

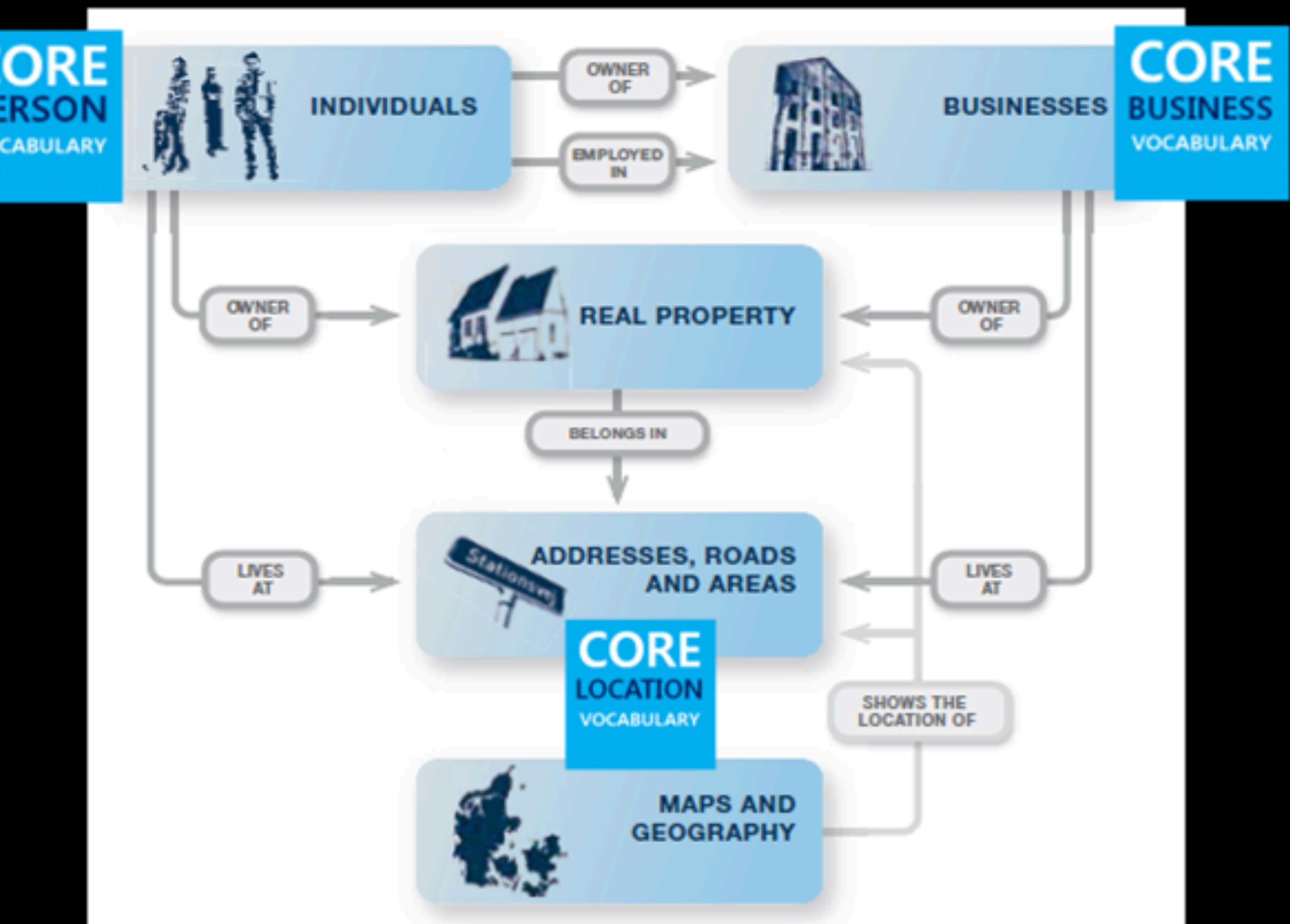
MARCO BRANDIZI 4/6/2017, #ODFEST2017

LINKED DATA PRACTICALITIES

FIND ME AT: [HTTPS://TINYURL.COM/SOD17LODPIPE](https://tinyurl.com/sod17lodpipe)

OVERVIEW

- We have learned about ISA CORE Vocabularies (i.e., about RDF, OWL/Schema)
- We have seen how to use them to represent our data



CREDITS: [HTTPS://GOO.GL/YDGEBO](https://goo.gl/YDGEBO)

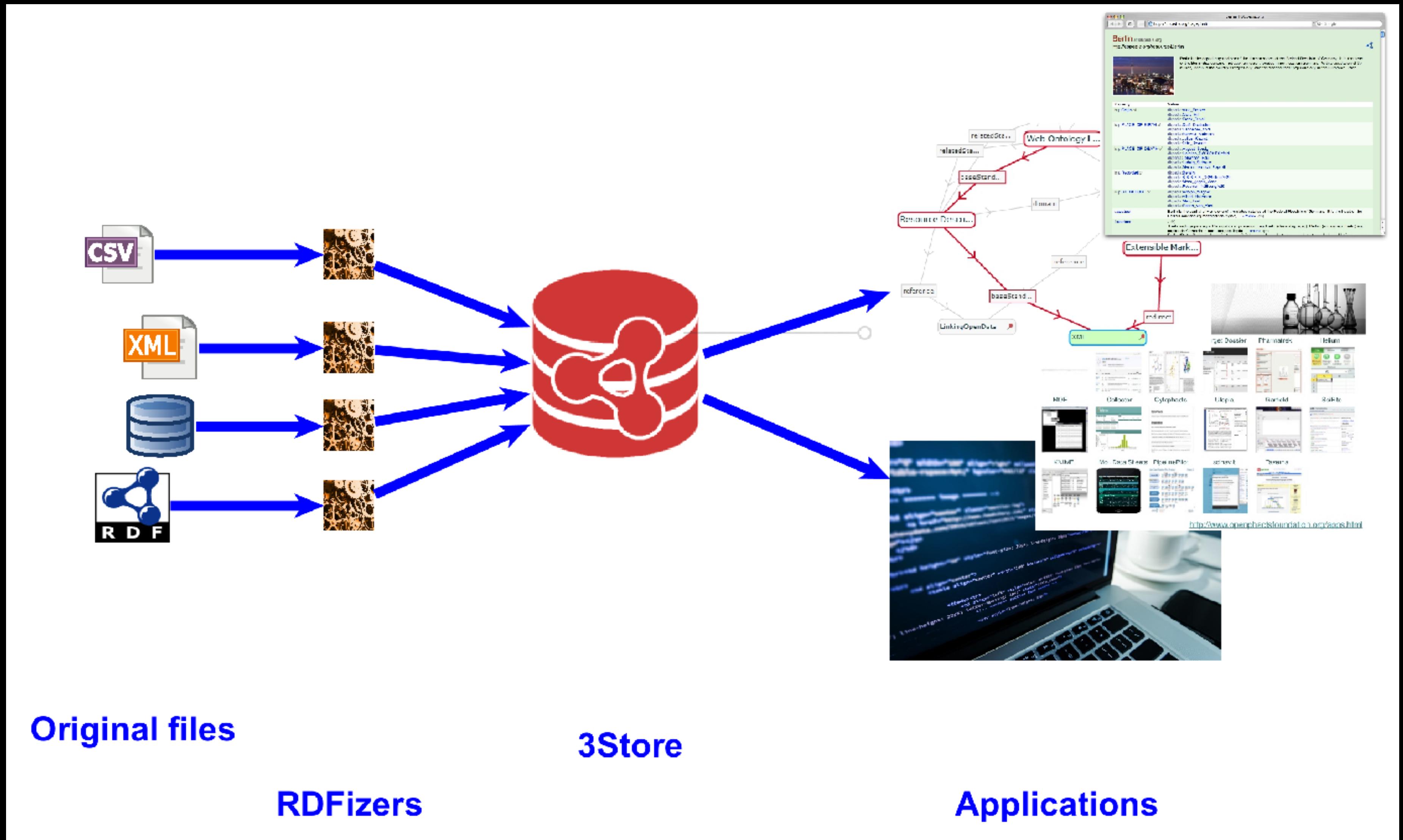


NOW, WHAT?

