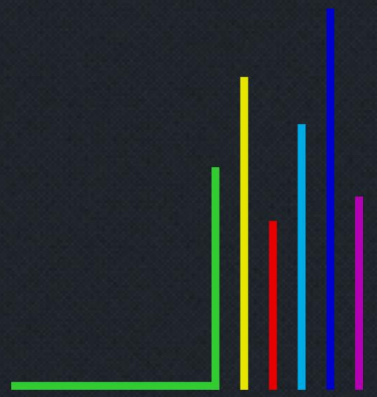


Summer School of Linked Data in Architecture and Construction

Coding with Linked Data

- **A brief intro to the RDFLib and Jena Frameworks**

Jakob Beetz, RWTH Aachen University
Georg Schneider, Fraunhofer IBP
Rui de Klerk, University of Lisbon

design computation

RWTH AACHEN UNIVERSITY

Coding with Linked Data, market overview

Programming APIs for Linked Data and the Semantic Web

- **Java:**
  - Jena
  - RDF4J
  - commons-rdf
  - owlapi - The OWL API is a Java API for creating, manipulating and serialising OWL Ontologies.

- **Python**
  - RDFlib - Pythonic RDF API.
  - SPARQLWrapper - A wrapper for a remote SPARQL endpoint.

C

librdf - Redland librdf RDF API and triple stores.
raptor - Redland Raptor RDF syntax library.

JavaScript

RDFJS - Github Organization that maintains modern JavaScript RDF libraries based on open, maintained standards

C#

dotNetRDF
RDFSharp

For an extensive list see Awesome-Semantic-Web[1]:

[1] https://github.com/uscholdm/awesome-semantic-web#programming

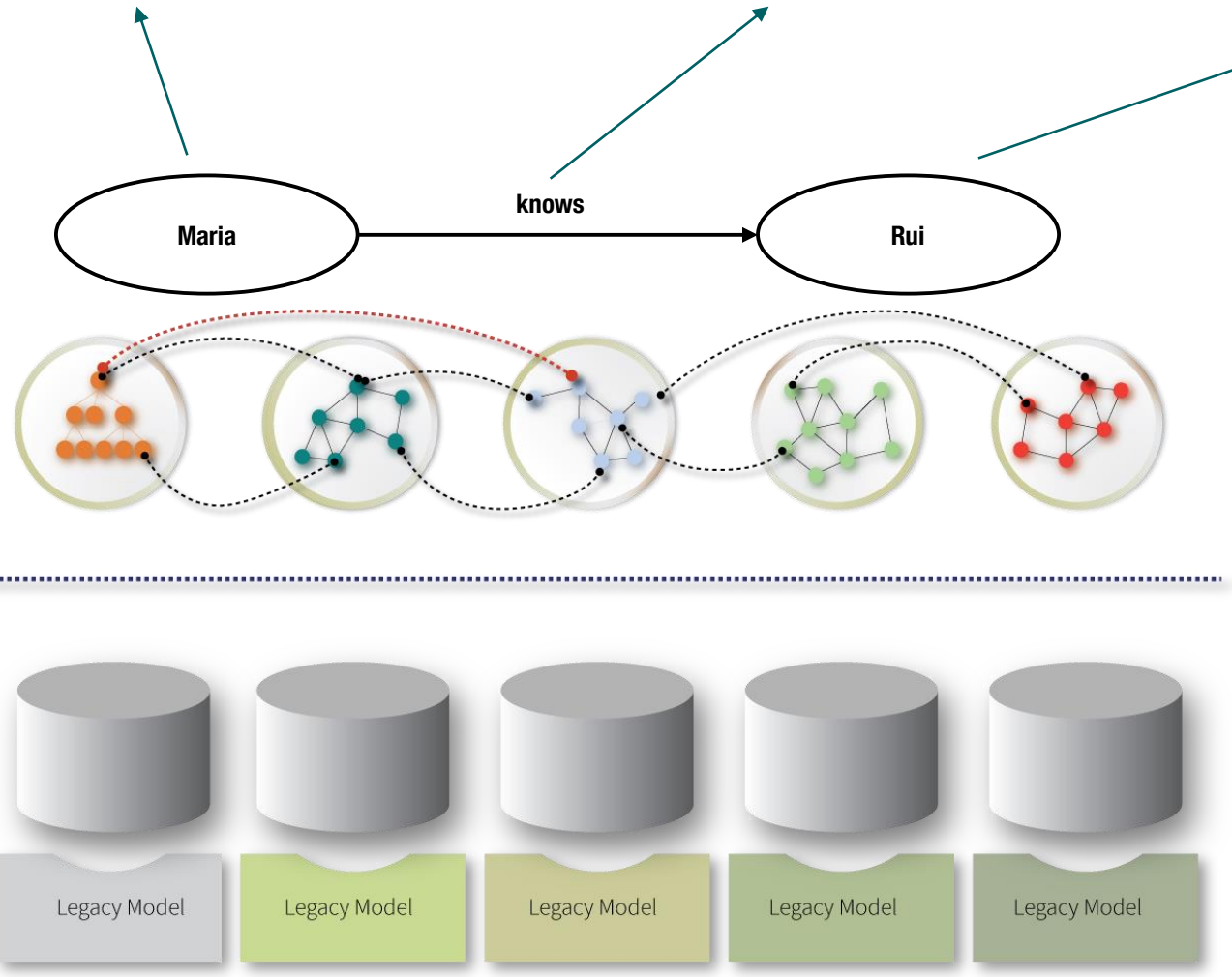
RDFLib vs. Jena

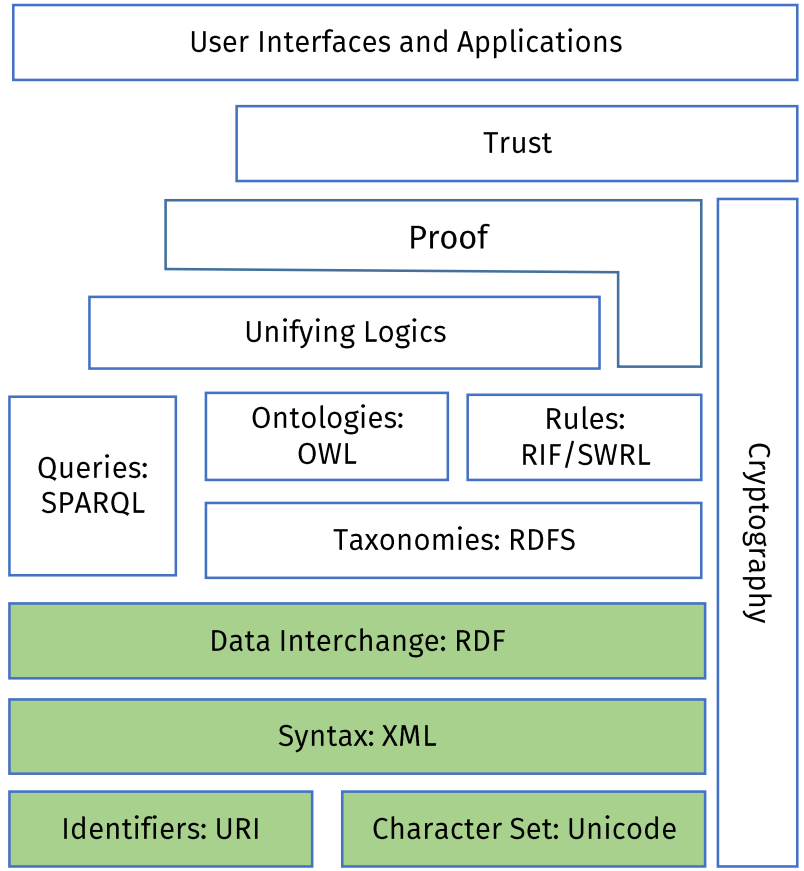
RDFLib:
- Ubiquitous, batteries included
- Pythonic
- Straight-forward

