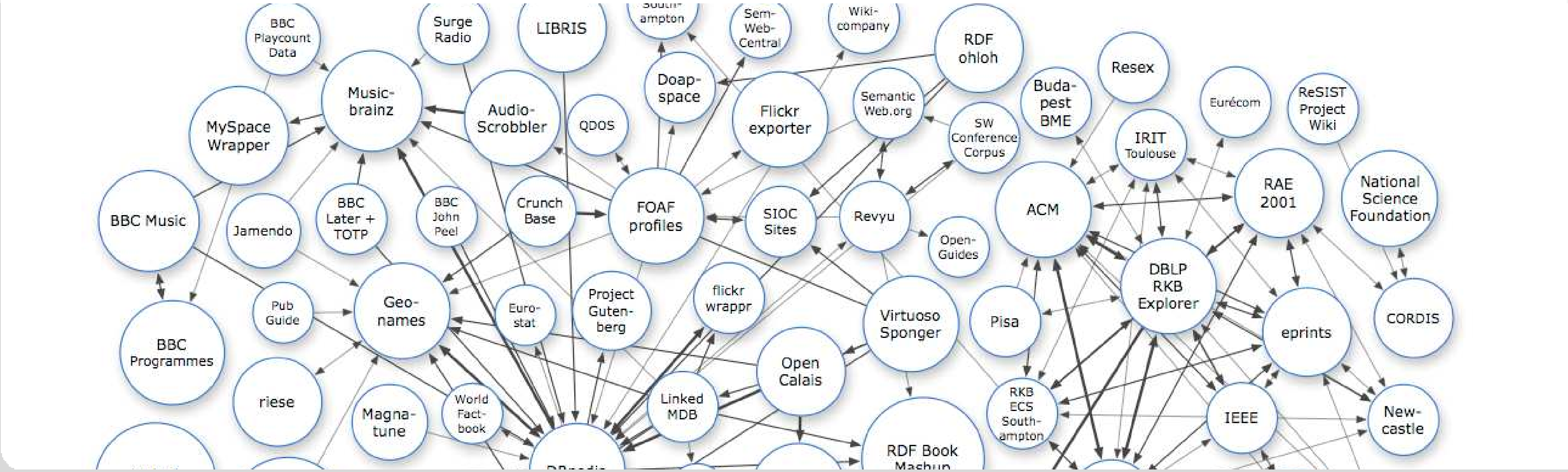
Karlsruhe Institute of Technology

Linked Data

Semantic Web Technologies 1 (2010/2011)

Sebastian Rudolph **Andreas Harth**

Institute AIFB

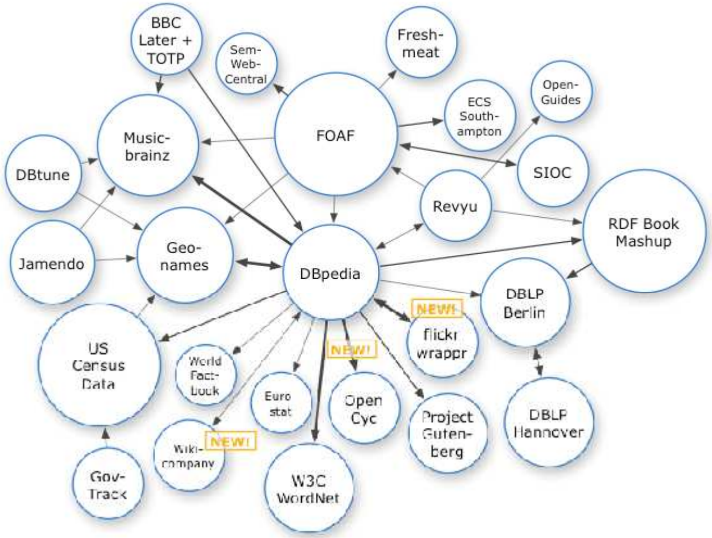


Data on the Web

- Increasingly, web sites provide direct access to data
- Using Semantic Web standards, e.g., via the Linking Open Data (LOD) initiative
- Using APIs, e.g., via JSON/REST

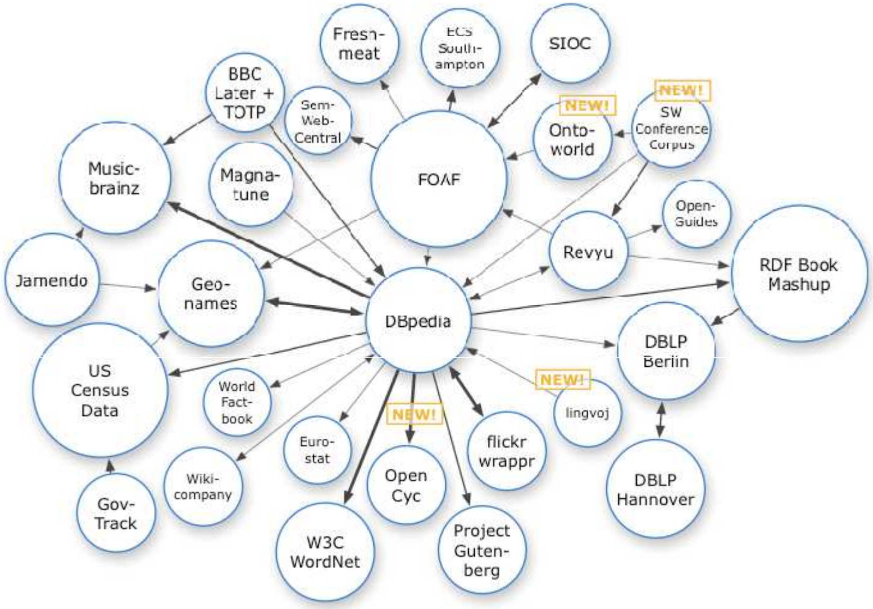
- Semantic Web technologies facilitate the integration of data from multiple sources
- Combining data from multiple sources enables insights