CREDITS: [HTTPS://GOO.GL/W9VMKY](https://goo.gl/W9VMKY)

LINKED DATA

PRODUCER TO CONSUMER



OK, LET'S SEE AN EXAMPLE

- Link to demo_data
- Hospital lists in 3 Italian regions
- 2xCSV, 1xXML
- Different column names, formats
- Slightly different contents
 - so, not the worst around



CREDITS: [HTTPS://GOO.GL/9ZDVMT](https://goo.gl/9ZDVMT)

ONE* TRIPLE TO RULE THEM ALL!

* or 1E6, 1E9...



FOR THE NOT-SO-GEEK: OPEN REFINER + GREFINE

- <http://refine.deri.ie/>
- See [demo_data/](#)
[hospitals_tuscany.openrefine.tar.gz](#)

RDF Refine

Home Showcases

RDF Schema Alignment

The RDF schema alignment skeleton below specifies how the RDF data that will get generated from your grid-shaped data. The cells in each record of your data will get placed into nodes within the skeleton. Configure the skeleton by specifying which column to substitute into which node.

Base URI: <http://data.fingal.ie/councilor/> [edit](#)

RDF Skeleton [RDF Preview](#)

Available Prefixes: [dr](#) [rdfs](#) [foaf](#) [rdf](#) [void](#) [+ add prefix](#) [* manage prefixes](#)

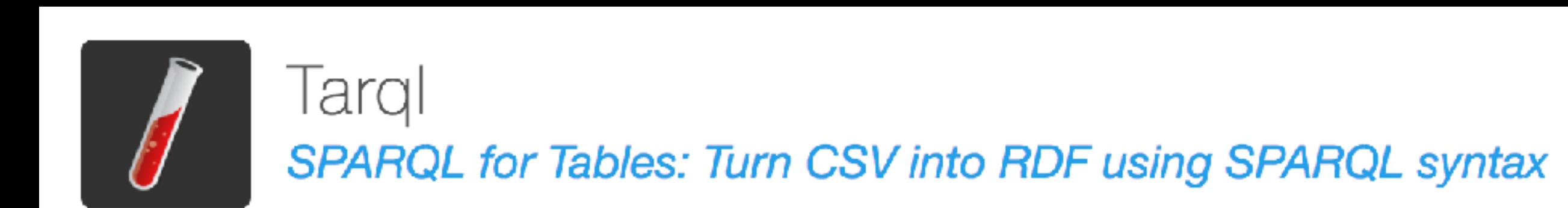
name URI	Value	Target Node
x:toaf:Person	>-foaf:name→	name cell
x:Councilor	>-party→	party URI
	add rdf:type	Party
		add rdfs:type
	>-foaf:mbox→	email cell
	>-foaf:phone→	tel cell
	>-foaf:depiction→	image URI
		add rdf:type
	>-councilor:O1→	area URI
		CouncilorDistrict
		add rdfs:type
		add property
	>-:currentCouncilor→	current member cell

Add another root node

OK Cancel

FOR THE SEMANTIC WEB LOVER: TARQL

- Link
- See demo_data/hospitals_tn.tarql



Tarql: SPARQL for Tables

Tarql is a command-line tool for converting CSV files to RDF using SPARQL 1.1 syntax. It's written in Java.

Introduction

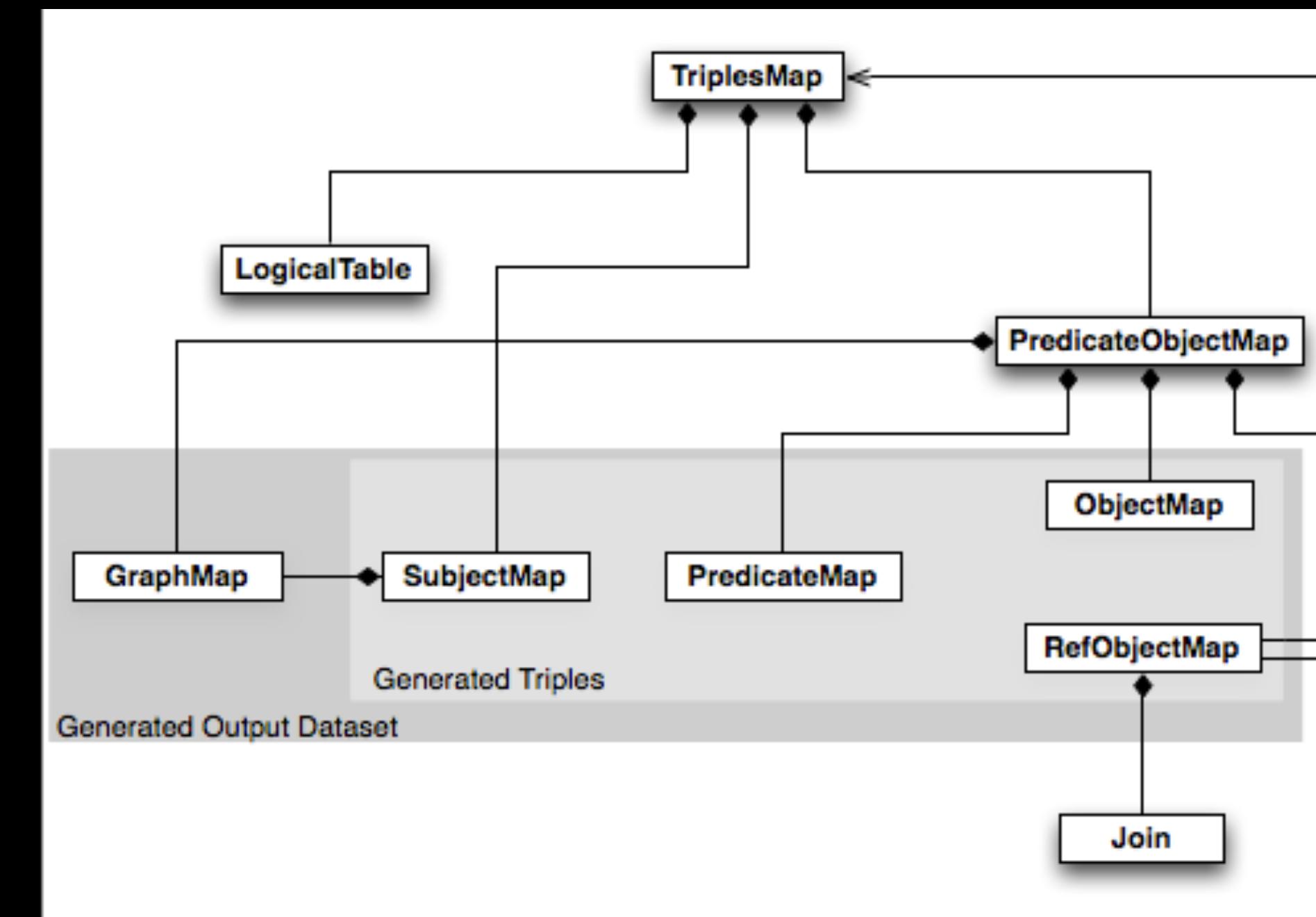
In Tarql, the following SPARQL query:

```
CONSTRUCT { ... }
FROM <file:table.csv>
WHERE {
  ...
}
```

is equivalent to executing the following over an empty graph:

