

Semantics, knowledge graphs and ontologies in practice

Jose Emilio Labra Gayo

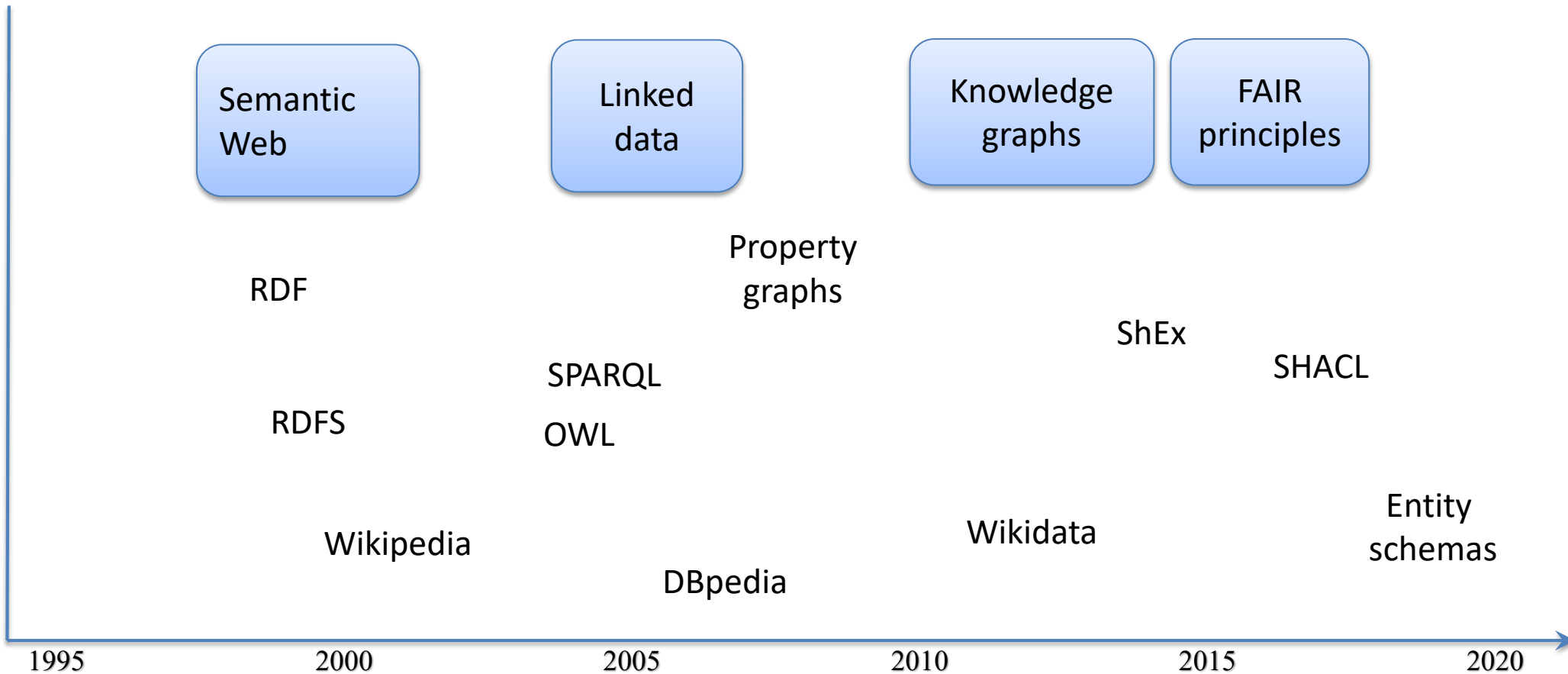
WESO Research group
University of Oviedo, Spain



Schedule

| Day | Title | Topics |
|--------|--|--|
| Day 1. | Semantic Technologies and Knowledge graphs | Semantic Web Linked data Knowledge graphs RDF data model Property graphs Wikibase graphs Examples and applications |
| Day 2. | RDF data modelling and SPARQL | Data modelling exercises with RDF and turtle SPARQL |
| Day 3. | Validating RDF data | Shape Expressions (ShEx) SHACL Validating Knowledge Graphs |
| Day 4. | Advanced topics | ShEx and SHACL compared Shape applications and tools Reasoning RDFS OWL Nanopublications |

ROADMAP



Session 3. Advanced topics

Jose Emilio Labra Gayo

WESO Research group
University of Oviedo, Spain



ShEx and SHACL compared

Some similarities

Similar goal: describe and validate RDF graphs

Both employ the word "shape"

