTR, UCASE, LCASE STRSTARTS, STRENDS, CONTAINS STRBEFORE, STRAFTER ENCODE_FOR_URI CONCAT langMatches REGEX REPLACE	xsd:integer string literal xsd:boolean literal simple literal string literal xsd:boolean xsd:boolean string literal
Functions on Dates and Times	now year, month, day, hours minutes seconds timezone tz	xsd:dateTime xsd:integer xsd:decimal xsd:dayTimeDurat ion simple literal

Source: http://www.w3.org/TR/sparql11-query/#SparqlOps

Query Form: CONSTRUCT (8)

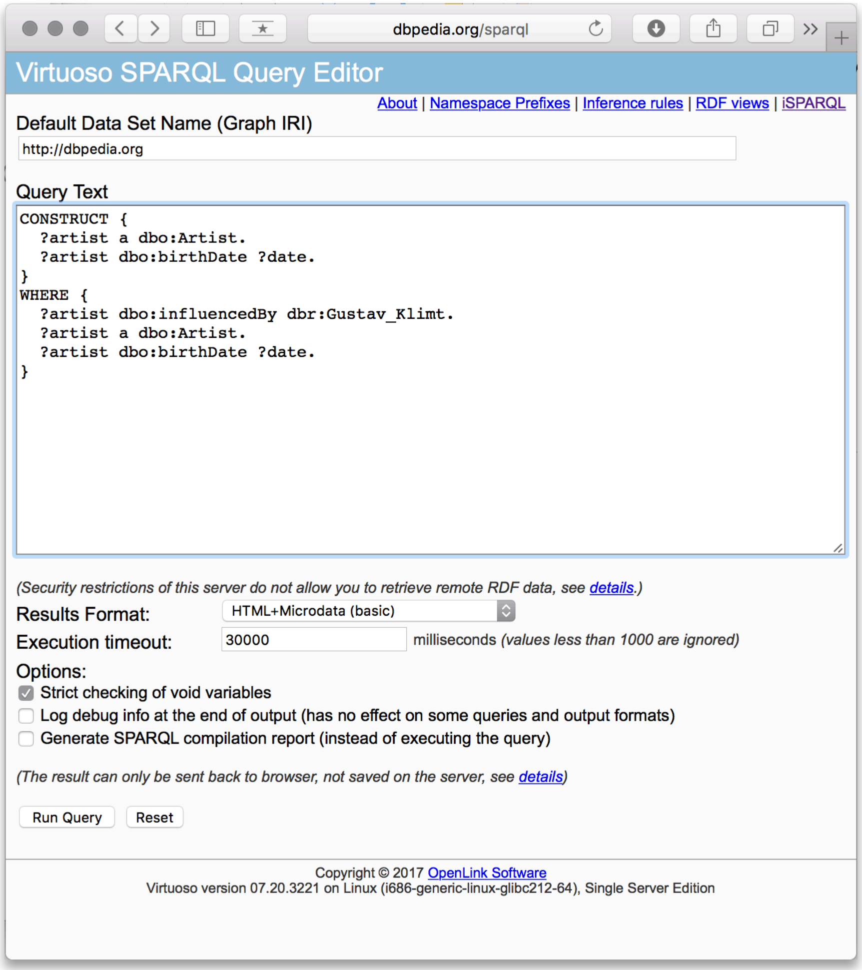
- Optional Graph Pattern
- OPTIONAL clause encloses the optional parts
- If variables in the construct clause are not bound in the optional, the triple patterns with these variables are not generated

Query: *Create the dc:creator and dc:depicts descriptions of artists.*

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>

CONSTRUCT {
    ?album dc:creator ?artist .
    ?picture dc:depicts ?artist . }
WHERE {
    ?artist foaf:made ?album .
    OPTIONAL {?artist foaf:depiction ?picture .}}
```

Example CONSTRUCT query

Q: Which Artists where inspired by Gustav Klimt?



Execute: http://tinyurl.com/ycfxjfr7

Inspired by http://client.linkeddatafragments.org/ sample queries

Combining SPARQL graph patterns (Summary)

- *Basic Graph Pattern:* `A . B`

  ⇨ Conjunction. Join together the results of solving A and B by matching the values of any variables in common.

- *Optional Graph Pattern:* `A OPTIONAL { B }`

  ⇨ Left join. Join together the results of solving A and B by matching the values of any variables in common, if possible. Keep all solutions from A whether or not there's a matching solution in B

- *Either-or Graph Patterns:* `{ A } UNION { B }`

  ⇨ Disjunction. Include both the results of solving A and the results of solving B.

- *"Subtracted" Graph Patterns (SPARQL 1.1):* `A MINUS { B }`

  ⇨ Negation. Solve A. Solve B. Include only those results from solving A that are *not compatible* with any of the results from B.

http://www.slideshare.net/LeeFeigenbaum/sparql-cheat-sheet

Summary: Querying Linked Data

- Basic concepts:
  triple patterns, graph patterns, SPARQL endpoint ...
- SPARQL Query:
  - Query forms: `ASK`, `SELECT`, `DESCRIBE`, `CONSTRUCT`
  - Query patterns: `BGP`, `UNION`, `OPTIONAL`, `FILTER`
  - Sequence modifiers: `DISTINCT`, `REDUCED`, `ORDER BY`, `LIMIT`, `OFFSET`
- SPARQL 1.1 Update:
  - Data management: `INSERT`, `DELETE`; `DELETE/INSERT`
  - Graph management: `LOAD`, `CLEAR`, `CREATE`, `DROP`, `COPY/MOVE/ADD`
- SPARQL Protocol: query operation, update operation

Q3: Official Languages of countries sharing a border with Austria

```
SELECT DISTINCT ?languageLabel WHERE {
      ?country  wdt:P47 wd:Q40;
                 wdt:P37 ?language.
      SERVICE wikibase:label {
            bd:serviceParam wikibase:language "en".
      }
}
```

SPARQL Resources

- The SPARQL specification

  http://www.w3.org/TR/rdf-sparql-query/

- SPARQL implementations

  http://esw.w3.org/topic/SparqlImplementations

- Public SPARQL endpoints

  http://esw.w3.org/topic/SparqlEndpoints

- Sparql endpoint health stats

  http://sparqles.ai.wu.ac.at

- SPARQL By Example

  http://www.cambridgesemantics.com/sparql-by-example/slides.html

EUCLID

Acknowledgements

- Alexander Mikroyannidis
- Alice Carpentier
- Andreas Harth
- Andreas Wagner
- Andriy Nikolov
- Barry Norton
- Daniel M. Herzig
- Elena Simperl
- Günter Ladwig
- Inga Shamkhalov
- Jacek Kopecky
- John Domingue
- Juan Sequeda
- Kalina Bontcheva
- Maria Maleshkova
- Maria-Esther Vidal
- Maribel Acosta
- Michael Meier
- Ning Li
- Paul Mulholland
- Peter Haase
- Richard Power
- Steffen Stadtmüller