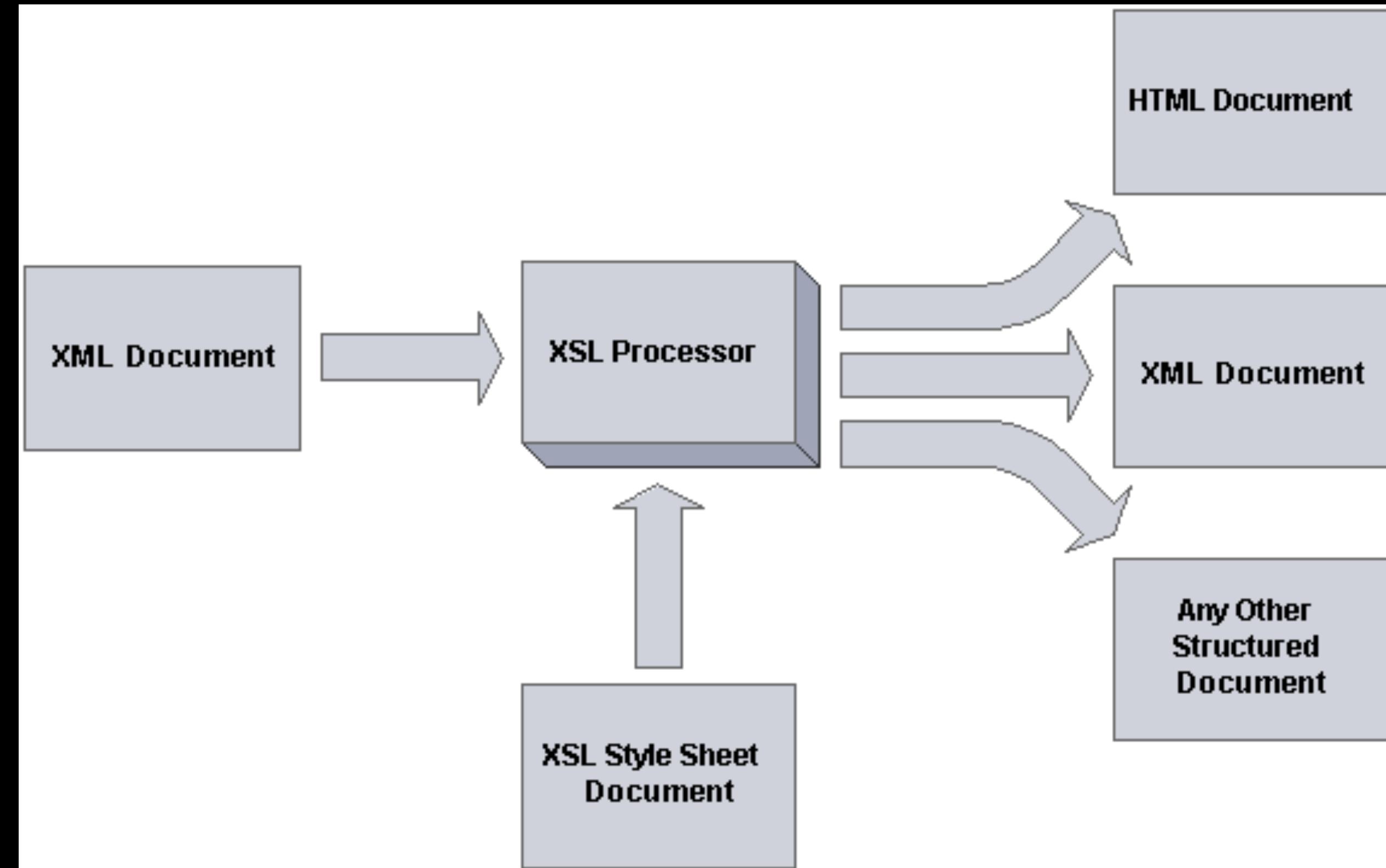
FOR THE TRADITIONALIST: RELATIONAL/RDF MAPPING

- Eg, R2ML
- <https://www.w3.org/TR/r2rml/#example-input-database>



FOR THE POOR DEVIL: XSL

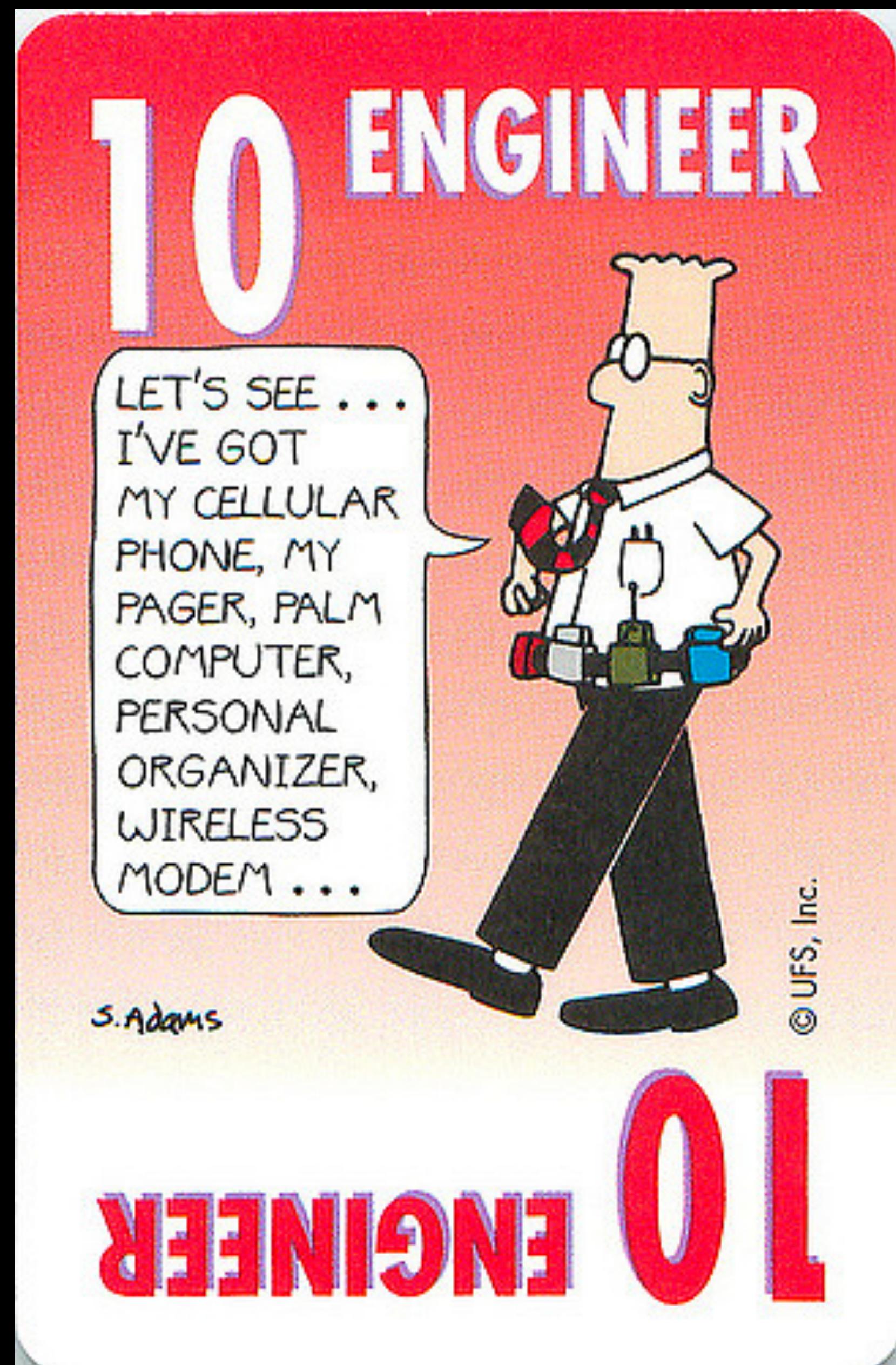
- See `demo_data/hospitals_lombardy.xsl`