Node constraints similar in both languages

Constraints on incoming/outgoing arcs

Both allow to define cardinalities

Both have RDF syntax

Both have an extension mechanism

ShEx and SHACL compared

Main differences

| | ShEx | SHACL |
|---------------------------|--------------------------------|---|
| Underlying philosophy | Structure definition | Constraint checking |
| Syntax | Compact syntax + RDF | RDF |
| Notion of shape | Only structure | Structure + target decls. |
| Default cardinalities | {1,1} | {0,*} |
| Shapes and inference | No | SHACL specific entailment |
| Recursion | Part of the language | Undefined |
| Repeated properties | Part of the language | Conjunction by default Requires qualifiedValueShapes |
| Property paths | Nested shapes | SPARQL like |
| Property pair comparisons | Unsupported in current version | Part of the language |
| Extension mechanism | Semantic actions | SHACL-SPARQL |
| Validation triggering | Query shape map | Target declarations |
| Result of validation | Result shape map | Validation report |

ShEx: Shape Expressions

Describe RDF data

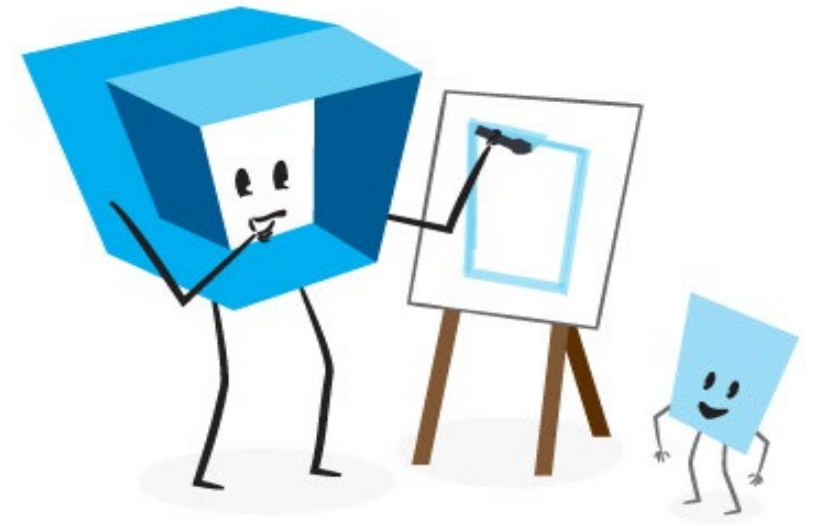
Descriptions focus on what is the shape of the RDF data graph

Focus = RDF graph

Validate = check if data matches the descriptions

Main focus = nodes that match (shape maps)

It can also check nodes that don't match



Description

SHACL: Shapes Constraint Language

Constraints on RDF data

Constraints focus on the RDF data graph and things we don't allow

Focus = RDF graph (data)

Validate = check if data doesn't violate the constraints

Focus = nodes that don't pass the constraints (violations)

But it can also check nodes that pass the constraints



Constraint

ShEx for Property graphs

Recent paper: "ProGS: Property graph shapes language"

<https://arxiv.org/abs/2107.05566>

Extends SHACL to support Property graphs

In the same way, PShEx has been defined as a ShEx extension to support Property graphs