Jena:
- Feature-Rich, universal
- Robust, proven
- complex

Jena Architecture Overview

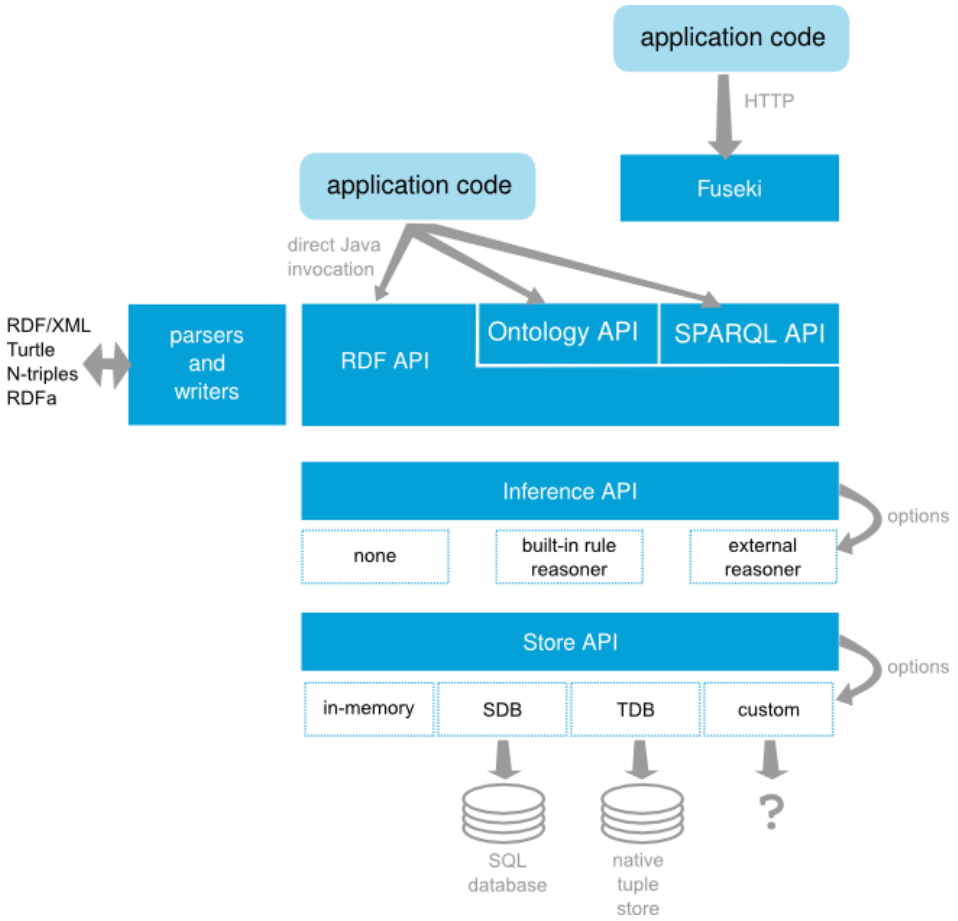
```
<http://lbd.org/people/Maria> <http://xmlns.com/foaf/0.1#knows> <http://dcg.fa.utl.pt/authors/rui-de-klerk> .
```

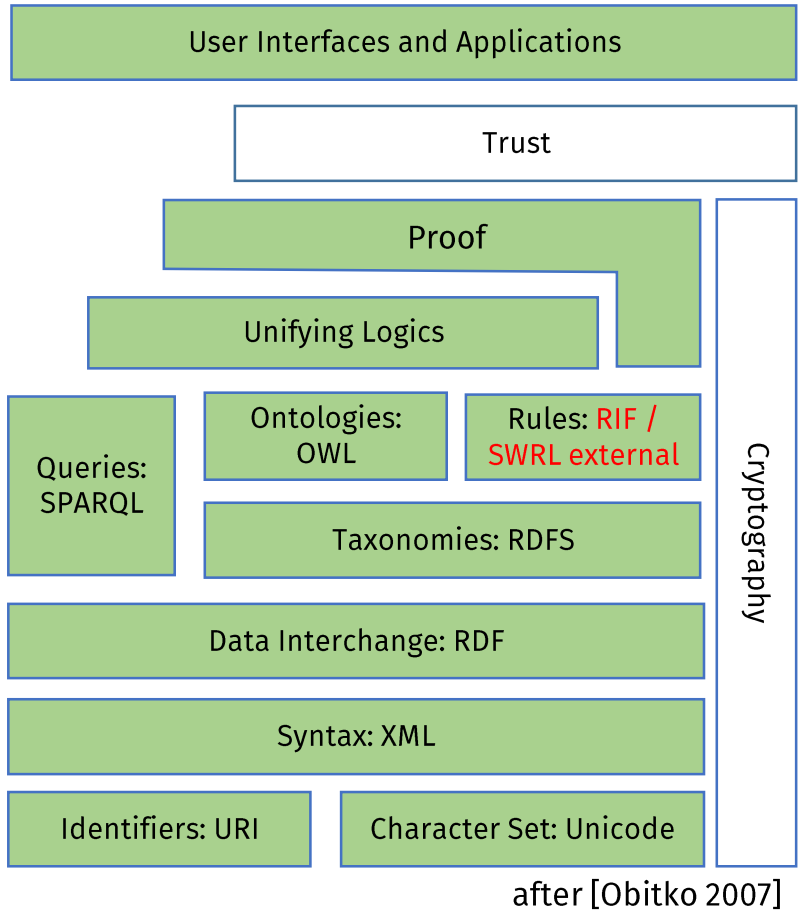




after [Obitko 2007]