Linked Data on the Web



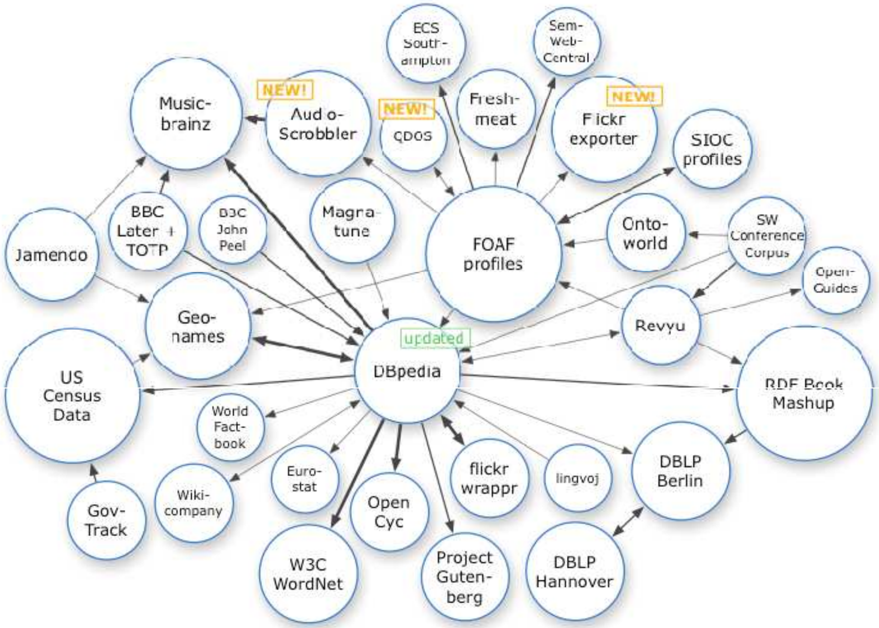
2007-10

Linked Data on the Web



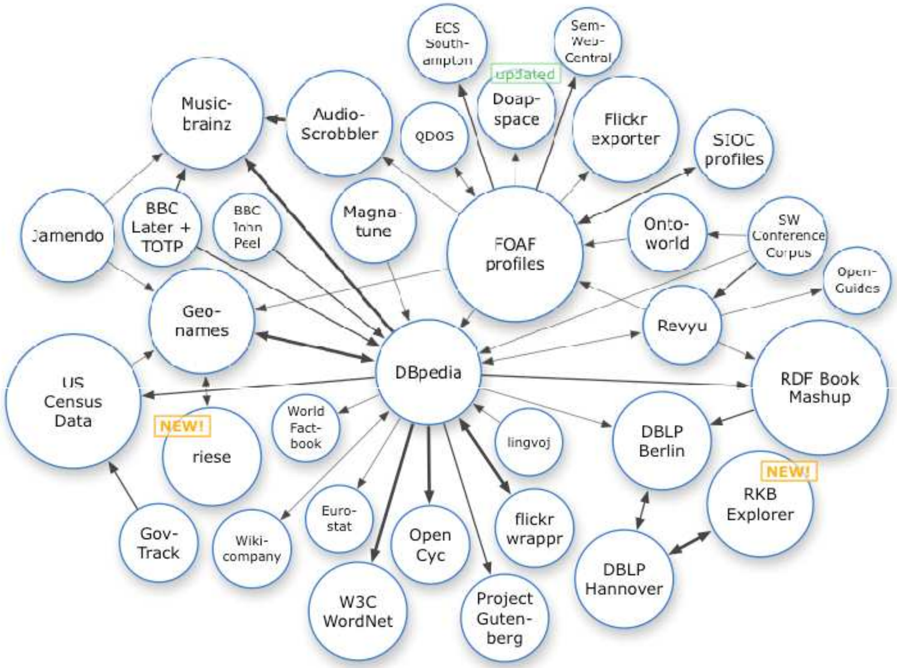
2007-11

Linked Data on the Web



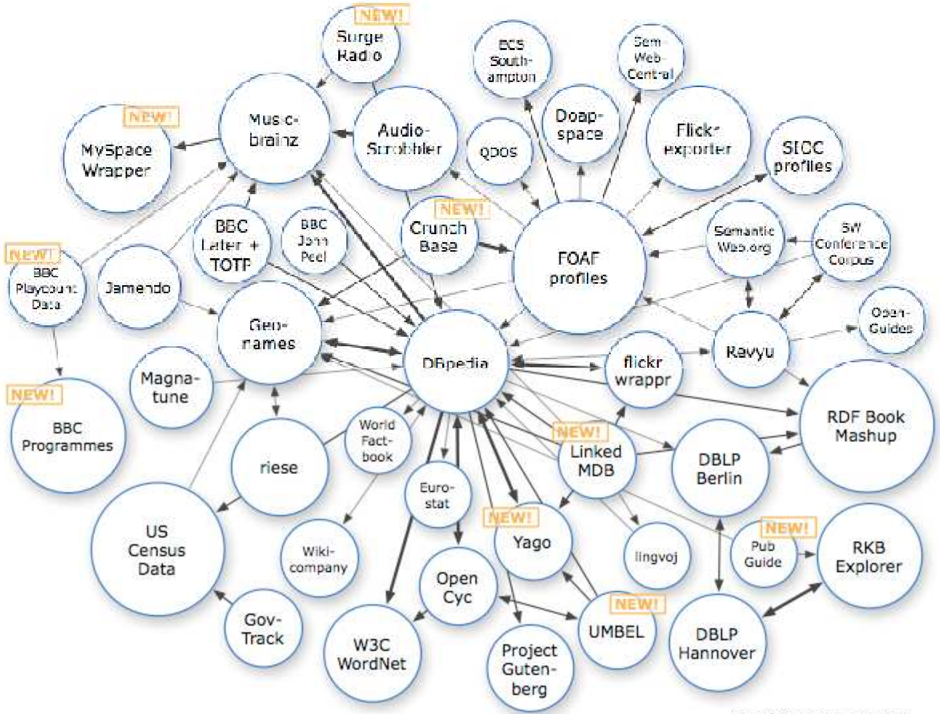
2008-02

Linked Data on the Web



2008-03

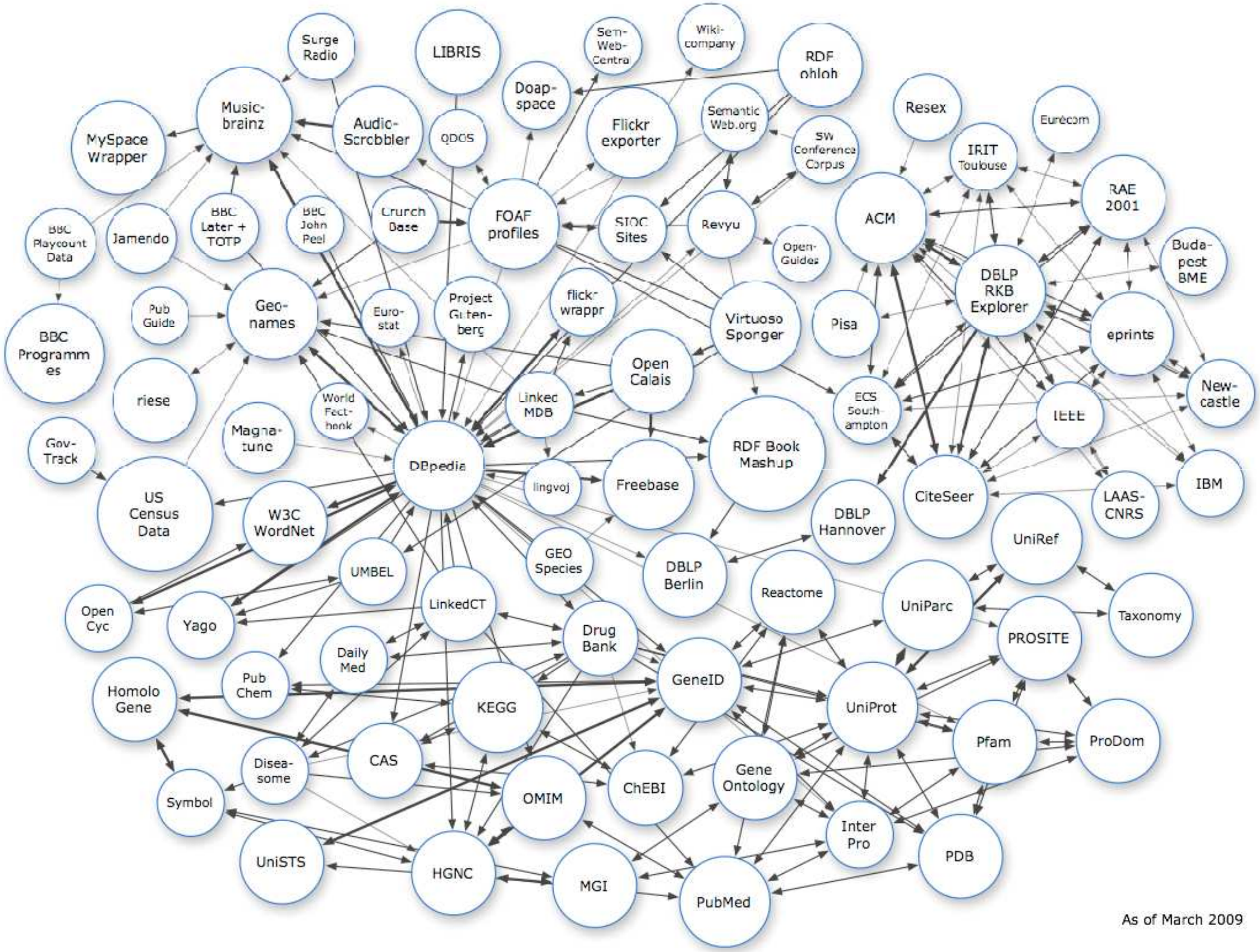
Linked Data on the Web



As of September 2008

2008-09

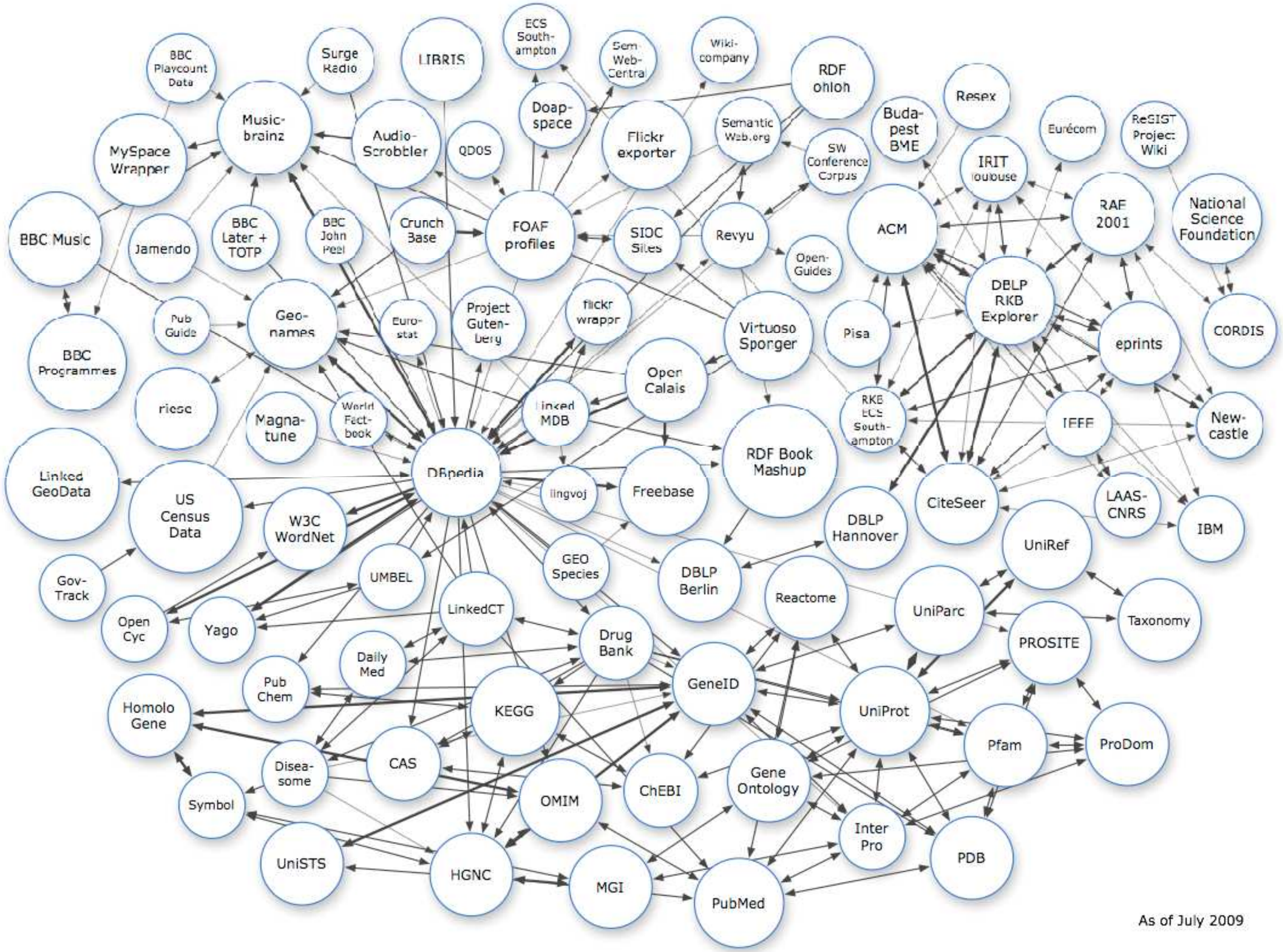
Linked Data on the Web



As of March 2009

2009-03

Linked Data on the Web



As of July 2009

2009-07

Semantic Web Technologies

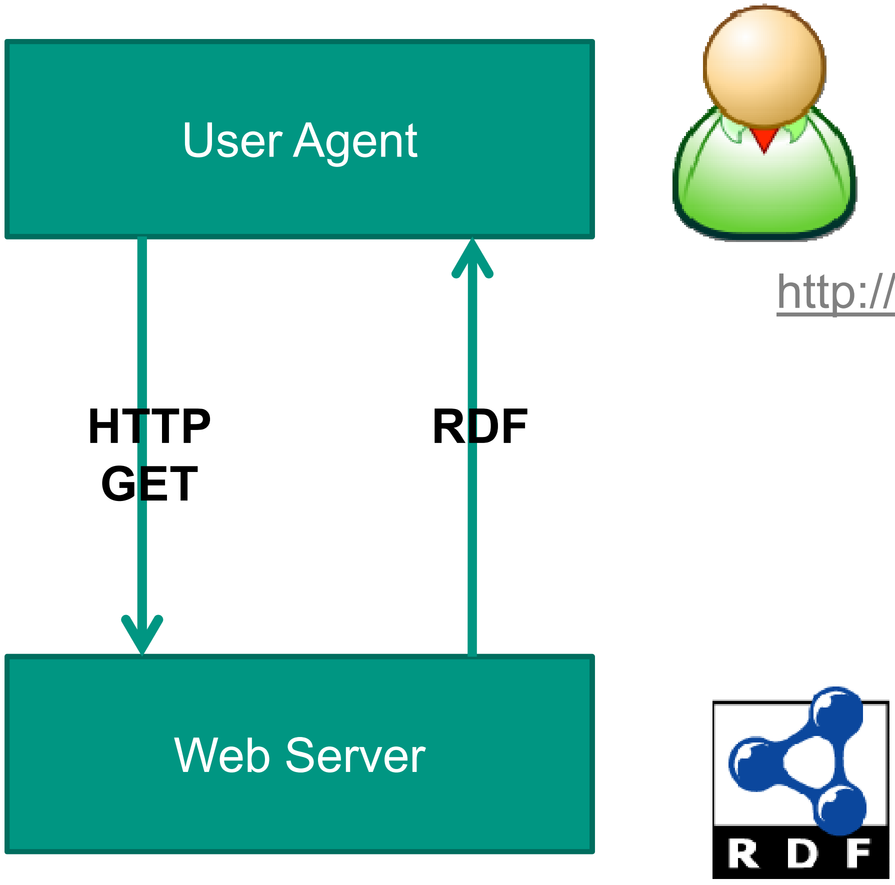
- Useful for data publishing, exchange, and integration
- Insights possible when combining data from multiple sources
- Semantic Web technologies are mature:
  - IRIs (IETF RFC 3987, 2005)
  - HTTP (IETF RFC 2616, 1999)
  - RDF (W3C recommendation, 1999, update in 2004)
  - RDFS (W3C recommendation, 2004)
  - SPARQL (W3C recommendation, 2008)
  - OWL (W3C recommendation, 2004, update in 2009)