CREDITS: [HTTPS://GOO.GL/Q9CZRY](https://goo.gl/Q9CZRY)

FOR THE TRUE HACKER: RDF FRAMEWORKS

- Python: rdflib
 - writing ex: <https://goo.gl/oF6KvK>
 - reading ex: <https://goo.gl/FbWgXW>
- Java: Jena
 - SPARQL ex, http://jena.apache.org/documentation/query/app_api.html
- Not enough yet? OWL-API, Tawny-OWL

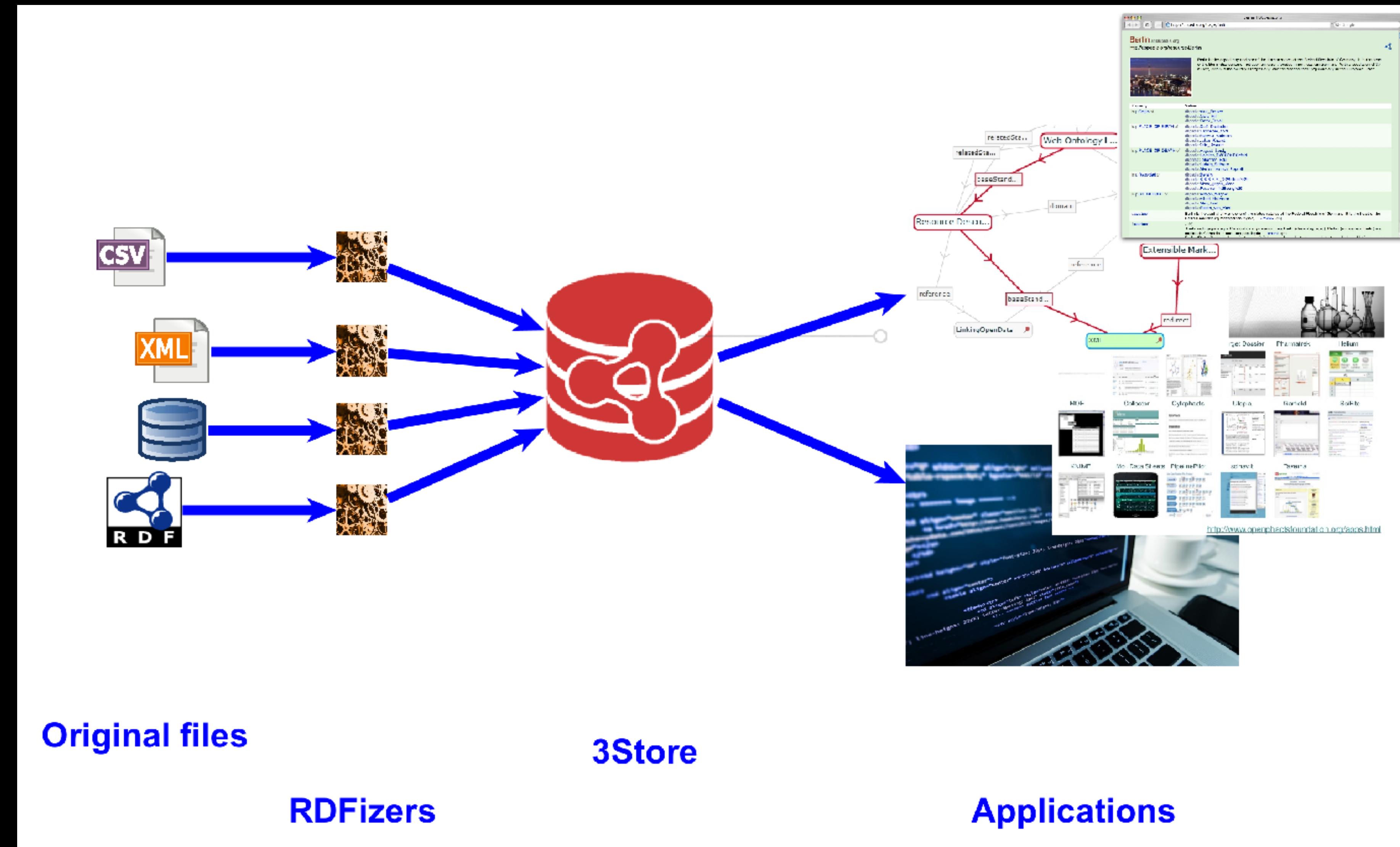


A photograph of Santa Claus from the chest up. He is wearing his traditional red suit with white fur trim on the cuffs and collar, and a black belt with a silver buckle. He has a large, bushy white beard and is looking slightly to his left with a thoughtful expression. In his left hand, he holds a dark blue glass bottle with a yellow label. His right hand rests on a vertical gold-colored pole or doorframe. The background is dark and out of focus.

SELF-PROMOTION TIME:

<http://github.com/EBIBioSamples/java2rdf>

NOW WHAT? PUBLISHING RDF



WHO ARE YOU? WHAT ABOUT YOU?