Jena Architecture Overview

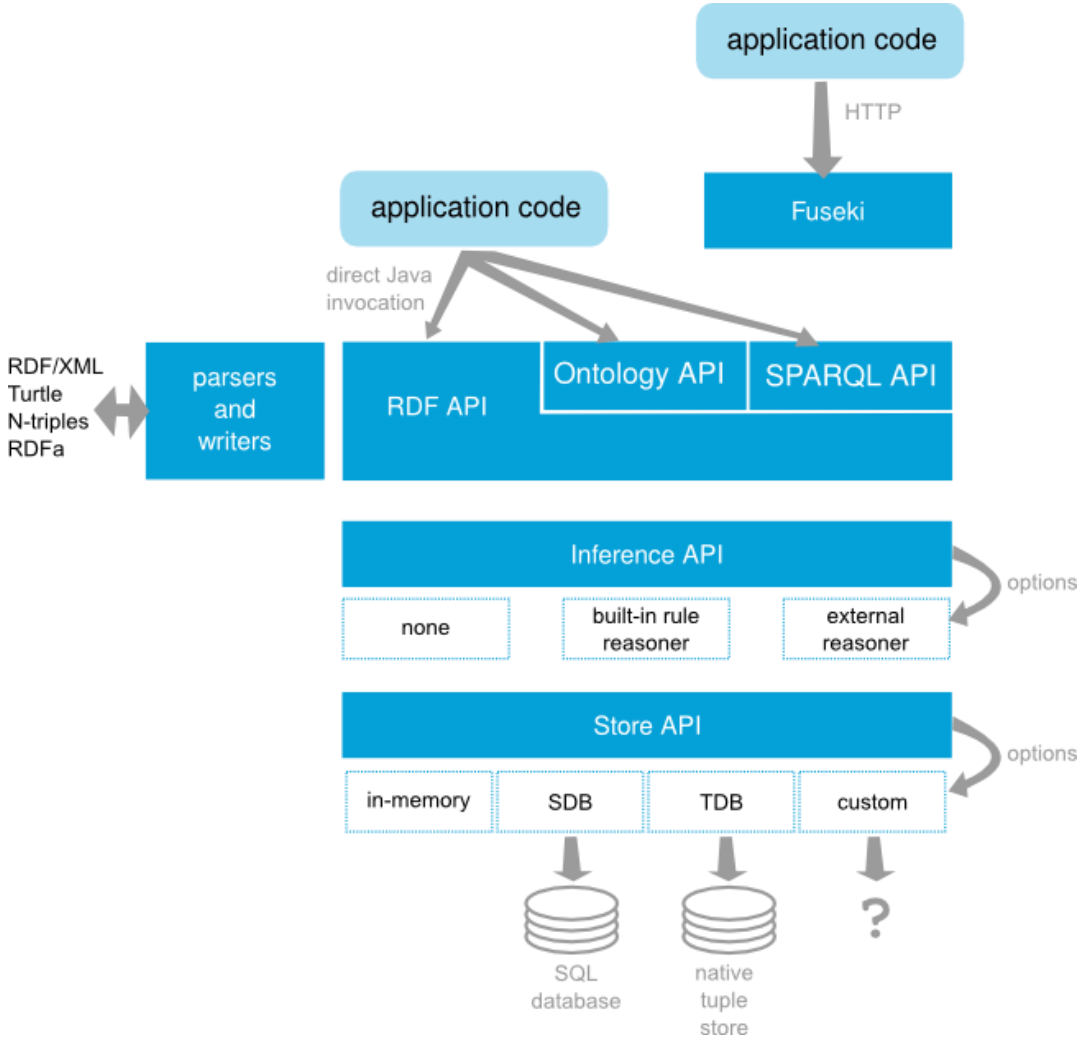




after [Obitko 2007]

Jena Architecture Overview

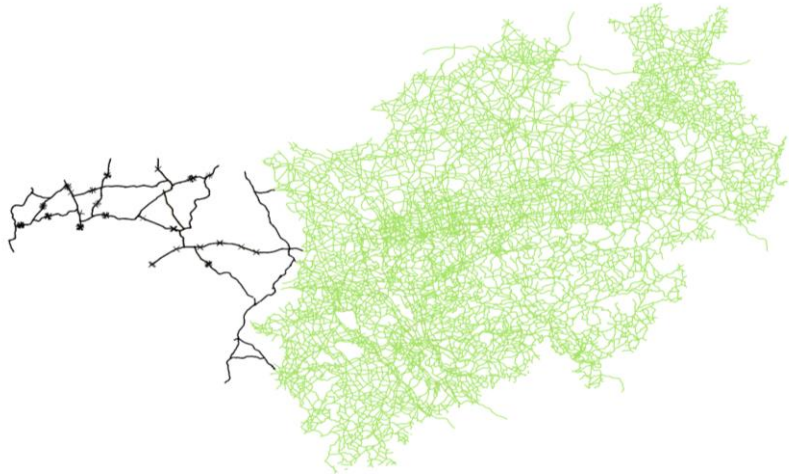
- **RDF**
  - **RDF API**
    - Interact with the core API to create and read Resource Description Framework (RDF) graphs. Serialise your triples using popular formats such as RDF/XML or Turtle.
  - **ARQ (SPARQL)**
    - Query your RDF data using ARQ, a SPARQL 1.1 compliant engine. ARQ supports remote federated queries and free text search.

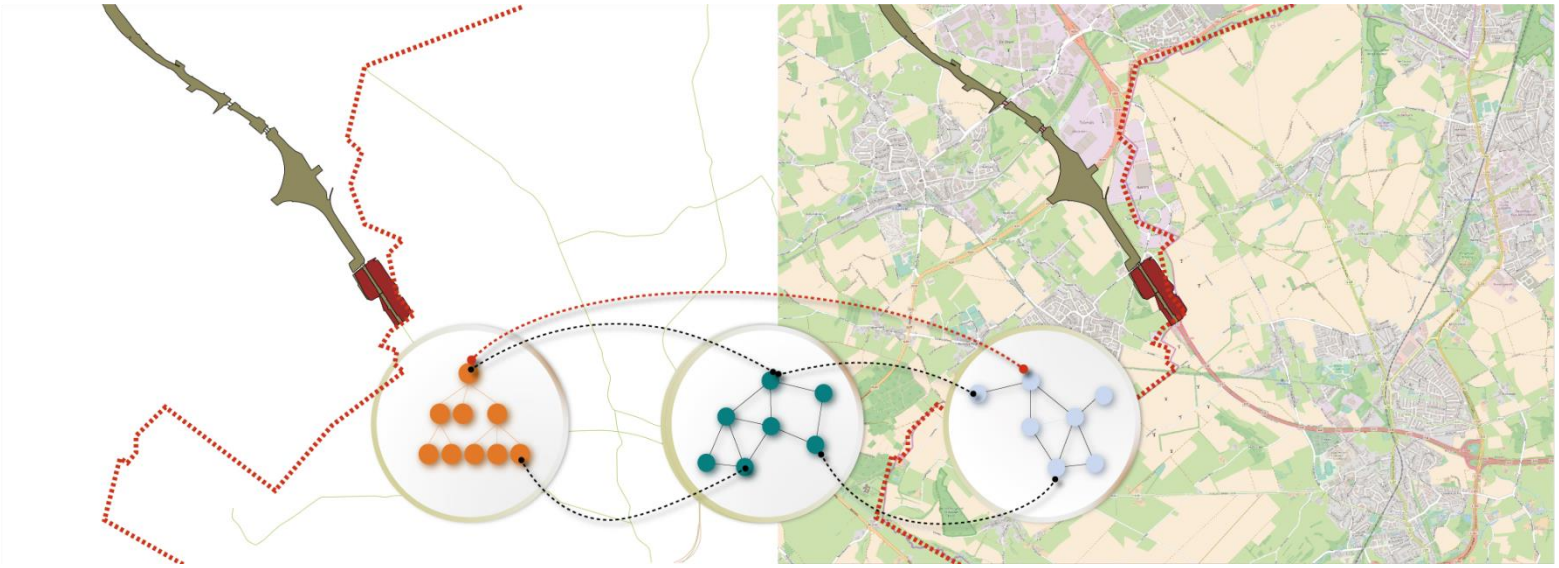


Jena efficiency example: okstraOWL

[Beetz & Borrmann 2018]

- okstraOWL and CB-NL instance data
- Additional classifications of road sections through schema mappings and SPARQL rules
- Graphs of > 140 Mio triples
- Queries across (physically distributed) graphs