Adds constraints on nodes/property qualifiers

Proposal recently published at: <https://arxiv.org/abs/2110.11709>

ShEx for Wikibase graphs

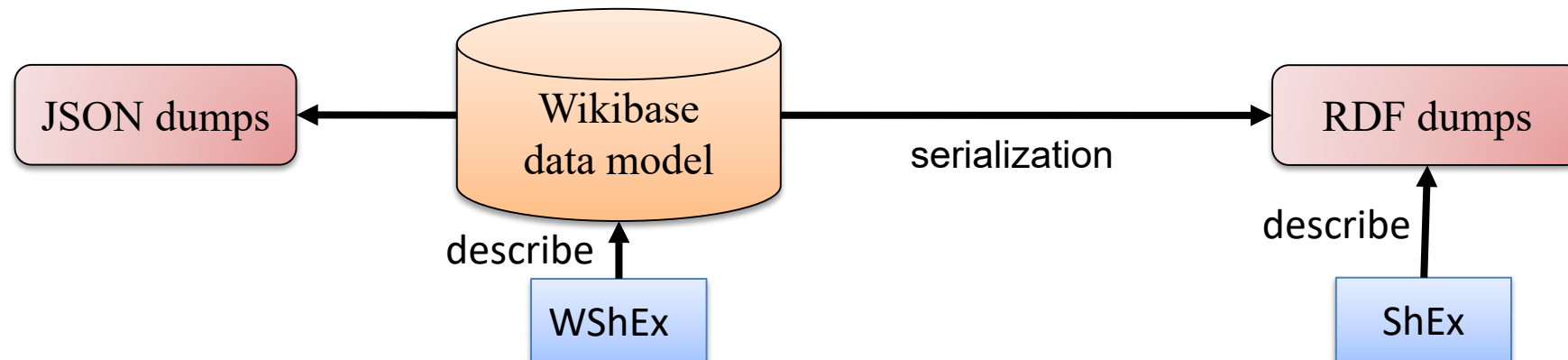
Wikidata already added support for ShEx

Entity schemas extension

WShEx = extension of ShEx that supports qualifiers/references

Can be used to describe and validate Wikibase graphs

Defined in: <https://arxiv.org/abs/2110.11709>



WShEx

```

<Researcher> {
  <birthPlace>    @<Place> ;
  <awarded>       @<Award> {{
    <togetherWith> @<Researcher> *
  }} *
}
<Place> {
  <country>       @<Country>
}
<Country> {}

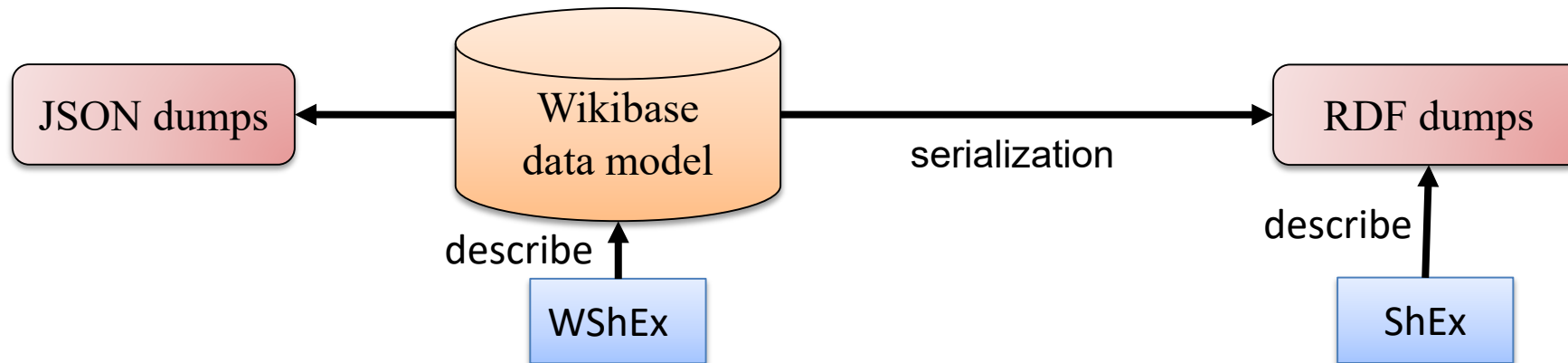
```

ShEx

```

<Researcher> {
  wdt:birthPlace @<Place> ;
  p:birthPlace {
    ps:birthPlace @<Place>
  } ;
  wdt:awarded    @<Awarded> ;
  p:awarded {
    ps:awarded    @<Awarded> ;
    pq:togetherWith @<Researcher> *;
  } *
}
<Place> {
  wdt:country @<Country> ;
  p:country {
    ps:country @<Country> }
}
<Country> {}

```



Shapes Applications and tools

Wikidata and wikibase



In May, 2019, Wikidata announced ShEx adoption

New namespace for schemas

Example:

<https://www.wikidata.org/wiki/EntitySchema:E2>

Wikibase also contains entity schemas

Online demo: [wikishape](#)

EntitySchema Discussion Read View history Search devwiki

Disease (E2)

| language code | label | description | aliases | edit |
|---------------|---------|--|-----------------------|----------------------|
| en | Disease | Shape Expression for diseases coming from Disease Ontology in Wikidata (Source: https://github.com/andrawaag/Genewiki-ShEx/blob/master/diseases/Wikidata-Disease-Ontology.shex) | Sickness Illness | edit |