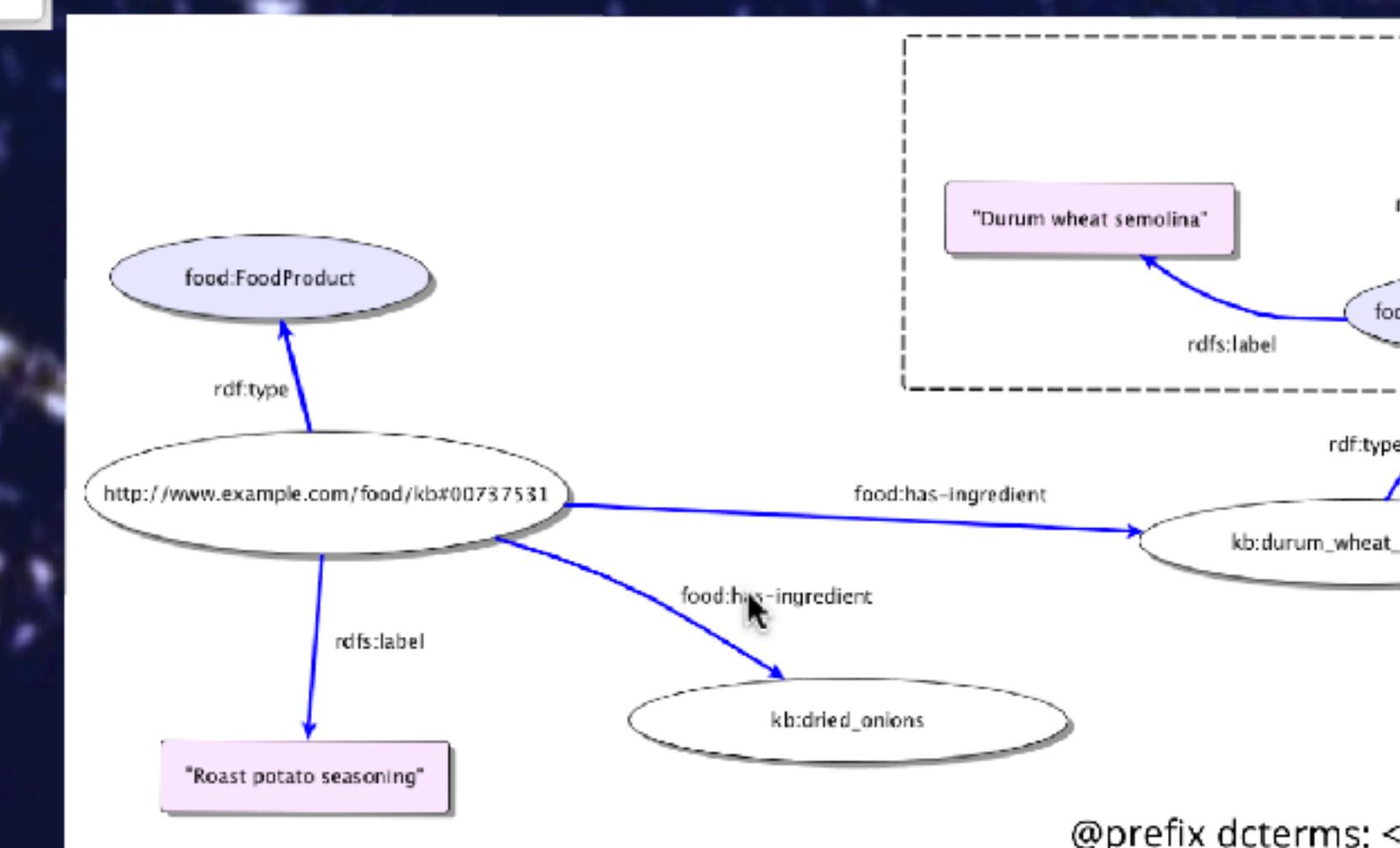
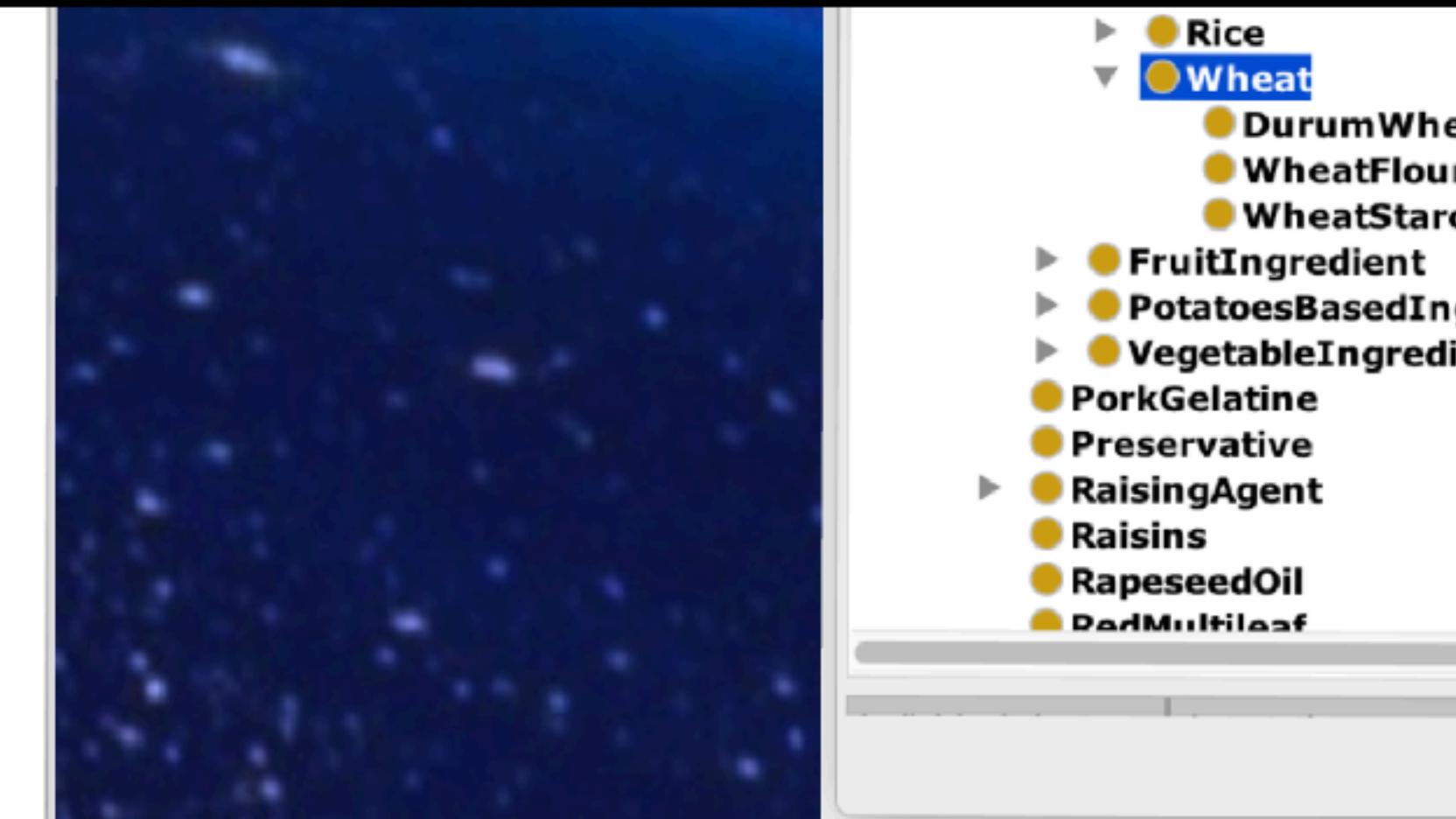
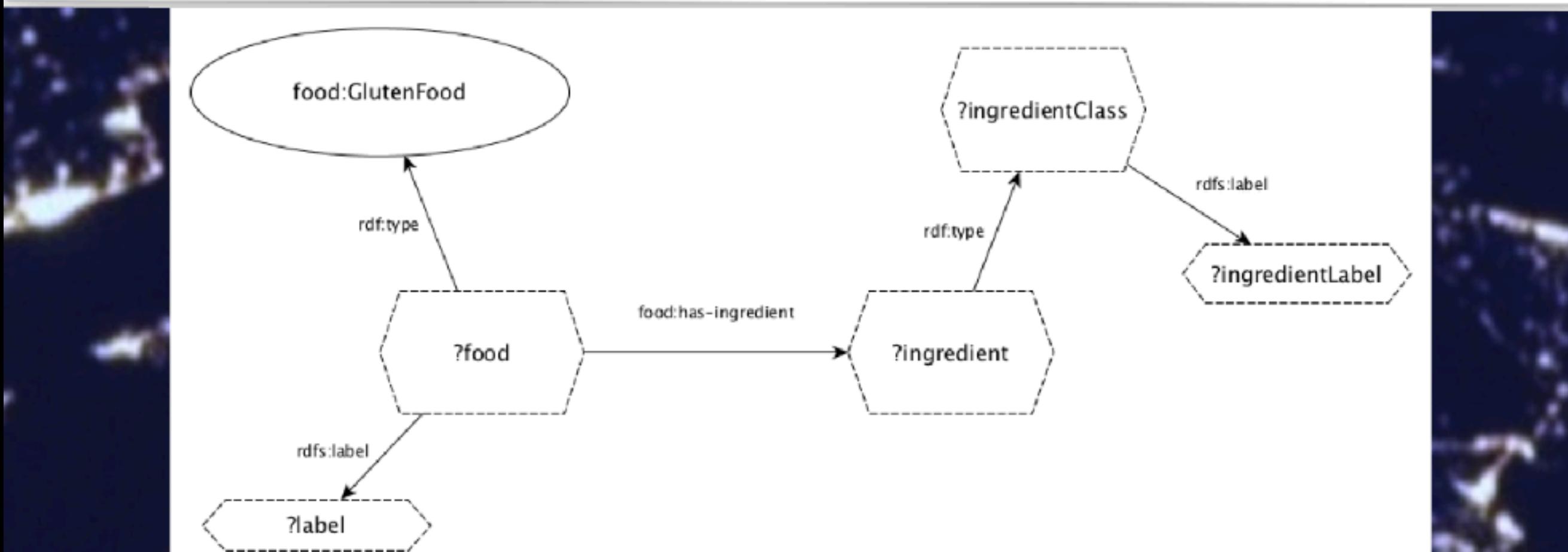
URIS