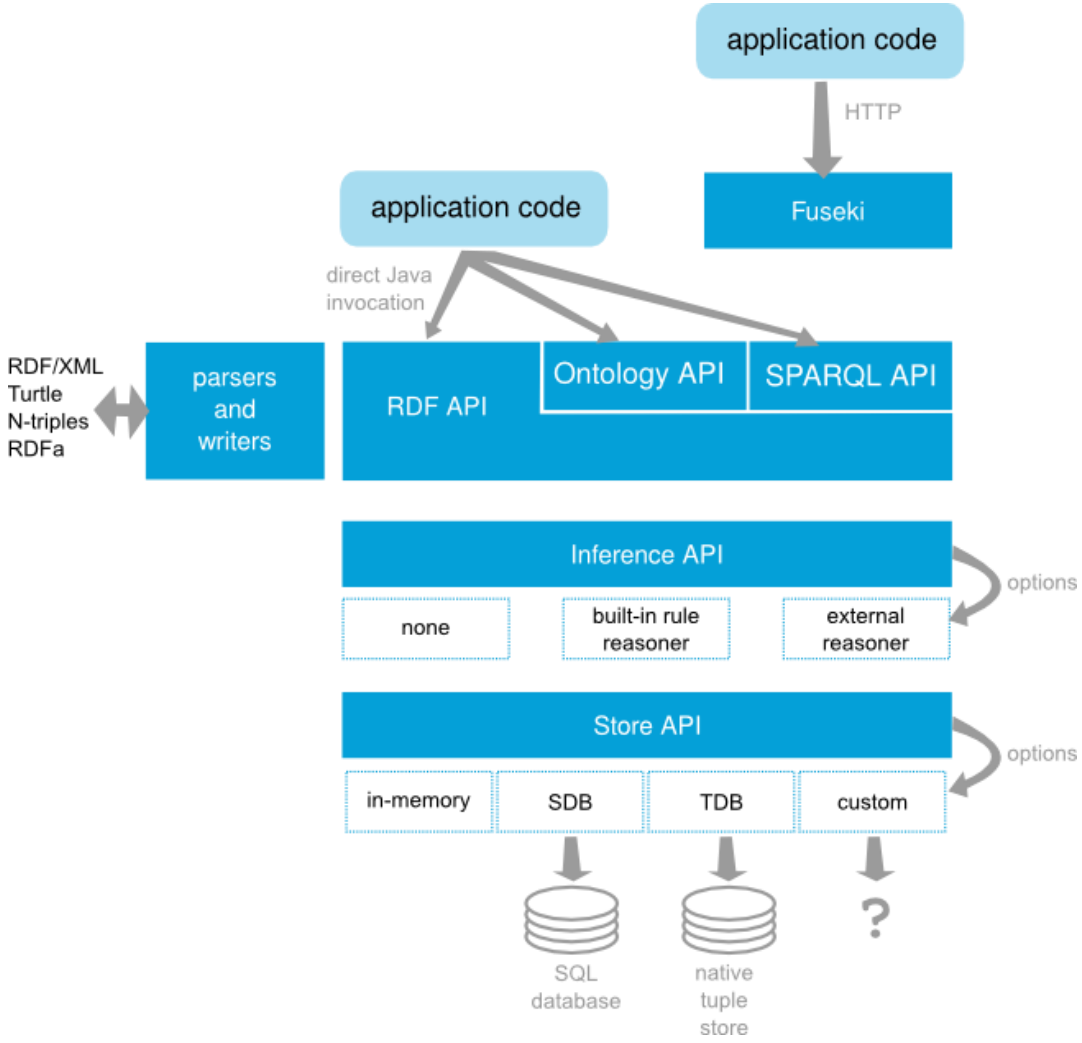




[Beetz & Borrmann 2018]

Jena Architecture Overview

- **RDF**
  - **RDF API**
    - Interact with the core API to create and read Resource Description Framework (RDF) graphs. Serialise your triples using popular formats such as RDF/XML or Turtle.
  - **ARQ (SPARQL)**
    - Query your RDF data using ARQ, a SPARQL 1.1 compliant engine. ARQ supports remote federated queries and free text search.



Extending ARQ : e.g. BIMSparql

[Zhang, Beetz, de Vries 2018]

Table 11

Query tested in the evaluation study

id	query body	use case types	description of the query including reference to provenance of real use case
Q1	Listing 2	model structure check	Find out windows and walls with the condition that the window is placed in the wall but they belong to different building storeys (Statsbygg, p. 66) [51].
Q2	Listing 3	quantity take-off	Count load bearing walls for each building storey (Solibri example) [48].
Q3	Listing 4	data consistency check	Retrieve walls which do not have the height quantity or height is inconsistent with its geometry representation (Solibri example) [48].
Q4	Listing 5	design check	Find out spaces which have the window-to-floor area ratio smaller than 0.3 (Solibri example) [48].
Q5	Listing 6	design check	Find geometry clashes (intersections) between walls and slabs (Rgd 2.1.6, p. 9) [45].
Q6	Listing 7	design check	Retrieve suspended ceilings that are too close (with the distance less than 0.4 meter) to the floor slabs in the above building storey (Statsbygg 56, p. 35) [51] .
Q7	Listing 8	design check	Find out walls that have bottom surfaces not touching upper surfaces of any floor slabs on the same building storey (Statsbygg 41 and 43, p. 30 and 31) [51] .

Table 12

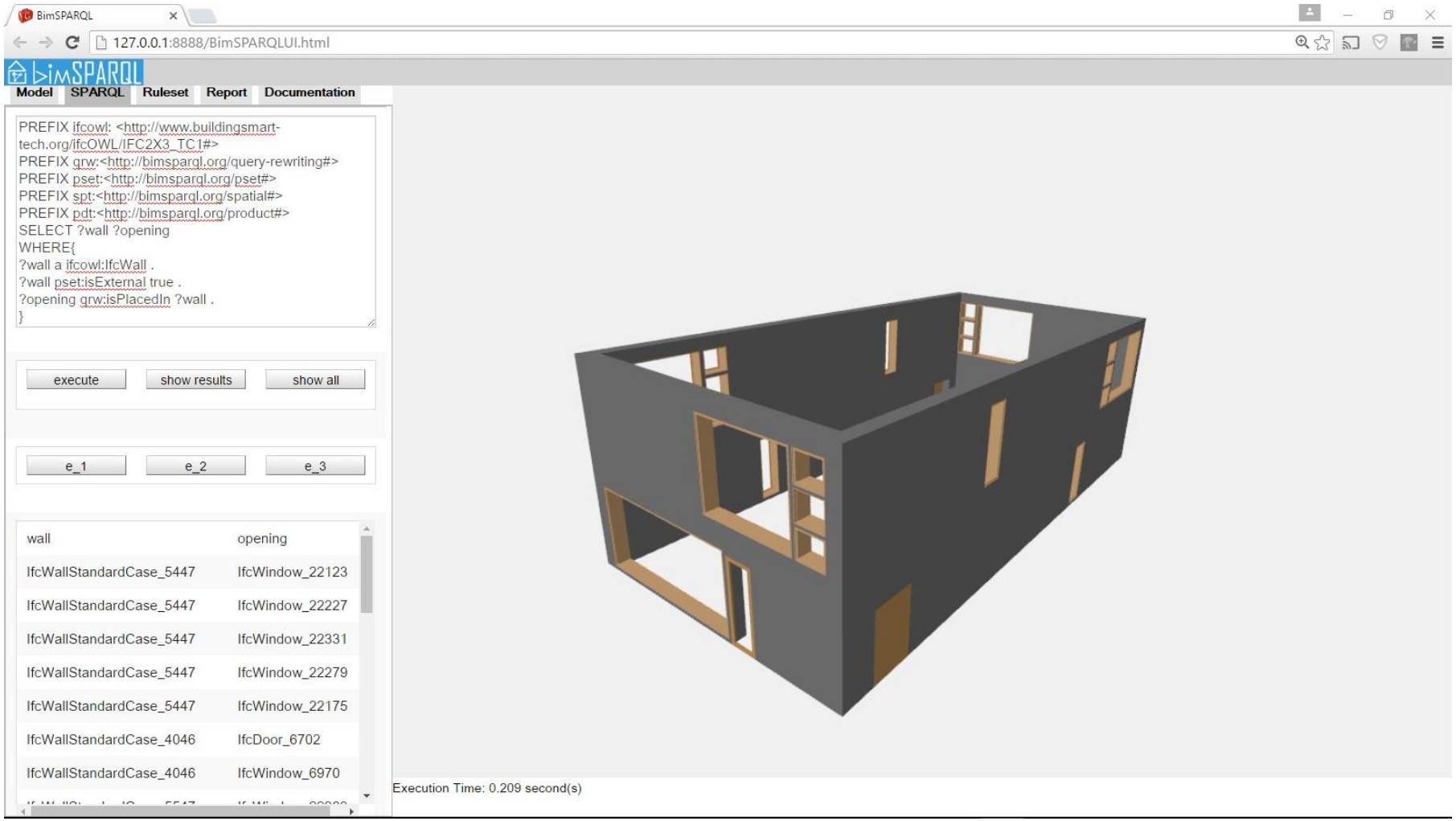
Query results and performance of Q1 to Q8 (see Table 11)on M1, M2, M3, M4 (see Table 10).