Solid project

SOLID (SOcial Linked Data): Promoted by Tim Berners-Lee

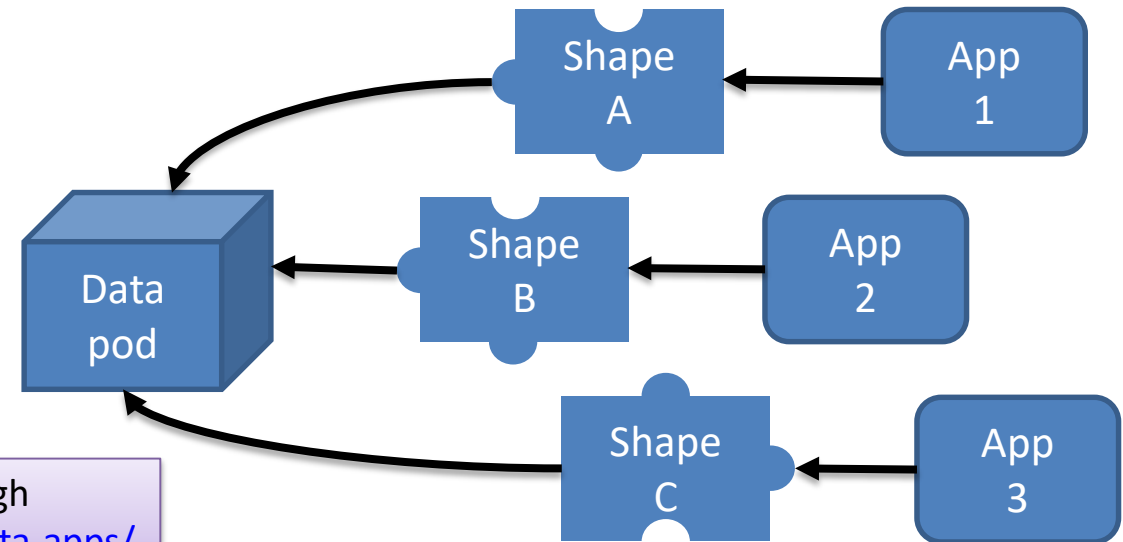
Goal: Re-decentralize the Web

Separate data from apps

Give users more control about their data

Internally using linked data & RDF

Shapes needed for interoperability



"...I just can't stop thinking about shapes.", Ruben Verborgh

<https://ruben.verborgh.org/blog/2019/06/17/shaping-linked-data-apps/>

Other use cases

HL7 FHIR.

Example: <https://www.hl7.org/fhir/observation.html>

ELI validator

SHACL shapes obtained from Excel sheets:

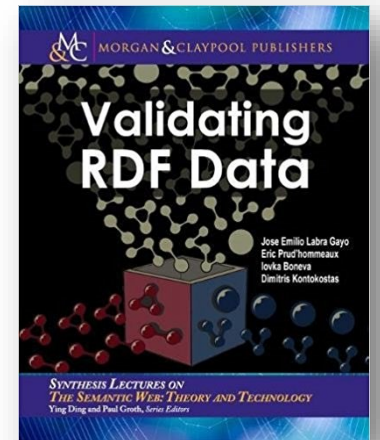
<https://webgate.ec.europa.eu/eli-validator/home>