Example from demo_data/:

```
python get_dbpuri_demo.py "http://dbpedia.org/resource/Trentino"
```

SPARQL (AND SERVICE/ENDPOINTS)

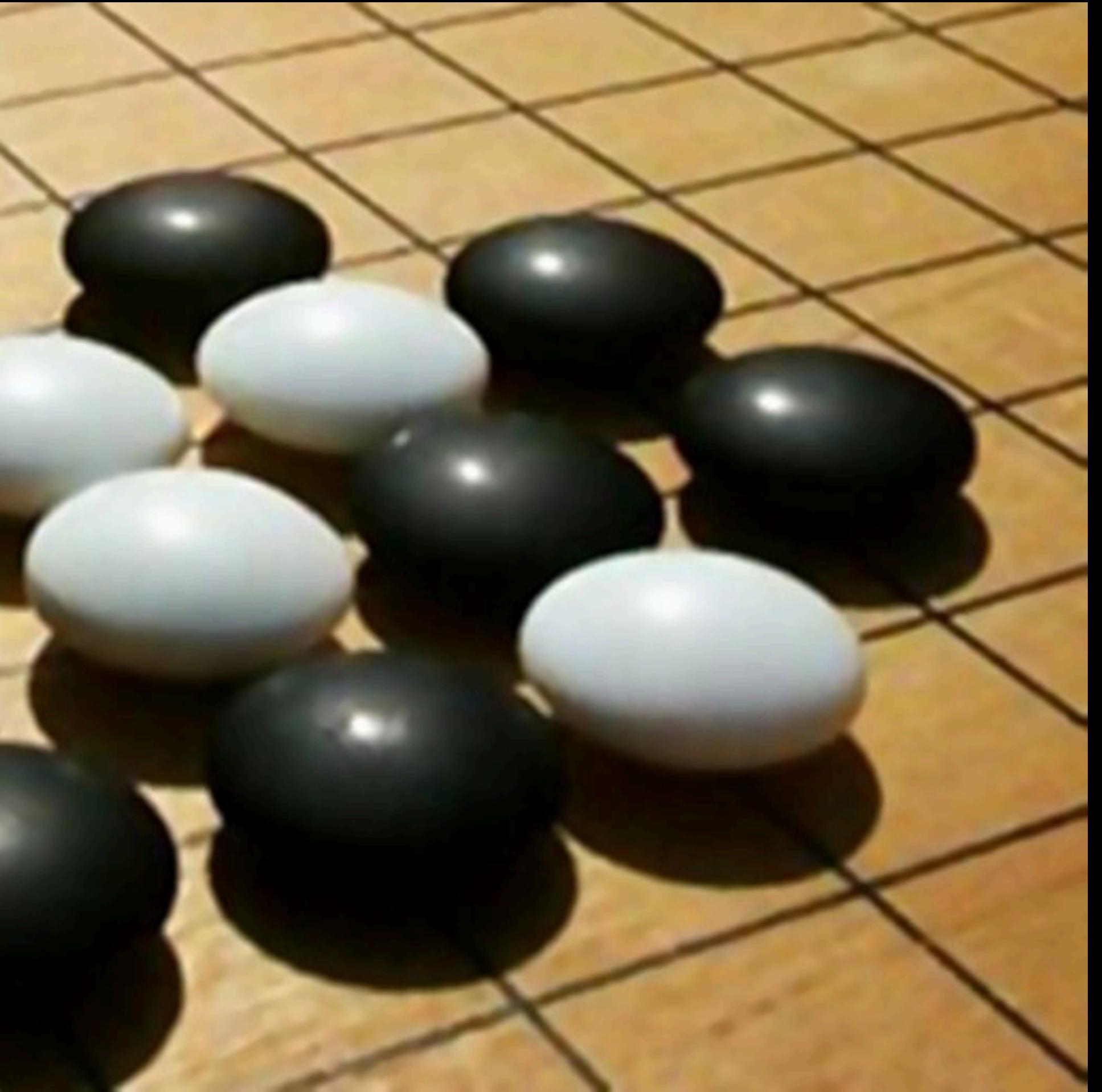
```
5
6 select distinct ?code ?label ?ingredientLabel
7 where
8 {
9   ?food rdf:type food:GlutenFood;
10    rdfs:label ?label;
11    dcterms:identifier ?code.
12
13   ?food food:has-ingredient ?ingredient.
14   ?ingredient a ?ingredientClass.
15   ?ingredientClass rdfs:label ?ingredientLabel.
16   FILTER ( REGEX ( ?ingredientLabel, 'durum', 'i' ) )
17 }
18 ORDER BY ?label
```



```
15   ?ingredientClass rdfs:label ?ingredientLabel.
16   FILTER ( REGEX ( ?ingredientLabel, 'durum', 'i' ) )
17 }
```