- Linked Data comprises a few principles for data publishing on the web

Linked Data Principles*

1. Use **URIs to name *things***; not only documents, but also people, locations, concepts, etc.
2. To enable agents (human users and machine agents alike) to look up those names, use **HTTP URIs**
3. When someone looks up a URI we **provide useful information**; with 'useful' in the strict sense we usually mean structured data in RDF.
4. Include **links to other URIs** allowing agents (machines and humans) to **discover more *things***

(*) http://www.w3.org/DesignIssues/LinkedData.html

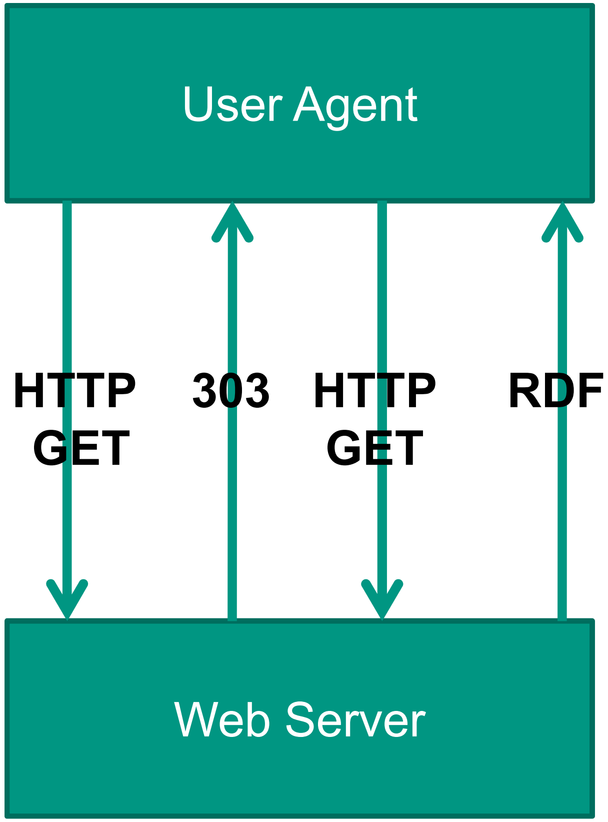
Correspondence between thing-URI and source-URI



http://www.polleres.net/foaf.rdf#me

http://www.polleres.net/foaf.rdf

Correspondence between thing-URI and source-URI



http://dbpedia.org/resource/Gordon_Brown

http://dbpedia.org/data/Gordon_Brown

http://dbpedia.org/page/Gordon_Brown

Background: Web Architecture & RDF

- URIs and HTTP
- RDF (Resource Description Framework)
- Ontologies (very brief)