query	model	triple count (total)	avg. querying (s)	stand. derivation	result count
Q1 (Listing 2)	M1	298,631	0.033	0.053	0
	M2	1,790,927	0.059	0.054	0
	M3	14,502,777	0.110	0.190	31
Q2 (Listing 3)	M1	298,631	0.169	0.033	1
	M2	1,790,927	1.437	0.062	1
	M3	14,502,777	1.610	0.221	1
Q3 (Listing 4)	M1	298,631	0.023	0.00064	49
	M2	1,790,927	0.345	0.0051	495
	M3	14,502,777	0.679	0.0079	750

Function	Illustration	Description
pdt:hasBodyGeometry		Returns the geometry form of a product represented as a WKT literal (see Section 4.5). It either retrieves a WKT literal (see Section 4.5) to represent a 3D triangulated surface (TIN Z), or a geometry collection (GeometryCollection Z) in WKT.
pdt:hasAABB		Returns the axis-aligned bounding box of a product as a WKT literal (see Section 4.5).
pdt:hasMVBB		Returns the oriented minimum volume bounding box of a product as a WKT literal (see Section 4.5).
pdt:hasOverallHeight		Returns the height of axis aligned bounding box of a product as a numerical value.
pdt:hasSurface		Returns all plain surfaces of a product. Each of the surfaces is generated as a new binding for the triple pattern which uses this function.
pdt:hasUpperSurface		Returns the upper surface of a product, which is defined as surfaces that have the highest elevation and have normals of nearly (0,0,1), represented as a WKT literal (see Section 4.5). A use case of it is shown in Listing 8.
pdt:hasVolume		Returns the volume of the product as a numerical value.

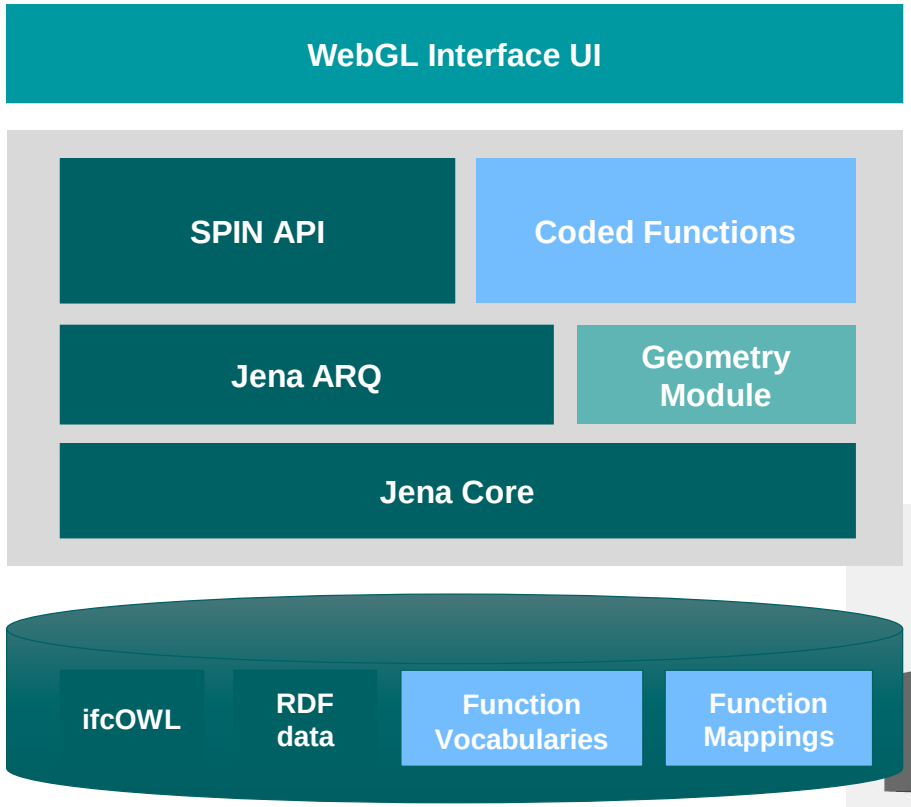
Extending ARQ : e.g. BIMSparql

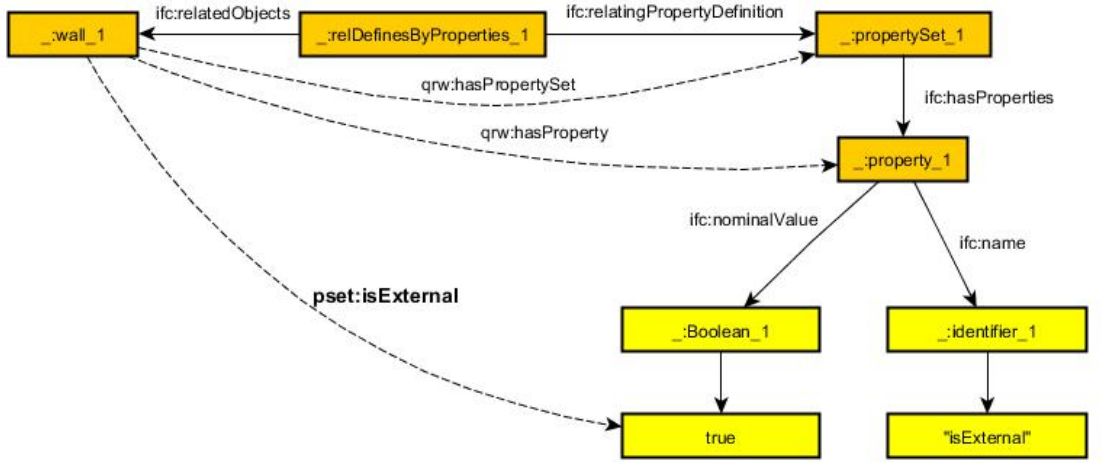
[Zhang, Beetz, de Vries 2018]