OK, LET'S PLAY WITH FUSEKI

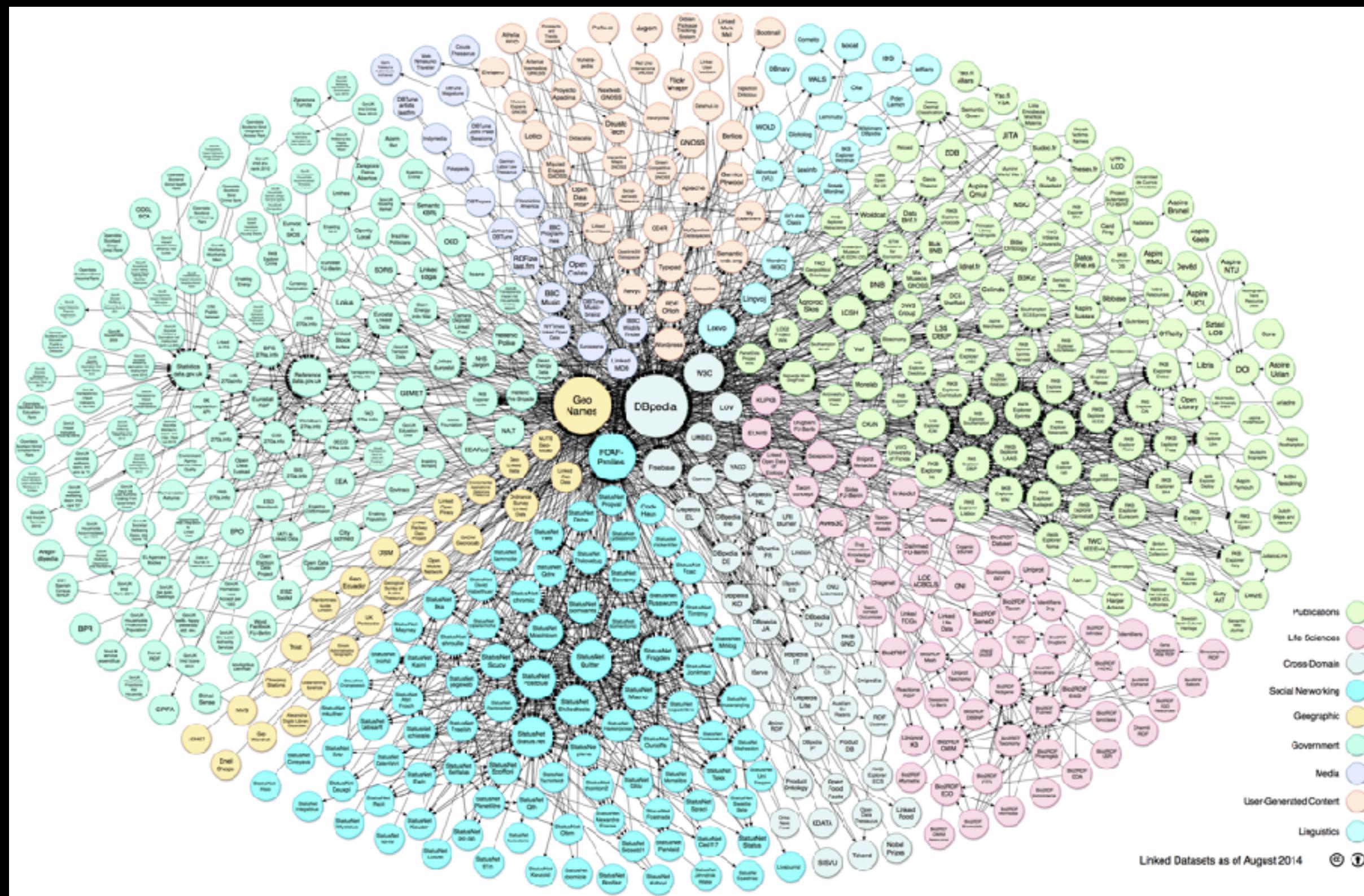
- Importing RDF files with `tdbloader` (including ontologies)
- Running Fuseki and playing with the end point (see `demo_data/sparql`)
- **Virtuoso** works pretty much the same



CREDITS: WIKIPEDIA

MORE LD PUBLISHING APPROACHES

- Linked Data Meets APIs
 - JSON-LD
 - Hydra
- RDF Dumps
- Are SPARQL endpoints a myth?
 - <https://goo.gl/l5mrJH>
 - <https://blog.muninn-project.org/node/68>



NOW WHAT? CONSUMING LD

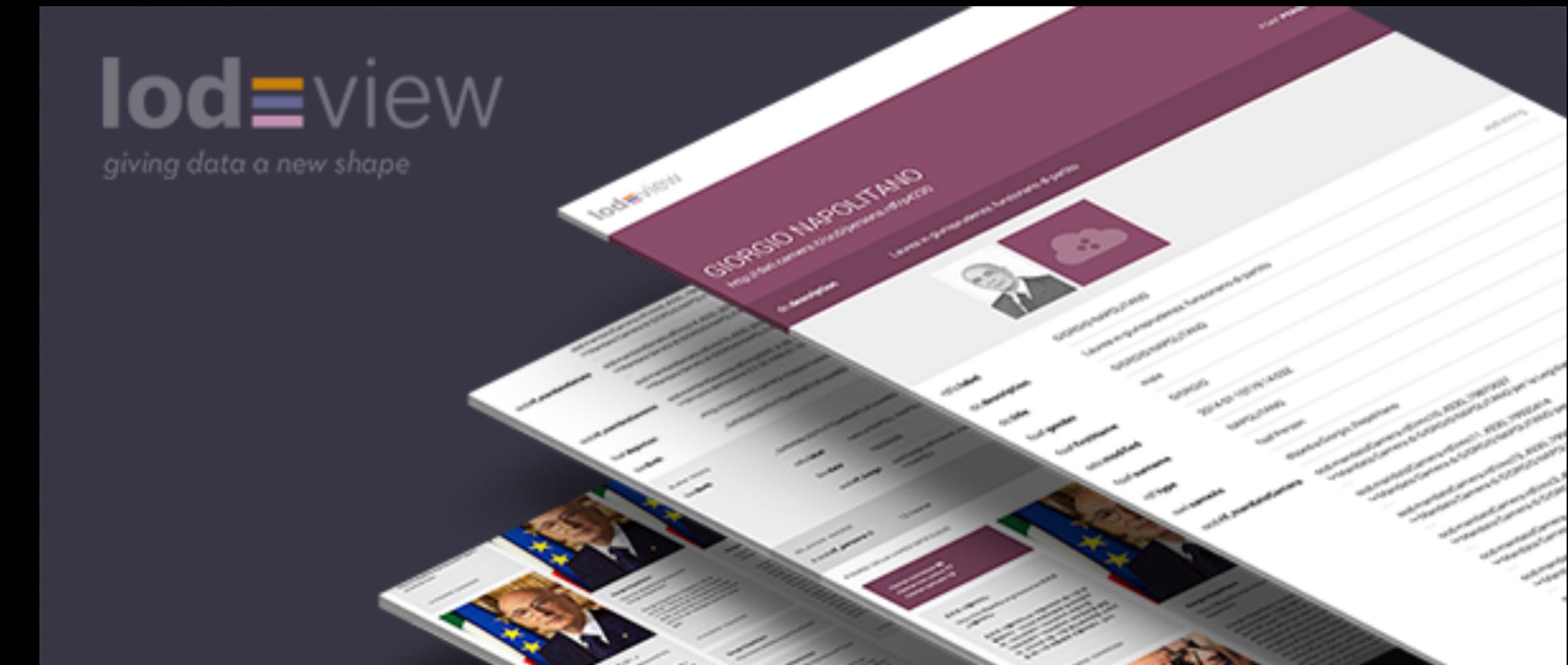
(JUST A FEW EXAMPLES)