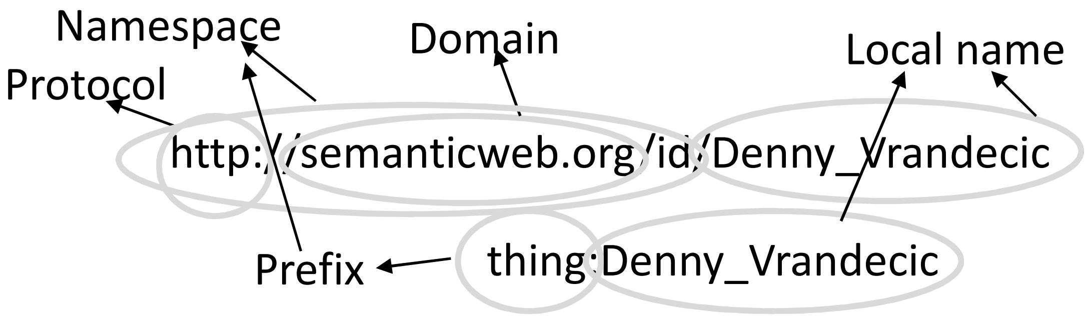
Uniform Resource Identifiers

- A Uniform Resource Identifier (URI) is a compact **sequence of characters** that **identifies** an abstract or physical **resource**. [RFC3986]
- Syntax

  URI = scheme ":" hier-part ["?" query] ["#" fragment]
- Example

```
foo://example.com:8042/over/there?name=ferret#nose
\_/   \______________/\_________/ \_________/ \__/
 |           |            |            |        |
scheme     authority       path        query   fragment
```

URIs/IRIs



Namespace

Domain

Local name

Protocol

http://semanticweb.org/id/Denny_Vrandecic

Prefix

thing:Denny_Vrandecic

- URIs are “Uniform Resource Identifiers”
  - IRI: Unicode-based “Internationalized Resource Identifiers”
- Every URI identifies one entity
- Semantic Web URIs usually use HTTP
  - HyperText Transfer Protocol
  - Can be resolved to get more data (ideally)
  - Linked Data

The Hypertext Transfer Protocol (HTTP) is

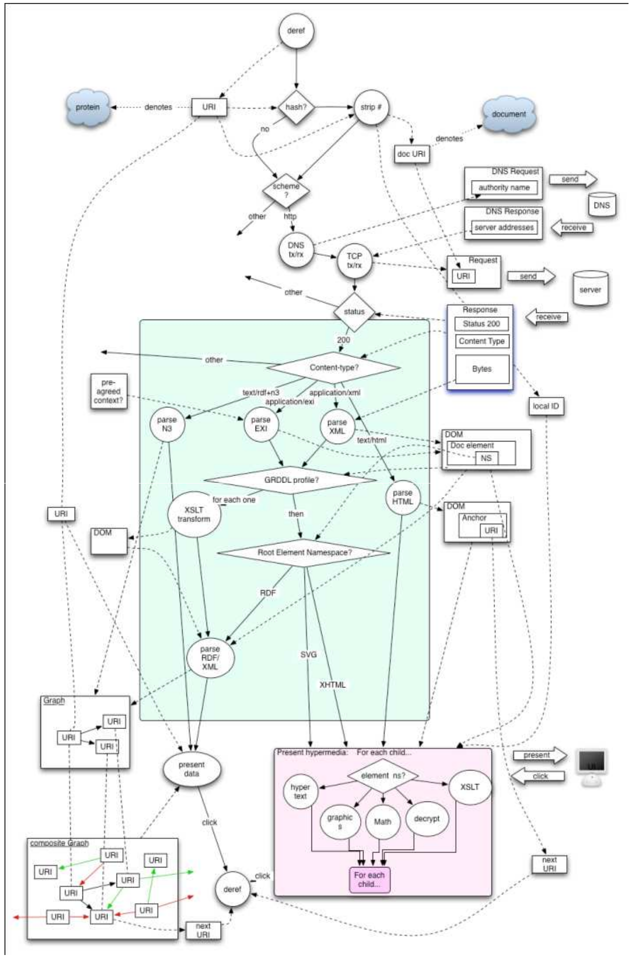
- an application-level protocol for distributed, collaborative, hypermedia information systems
- a generic, stateless, protocol which can be used for many tasks beyond its use for hypertext
- a protocol which includes the typing and negotiation of data representation, allowing systems to be built independently of the data being transferred.

[RFC2616]

HTTP Overview

- HTTP messages consist of **requests** from client to server and **responses** from server to client
- Set of **methods** is predefined (such as GET, POST, etc.), but can be expanded
- Set of **status codes** is defined
  - Informational 1xx, provisional response, (*100 Continue*)
  - Successful 2xx, request successfully received, understood, and accepted (*201 Created*)
  - Redirection 3xx, further action needs to be taken by user agent to fulfill the request (*301 Moved Permanently*)
  - Client Error 4xx, client erred (*405 Method Not Allowed)*
  - Server Error 5xx, server encountered an unexpected condition (*501 Not Implemented*)

HTTP Lookups



Web's Standard Retrieval Algorithm as of [SDD]:

1. parse URI and find HTTP protocol
2. look up DNS name to determine the associated IP address
3. open a TCP stream to port 80 at the IP address determined above
4. format an HTTP GET request for resource and sends that to the server
5. read response from the server
6. from the status code (200) determine that a representation of the resource is available
7. inspect the returned Content-Type
8. pass the entity-body to its HTML rendering engine

HTTP Example Request/Response

REQUEST

```
GET /html/rfc2616 HTTP/1.1
Host: tools.ietf.org
User-Agent: Mozilla/5.0
Accept: text/html,application/xhtml+xml;q=0.9,*/*
```

RESPONSE

```
HTTP/1.x 200 OK
Date: Thu, 05 Mar 2009 08:17:33 GMT
Server: Apache/2.2.11
Content-Location: rfc2616.html
Last-Modified: Tue, 20 Jan 2009 09:16:04 GMT
Content-Type: text/html; charset=UTF-8
```

HTTP Content Negotiation

- Content Negotiation (CN, conneg) is the process of **selecting** the best representation for a given response when there are **multiple representations available**
- Three types: server-driven, agent-driven, transparent

```
$ curl -H "Accept: application/rdf+xml“
   http://dbpedia.org/resource/Galway

HTTP/1.1 303 See Other
Content-Type: application/rdf+xml
Location: http://dbpedia.org/data/Galway.rdf
$
```

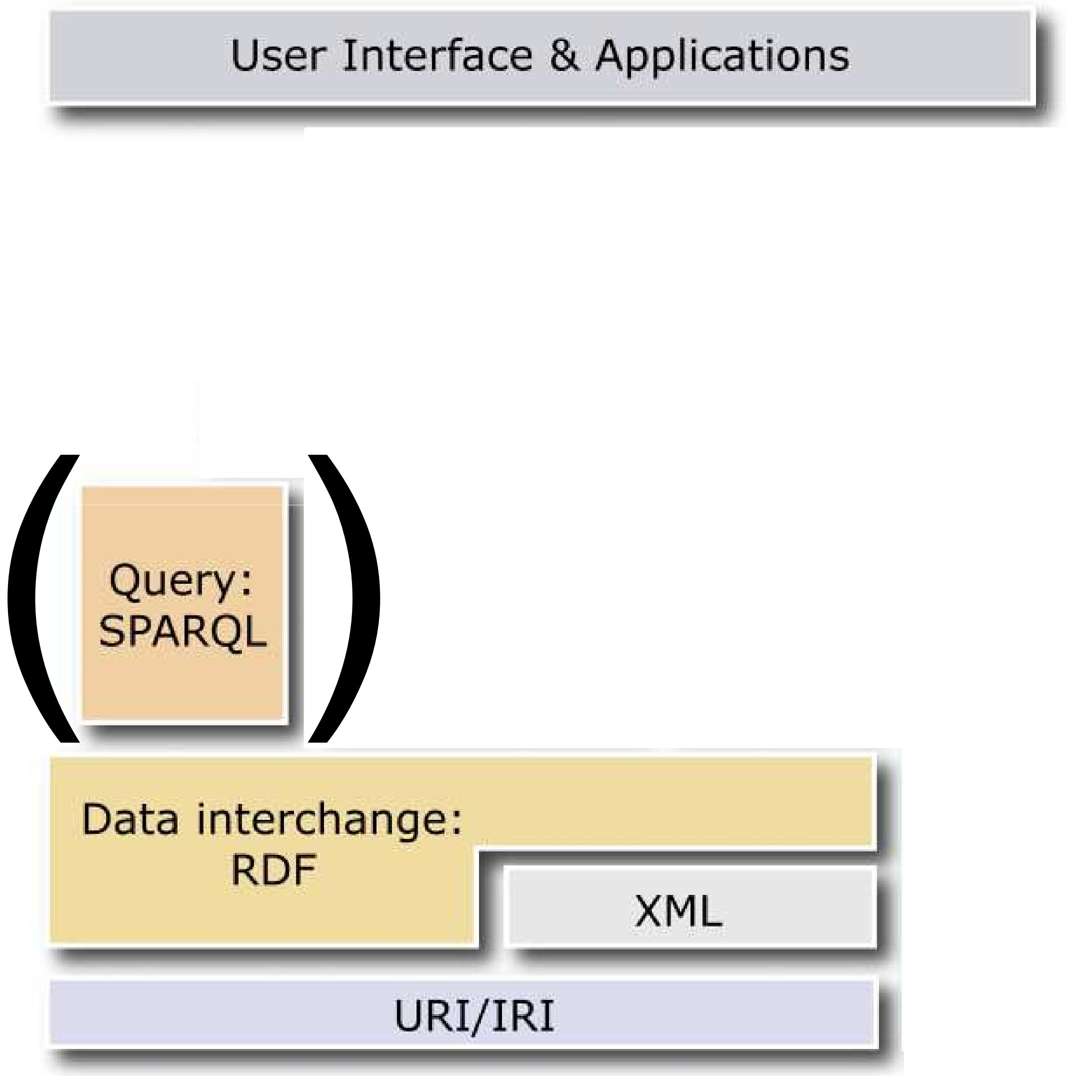
RDF as Linked Data

```
<?xml version="1.0"?>

<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:foaf="http://xmlns.com/foaf/0.1/">

  <foaf:Person rdf:about="#ah">
    <foaf:name>Andreas Harth</foaf:name>
  </foaf:Person>
</rdf:RDF>
```