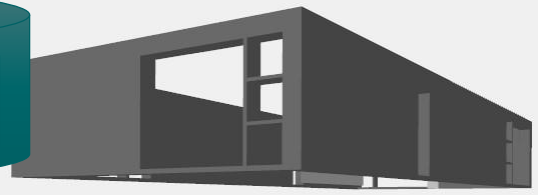


https://www.youtube.com/watch?v=0y4oT7zpEDQ

Using the Jena framework in your application stack







"Count external walls (walls with IsExternal property of true) for each storey"

```
SELECT ?storey (COUNT(?wall) AS ?q)
WHERE{
  ?wall a ifc:IfcWallStandardCase .
  ?wall pset:isExternal true .
  ?wall qrw:isContainedIn ?storey .
  ?storey a ifc:IfcBuildingStorey .
} GROUP BY ?storey
```

Extending ARQ : e.g. BIMSparql

[Zhang, Beetz, de Vries 2018]

TABLE 705.8
MAXIMUM AREA OF EXTERIOR WALL OPENINGS BASED ON FIRE SEPARATION DISTANCE AND DEGREE OF OPENING PROTECTION

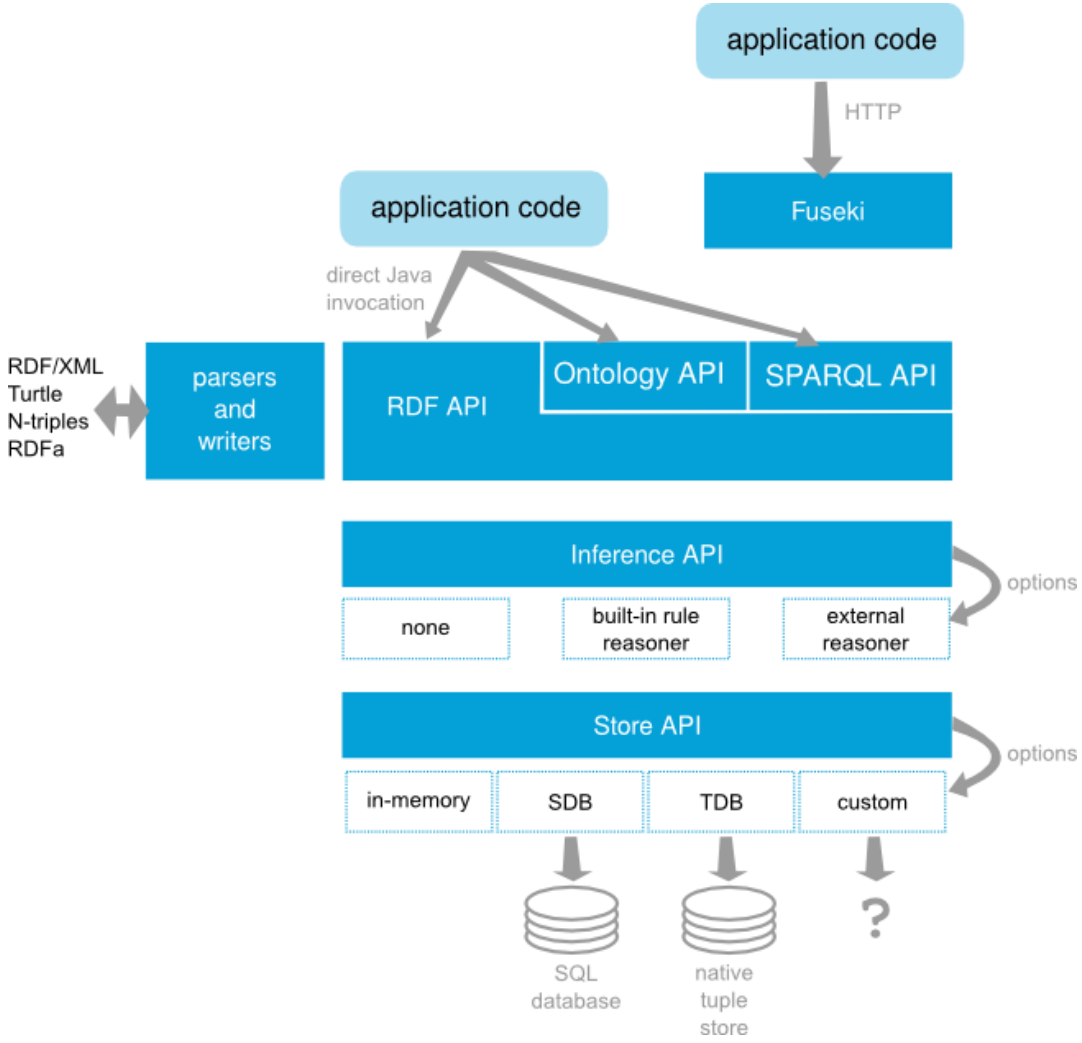
FIRE SEPARATION DISTANCE (feet)	DEGREE OF OPENING PROTECTION	ALLOWABLE AREA[a]
0 to less than 3[b,c]	Unprotected, Nonsprinklered (UP, NS)	Not Permitted
	Unprotected, Sprinklered (UP, S)i	Not Permitted
	Protected (P)	Not Permitted
3 to less than 5[d,e]	Unprotected, Nonsprinklered (UP, NS)	Not Permitted
	Unprotected, Sprinklered (UP, S)i	15%
	Protected (P)	15%
5 to less than 10[e,f]	Unprotected, Nonsprinklered (UP, NS)	10%h
	Unprotected, Sprinklered (UP, S)i	25%
	Protected (P)	25%
10 to less than 15[e,f,g]	Unprotected, Nonsprinklered (UP, NS)	15%h
	Unprotected, Sprinklered (UP, S)i	45%
	Protected (P)	45%
15 to less than 20[f,g]	Unprotected, Nonsprinklered (UP, NS)	25%
	Unprotected, Sprinklered (UP, S)i	75%
	Protected (P)	75%
20 to less than 25[f,g]	Unprotected, Nonsprinklered (UP, NS)	45%
	Unprotected, Sprinklered (UP, S)i	No Limit
	Protected (P)	No Limit
25 to less than 30[f,g]	Unprotected, Nonsprinklered (UP, NS)	70%
	Unprotected, Sprinklered (UP, S)i	No Limit
	Protected (P)	No Limit
30 or more	Unprotected, Nonsprinklered (UP, NS)	No Limit
	Unprotected, Sprinklered (UP, S)i	Not Required
	Protected (P)	Not Required

```
CONSTRUCT{
 _:b0 a spin:ConstraintViolation .
 _:b0 spin:violationRoot ?storey .
 _:b0 spin:violationLevel spin:Warning .

}WHERE {
?storey a ifc:IfcBuildingStorey .
?fs a ibc:FireSeparation .
(?storey, ?fs) spt:distanceXY ?d .
?storey pset:sprinklerProtection ?bool .
?storey ibc:protectedOpeningArea ?Ap .
?storey ibc:unprotectedOpeningArea ?Au .

{?b1 ibc:minFSDistance ?min .
 ?b1 ibc:maxFSDistance ?max .
 ?b1 ibc:sprinklerProtection ?bool .
 ?b1 ibc:openingProtection true .
 ?b1 ibc:allowableArea ?ap
 FILTER (?d>=?min && ?d<?max )
}
UNION
{?b2 ibc:minFSDistance ?min .
 ?b2 ibc:maxFSDistance ?max .
 ?b2 ibc:sprinklerProtection ?bool .
 ?b2 ibc:openingProtection false .
 ?b2 ibc:allowableArea ?au
 FILTER (?d>=?min && ?d<?max )
}

FILTER (?Ap/?ap+?Au/au>1)
}
```