LD BROWSERS

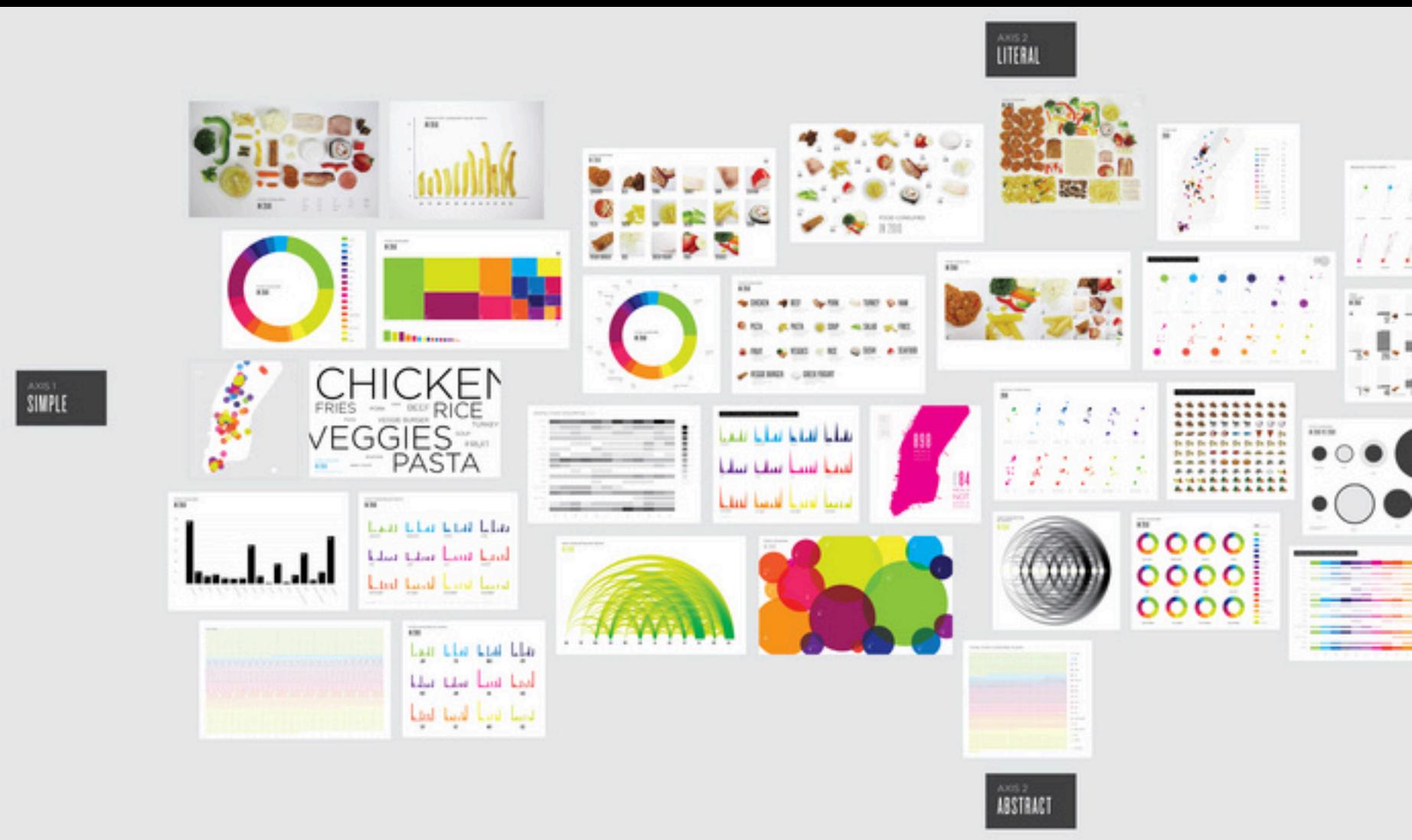
JUST A FEW AMONG MANY

- LODEStar
 - (<https://github.com/EBISPORT/lodestar>)
 - let's see a demo
- LODView (<http://lodview.it/>)



DATA ANALYSIS AND GRAPHICAL VISUALISATION

- eg, Silk
 - <http://alice-corona.nl/pages/events/odfest2017>
 - Many, many thanks to Alice!
- eg, D3SPARQL, <https://github.com/ktym/d3sparql>



[HTTPS://GOO.GL/3QWNBF](https://goo.gl/3QWNBF)

DATA MASHUPS

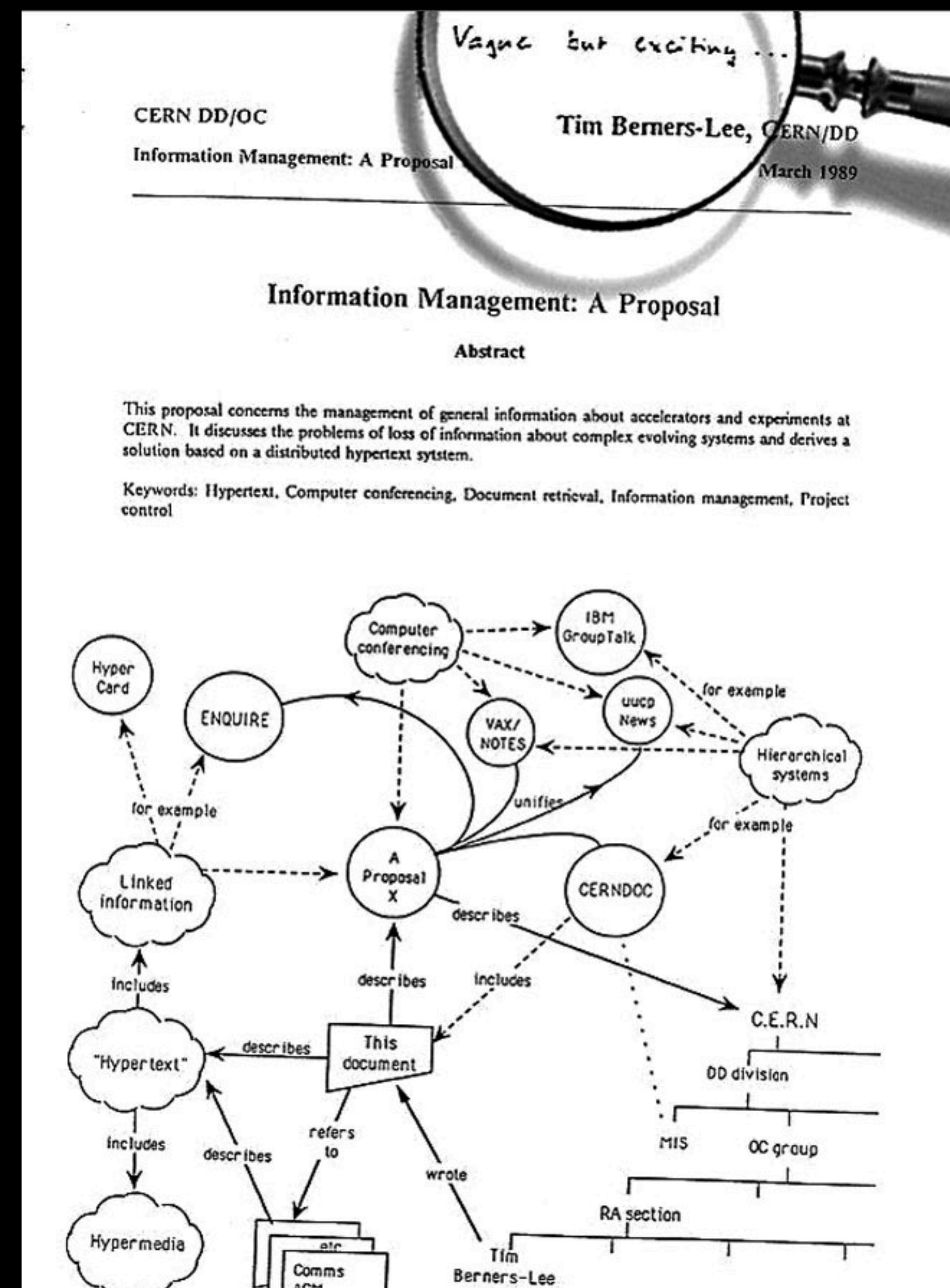
- eg, <http://www.simile-widgets.org/exhibit/>



[HTTPS://GOO.GL/KGXXY4](https://goo.gl/kgXXY4)

INTEGRATING WITH TRADITIONAL WEB

- Semantic CMSs and alike:
 - <https://goo.gl/TrvoSQ>



CREDITS: [HTTPS://GOO.GL/N27GVW](https://GOO.GL/N27GVW)

BIG PLAYERS JUMPS IN

- <http://schema.org/>
- SEO & schema.org:
 - <https://goo.gl/aRZKL>
 - <https://goo.gl/wGJ62i>
- practical examples:
 - <https://goo.gl/HRA4Lf>
 - <https://goo.gl/BHWJkO>



HTTPS://GOO.GL/ZTAOEN

THAT'S ALL!

THANKS!

- Marco Brandizi
 - name.surname@gmail.com
(or LinkedIn, Facebook, Twitter...)
- Find this slides at:
 - <https://tinyurl.com/sod17lodpipe>



CREDITS: WIKIPEDIA