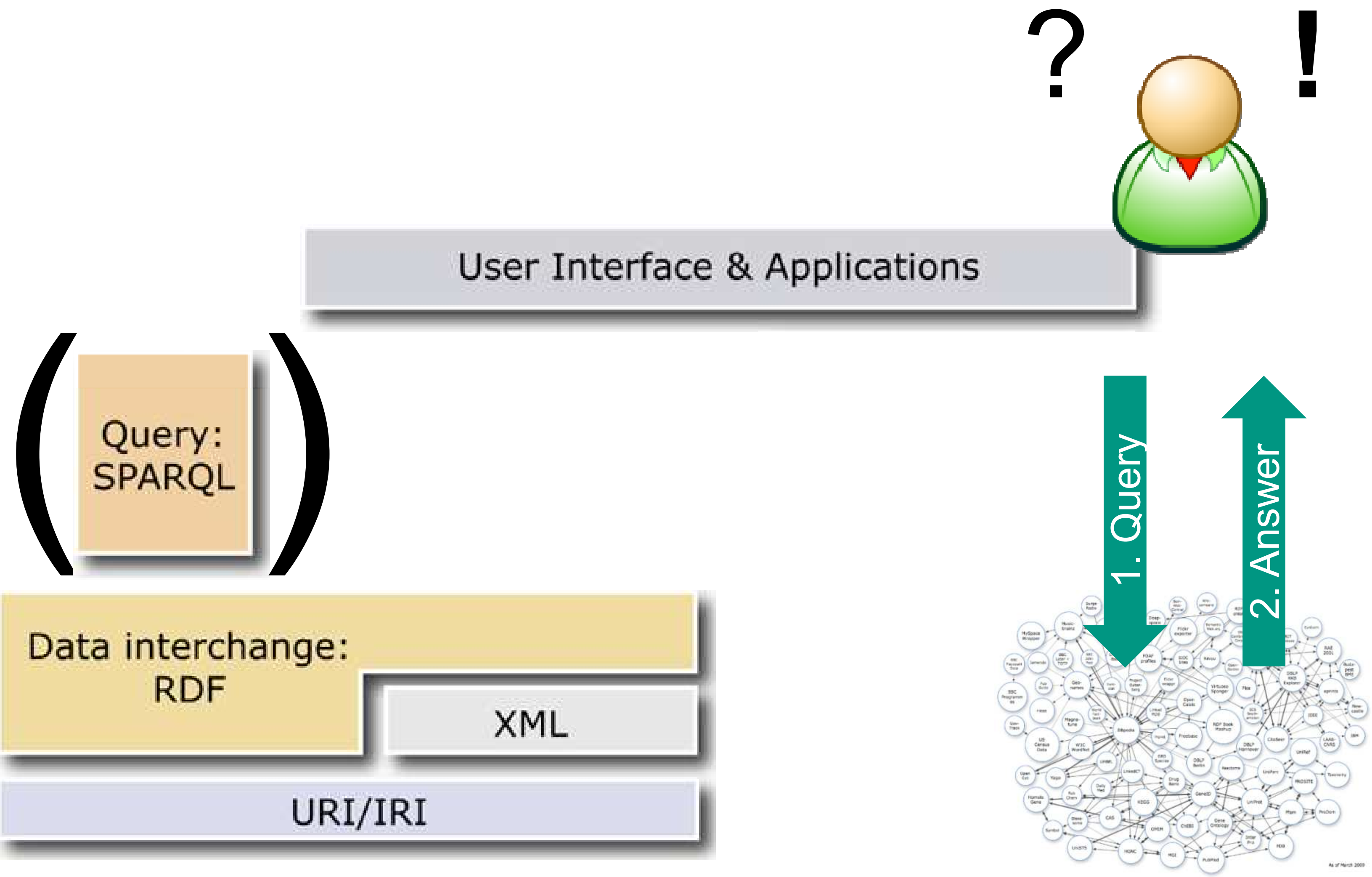
File published at http://harth.org/andreas/foaf.rdf

URI denoting Andreas: http://harth.org/andreas/foaf.rdf#ah

Semantic Web Application Architecture



Linked Data Application: Minimal Architecture



User Interface & Applications

(Query: SPARQL)