https://www.youtube.com/watch?v=0y4oT7zpEDQ

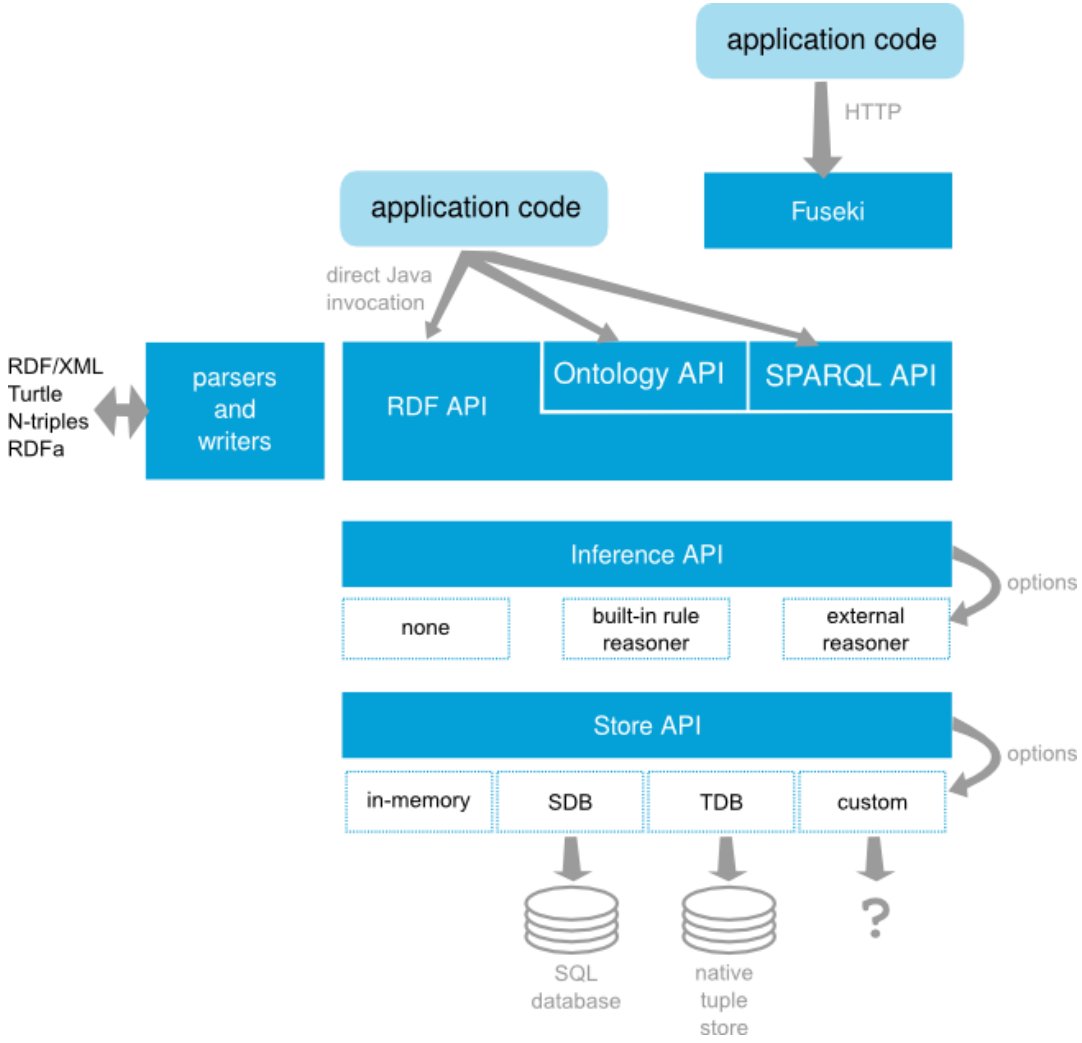
Jena Architecture Overview

- **OWL**
  - **Ontology API**
    - Work with models, RDFS and the Web Ontology Language (OWL) to add extra semantics to your RDF data.
  - **Inference API**
    - Reason over your data to expand and check the content of your triple store. Configure your own inference rules or use the built-in OWL and RDFS reasoners.



Jena Architecture Overview

- **Triple store**
  - **TDB**
    - Persist your data using TDB, a native high performance triple store. TDB supports the full range of Jena APIs.
  - **Fuseki**
    - Expose your triples as a SPARQL end-point accessible over HTTP. Fuseki provides REST-style interaction with your RDF data.



Jena Utilities: riot

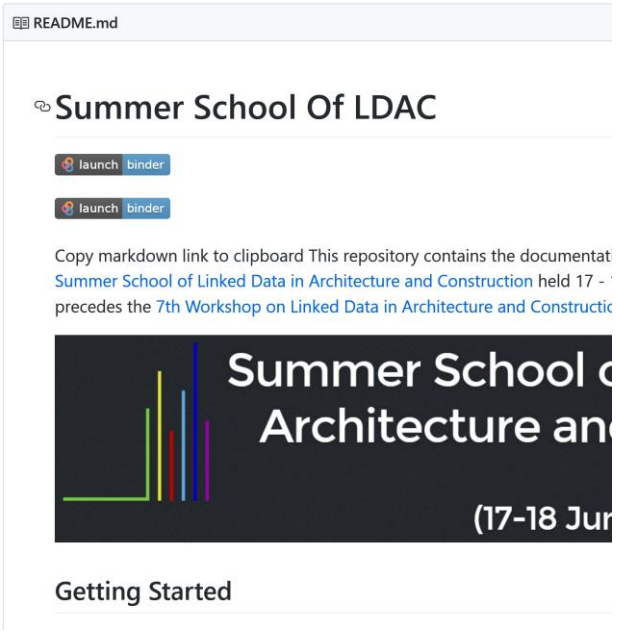
- Command line tool and utility class
- Blazing fast conversion, I/O
- pretty-printing
- Streaming