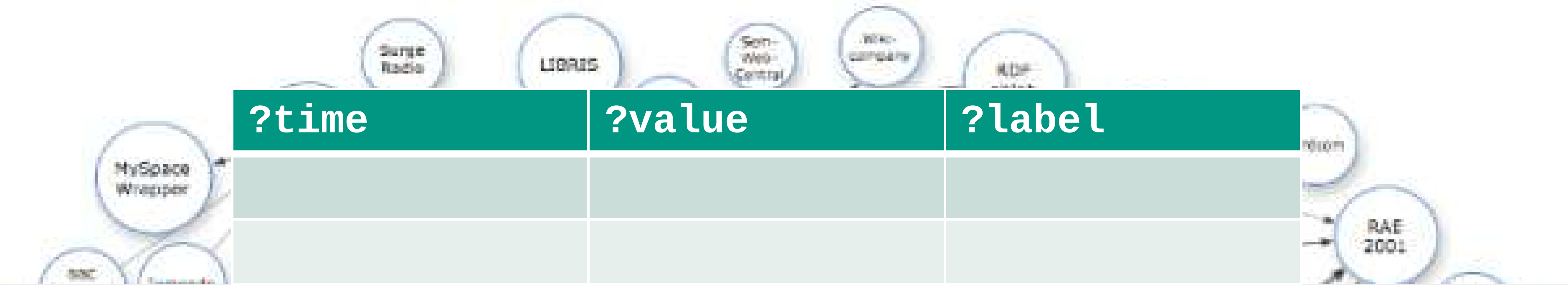
Data interchange: RDF

XML

URI/IRI

Queries over Linked Data

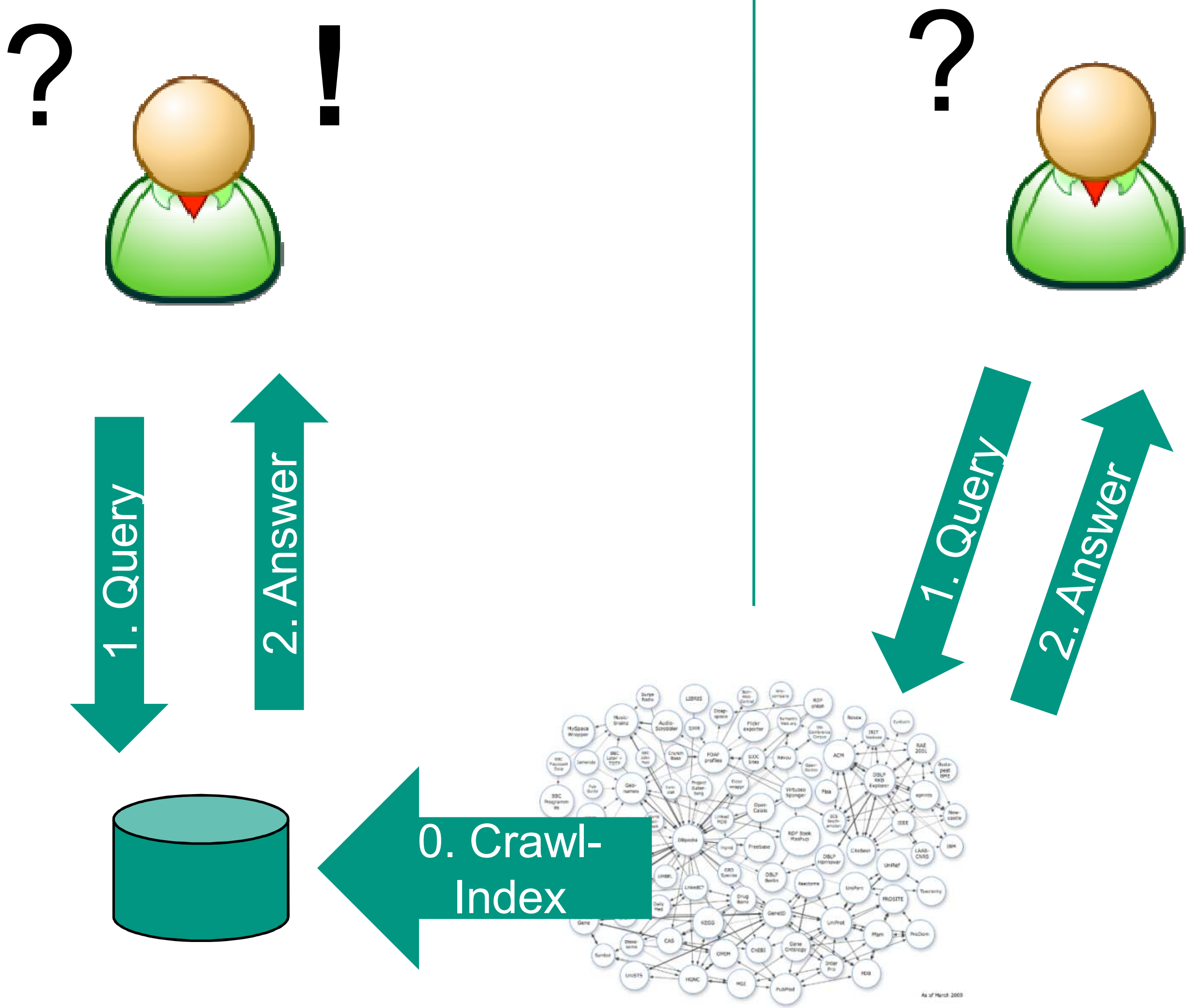
```
SELECT ?time ?value ?label
WHERE {
 ?s qb:dataset
    <http://gesis-lod.appspot.com/data?code=14111#ds> .
 ?s dcterms:date ?time .
 ?s gesis:partei ?partei .
 ?partei rdfs:label ?label .
 ?s sdmx-measure:obsValue ?value .
}
```



?time	?value	?label

Warehousing/Crawl-Index-Serve

On-Demand



?

!

1. Query

2. Answer

0. Crawl-Index

?

!

1. Query

2. Answer

Example: Visualising Election Results

- Data from IT.NRW (statistics office of Northrhine Westphalia) in CSV
- Step 1: convert to RDF (via Google App Engine wrapper)
- Step 2: query Linked Data
- Step 3: visualise results

http://gesis-lod.appspot.com/vis/

Example: Visualising Economic Situation

- Data from GESIS (German archive for social sciences)
- Step 1: convert to RDF (static file) and publish online
- Step 2: query Linked Data
- Step 3: visualise results

http://gesis-lod.appspot.com/vis/

Example: Visualising Eurostat

- Data from Eurostat (statistics office of the European Union) avialable as CSV and SDMX
- Step 1: convert to RDF (via Google App Engine wrapper)
- Step 2: query Linked Data
- Step 3: visualise results

http://estatwrap.ontologycentral.com/page/tsieb010

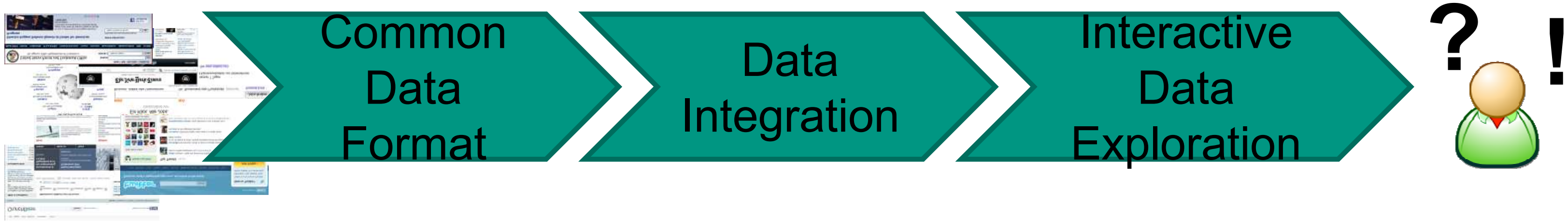
Linked Data Services

- There are data sources which provide only selective access to their data (e.g., APIs of social networking sites)
- Sometimes more than one parameter is required (e.g., calculating the shortest route between two points)