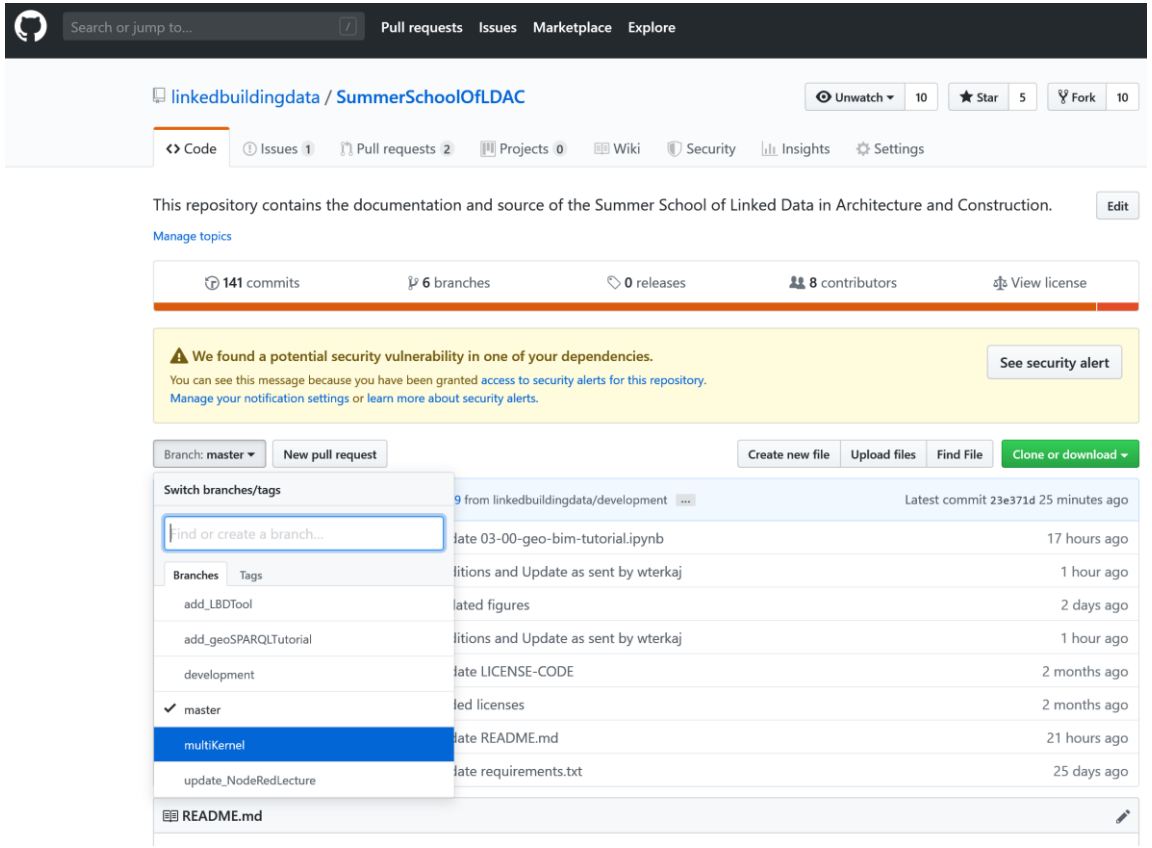
Extension	Language
.ttl	Turtle
.nt	N-Triples
.nq	N-Quads
.trig	TriG
.rdf	RDF/XML
.owl	RDF/XML
.jsonld	JSON-LD
.trdf	RDF Thrift
.rt	RDF Thrift
.rj	RDF/JSON
.trix	TriX

It’s your turn now!

https://github.com/linkedbuildingdata/SummerSchoolOfLDAC

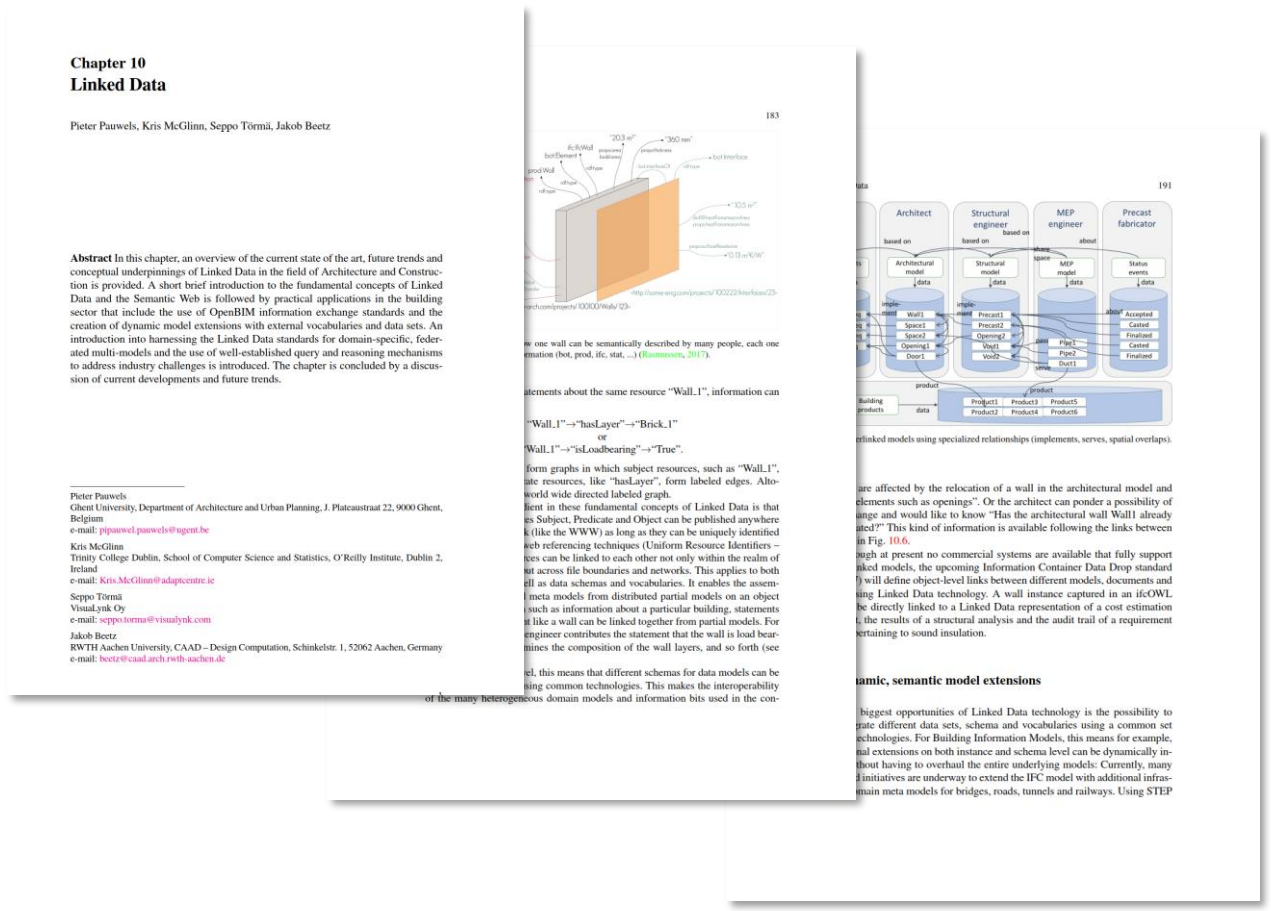
- Switch to the multikernel branch
- Press Launch Binder button
- Start with Hello world





Thank you Advertisement

Thank you j.beetz@caad.arch.rwth-aachen.de



Chapter 10
Linked Data

Pieter Pauwels, Kris McGlinn, Seppo Törmä, Jakob Beetz

Abstract In this chapter, an overview of the current state of the art, future trends and conceptual underpinnings of Linked Data in the field of Architecture and Construction is provided. A short brief introduction to the fundamental concepts of Linked Data and the Semantic Web is followed by practical applications in the building sector that include the use of OpenBIM information exchange standards and the creation of dynamic model extensions with external vocabularies and data sets. An introduction into harnessing the Linked Data standards for domain-specific, federated multi-models and the use of well-established query and reasoning mechanisms to address industry challenges is introduced. The chapter is concluded by a discussion of current developments and future trends.

Pieter Pauwels
Ghent University, Department of Architecture and Urban Planning, J. Plateaustraat 22, 9000 Ghent, Belgium
e-mail: pipauwel.pauwels@ugent.be
Kris McGlinn
Trinity College Dublin, School of Computer Science and Statistics, O'Reilly Institute, Dublin 2, Ireland
e-mail: Kris.McGlinn@adaptcentre.ie
Seppo Törmä
VisuaLynk Oy
e-mail: seppo.torma@visualynk.com
Jakob Beetz
RWTH Aachen University, CAAD – Design Computation, Schinkelstr. 1, 52062 Aachen, Germany
e-mail: beetz@caad.arch.rwth-aachen.de