- We’d like to leverage Linked Data technology for integrating those services

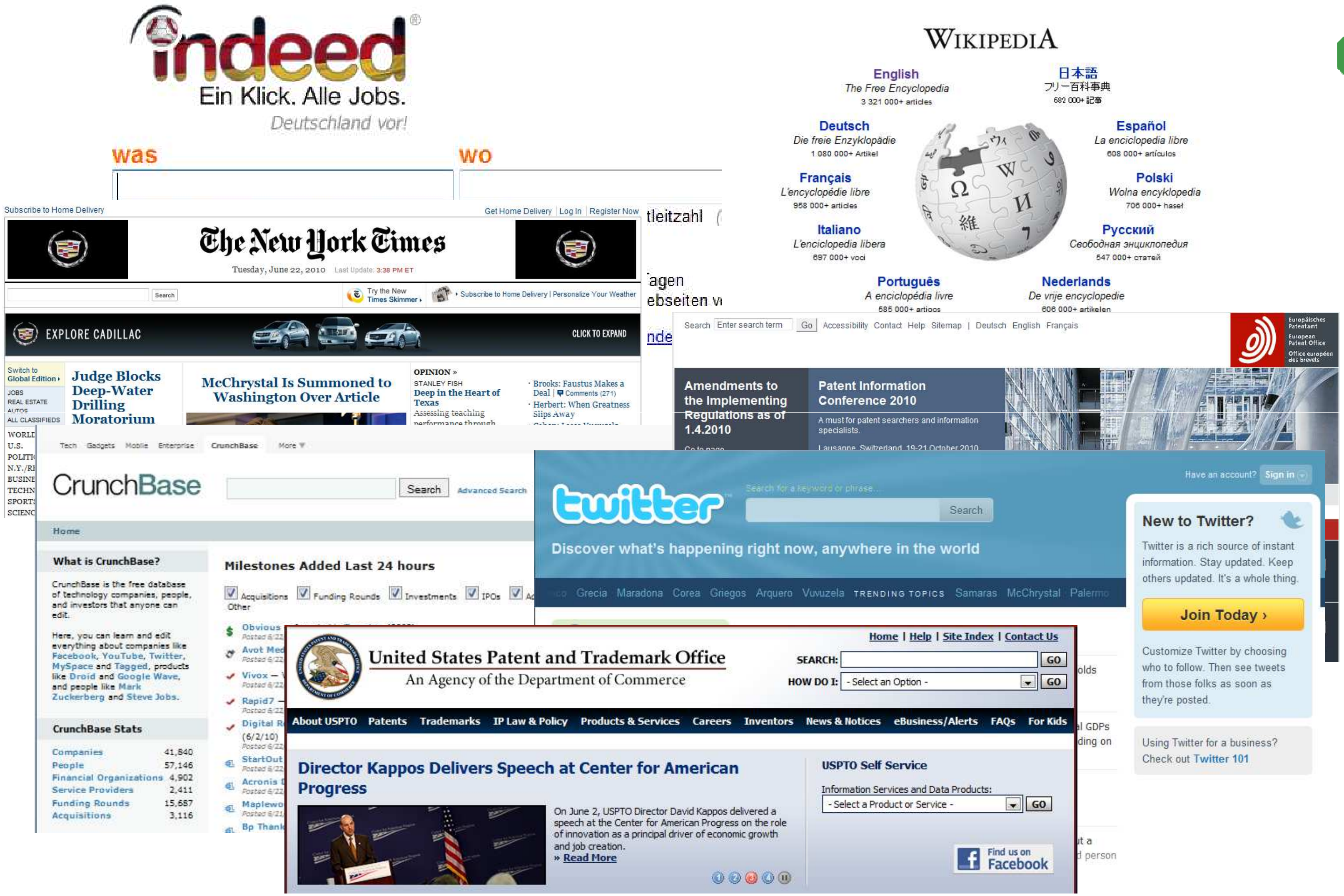
Scenario

- Typical data integration scenario

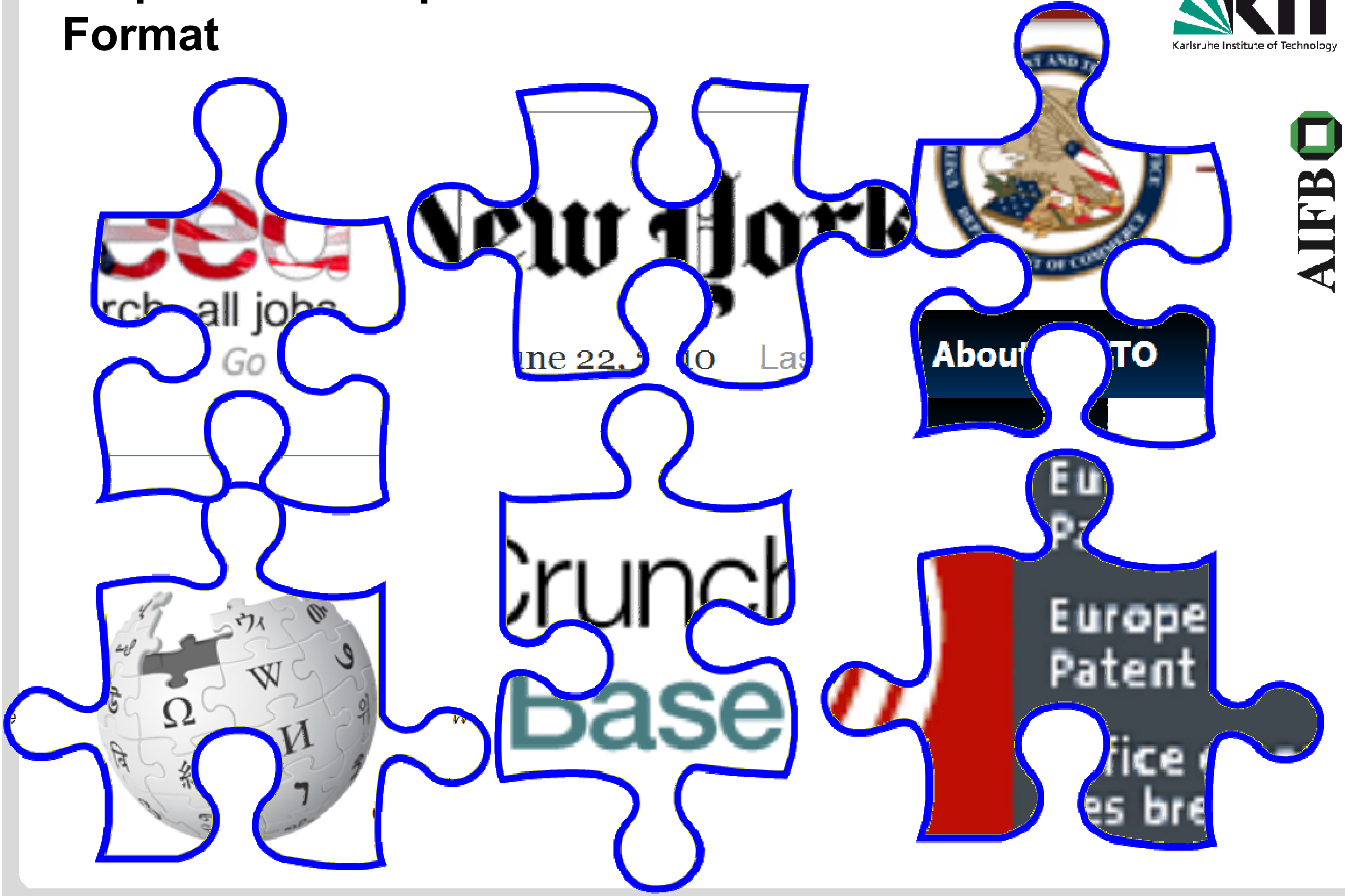


- Q: job offerings of competitors of Facebook?
- Q: funding pattern of Vulcan Capital?

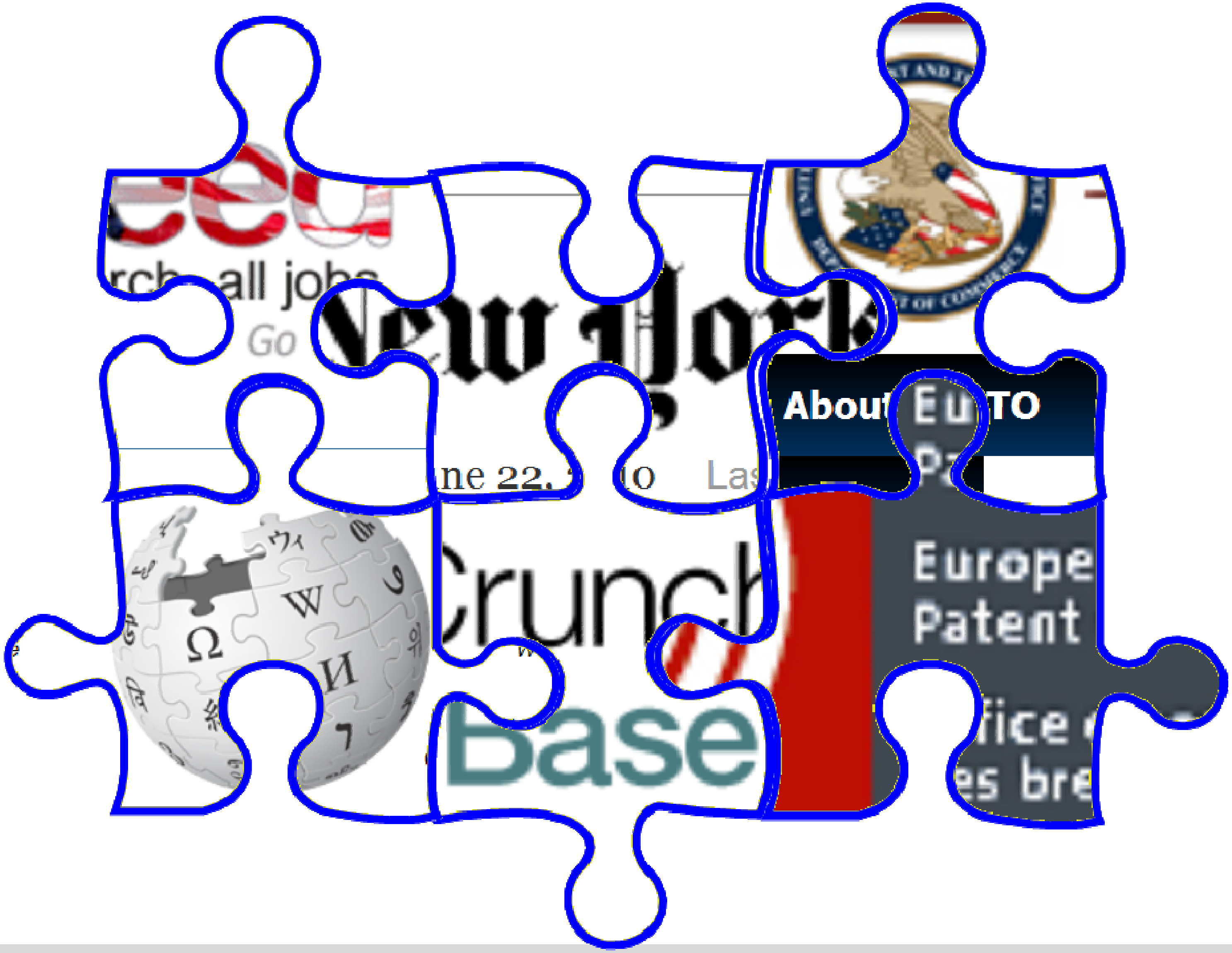
Data Sources



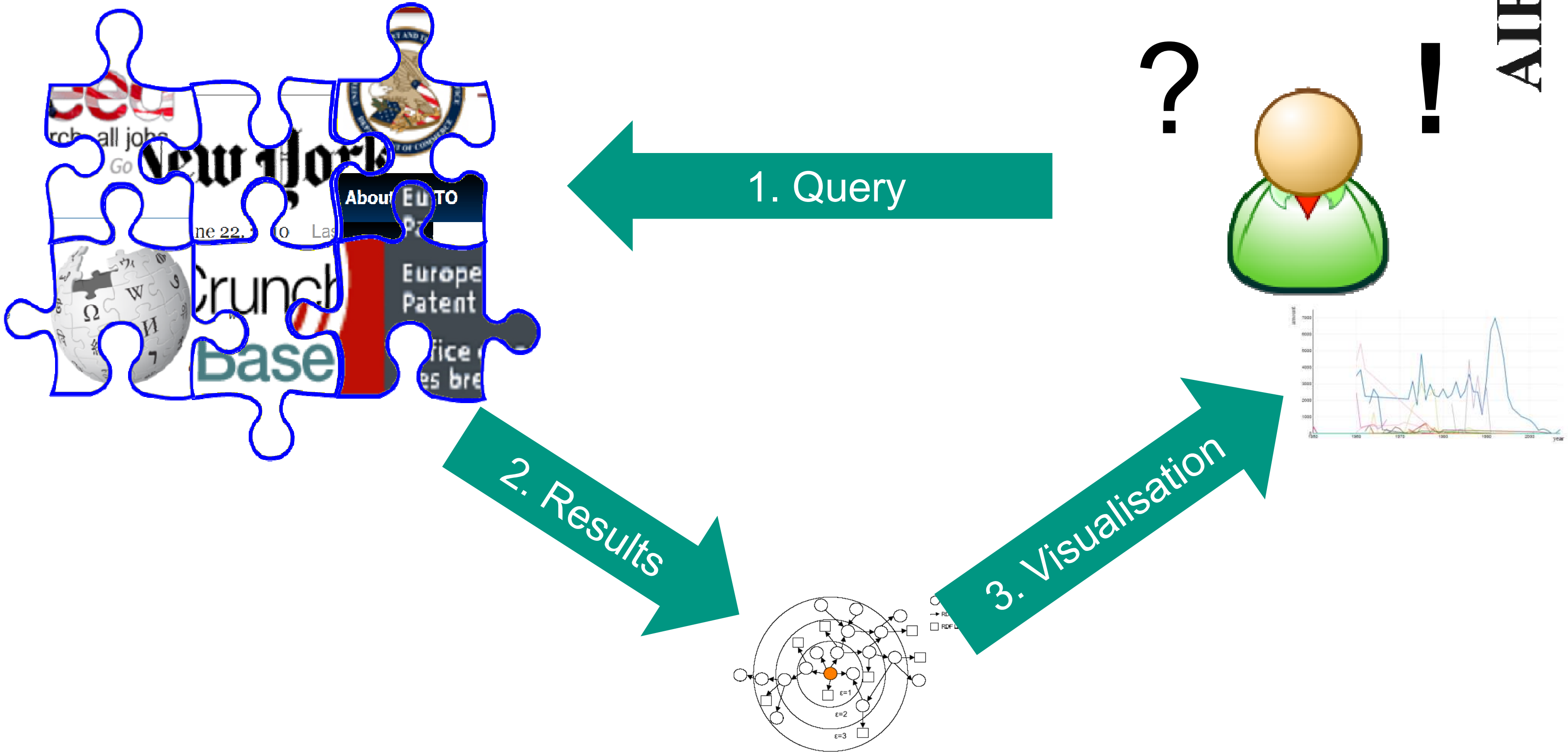
Step 1: Data Preparation – Common Data Format



Step 2: Data Integration



Step 3: Interactive Data Exploration

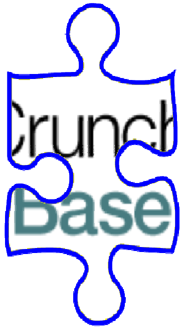


1. Query

2. Results

3. Visualisation

Interlinking Data with Data from Services?



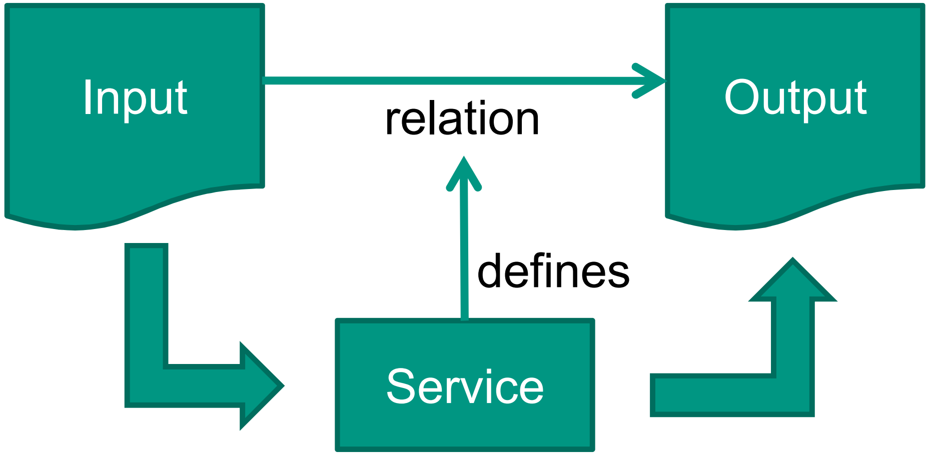
```
:facebook foaf:name “Facebook” .
:facebook cb:has_office #facebook-hq .
:facebook-hq geo:lat “37.416” .
:facebook-hq geo:long “122.152” .
:facebook-hq vc:locality “Palo Alto, CA” .
```

Given company name and location, return job openings

Given lat/lon, return nearby places (via GeoNames)

Data Services

- Given input, provide output
- Input and output are related in a service-specific way
- Do not change the state of the world



- E.g. GeoNames findNearbyWikipedia service
  - Input: lat/lon
  - Output: places
  - Relation: output places that are *nearby* input place

Enter LIDS: Linked Data Services

- We’d like to integrate data services with Linked Data

1. LIDS need to adhere to Linked Data principles

- We’d like to use data services in software programs

2. LIDS need machine-readable descriptions of input and output

1. Data Services as Linked Data

- Input is given as URI

Service Endpoint

```
http://geowrap.openlids.org/findNearbyWikipedia
?lat=37.416&lng=-122.152
#point
```

Parameters

Input Identifier

- Resolving the URI yields RDF:

Relation

Input

Output

```
@prefix dbp: <http://dbpedia.org/resource/> .
@prefix : <http://geo..Wiki?lat=37.416&lng=-122.152#>
:point
      foaf:based_near dbp:Palo_Alto%2C_California ;
      foaf:based_near dbp:Packard%27s_garage .
```

2. LIDS Descriptions using SPARQL

- Given specific input, corresponding output can be retrieved from implicit data source. Corresponds to SPARQL Construct Query

```
CONSTRUCT { [output] } FROM [endpoint]
      WHERE { [input]  }
```

- Input describes needed data as a basic graph pattern
- Endpoint is the base URI for constructing a service input
- Output describes data that is delivered by service, using unsafe variables (more about that in the TR)

```
CONSTRUCT { ?point foaf:based_near ?feature. }
  FROM <http://geowrap.openlids.org/findNearbyWikipedia>
  WHERE { ?point a Point . ?point geo:lat ?lat .
                           ?point geo:long ?lng }
```

Linked Data Services Summary

- Dynamic sources (GeoNames Wrapper, Twitter Wrapper, Feeds Wrapper) can be integrated into Linked Data Web
- LIDS useful for
  - Inserting links to LIDS into static RDF data sets
  - Linked Data endpoints that dynamically add links from their data to LIDS
  - LD browsers that augment retrieved data with data retrieved from LIDS
  - Integrating LIDS into SPARQL query processing

- LIDS provide means for publishing and reusing data services on the web

Demo-Application

- Job openings at competitors of Facebook
- Funding patterns of Vulcan Capital

Conclusion

- Amount of available data keeps growing
- Need semantics for the ability to integrate data from multiple sources
- Possible to query and visualise datasets in combination

- Processing and quering data from multiple sources increases transparency and facilitates research as hypothesis testing becomes easy

Acknowledgements

- Slides from
  - Michael Hausenblas, DERI
  - Denny Vrandecic, AIFB, KIT
- Joint work with
  - Aidan Hogan, DERI
  - Juergen Umbrich, DERI
  - Sebastian Speiser, AIFB, KIT
  - Marcel Karnstedt, DERI
  - Katja Hose, MPI
  - Robert Isele, FU Berlin
  - Kai-Uwe Sattler, TU Illmenau
  - Axel Polleres, DERI
  - Stefan Decker, DERI
  - Benjamin Zapilko, GESIS