SHACL adoption supported by Top Quadrant

See: <https://www.topquadrant.com/technology/shacl/>

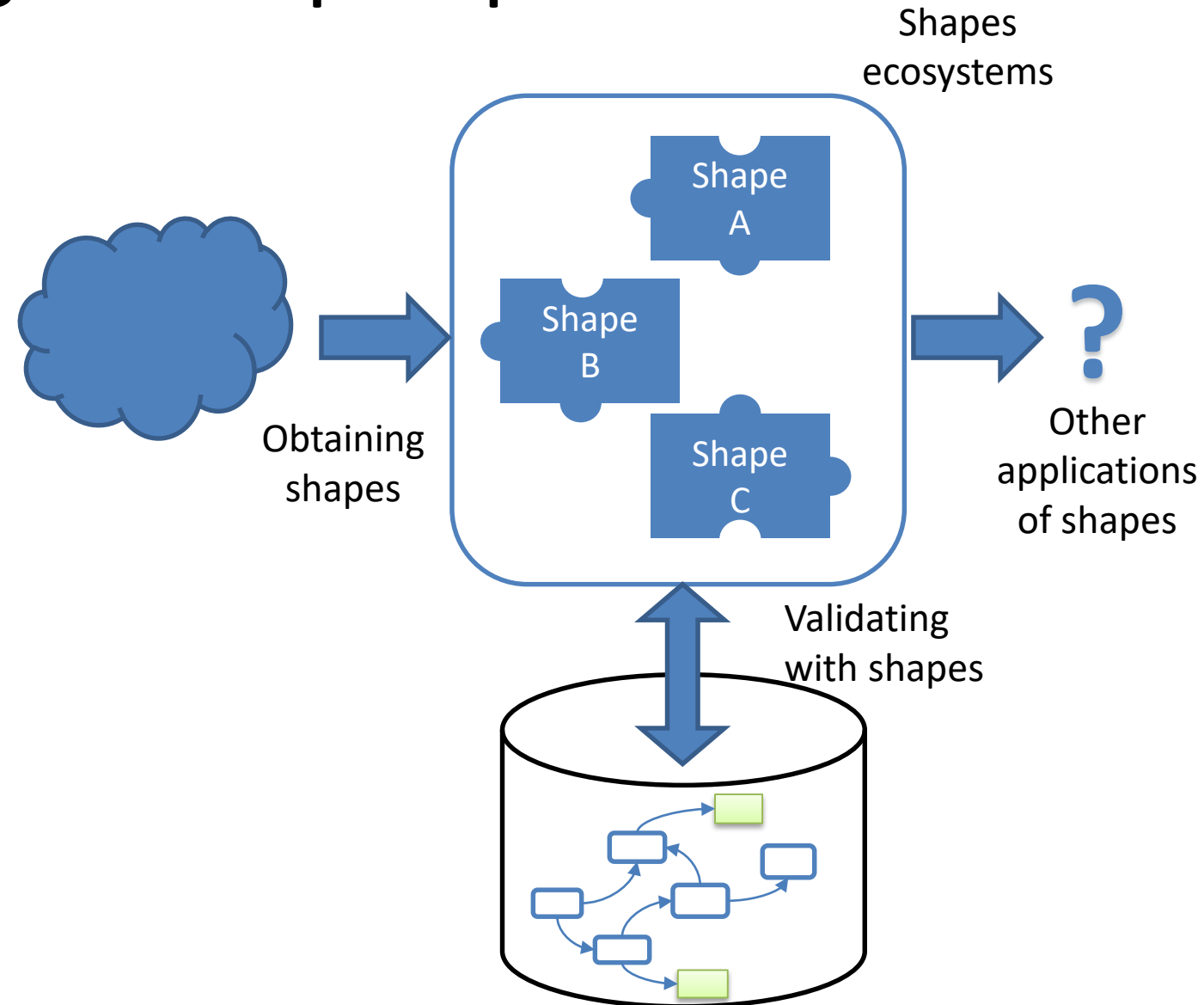
More info:

Chapter 6 of Validating RDF data: <http://book.validatingrdf.com/bookHtml012.html>



Tools: challenges and perspectives

Validating with shapes
Obtaining shapes
Other applications of shapes
Shapes ecosystems



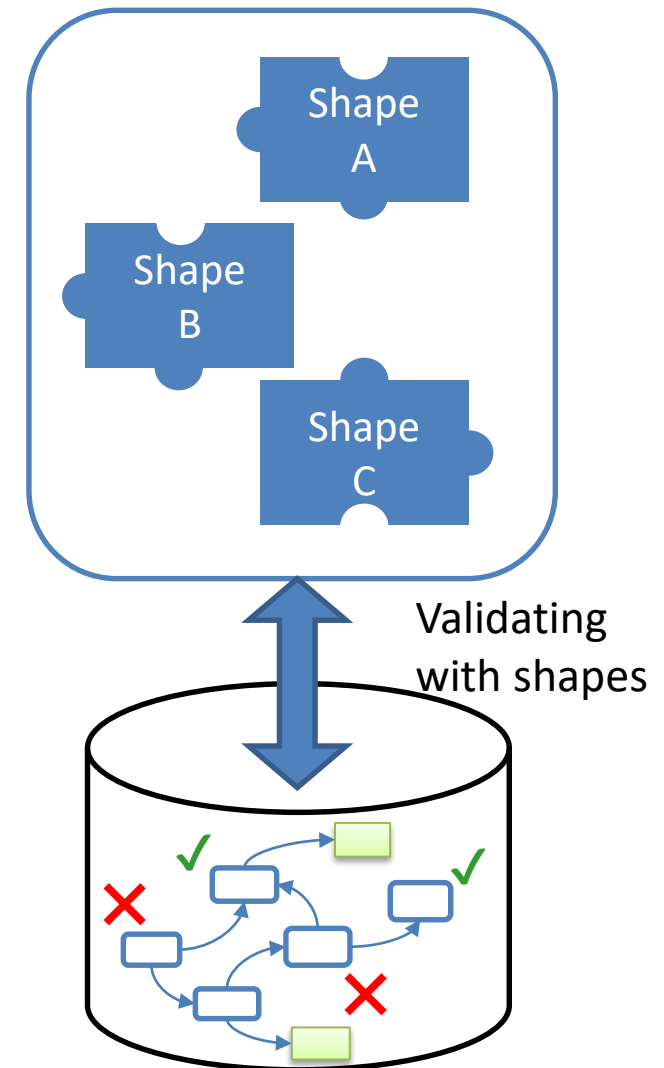
Validating with shapes

Libraries and command line validators

Online demos

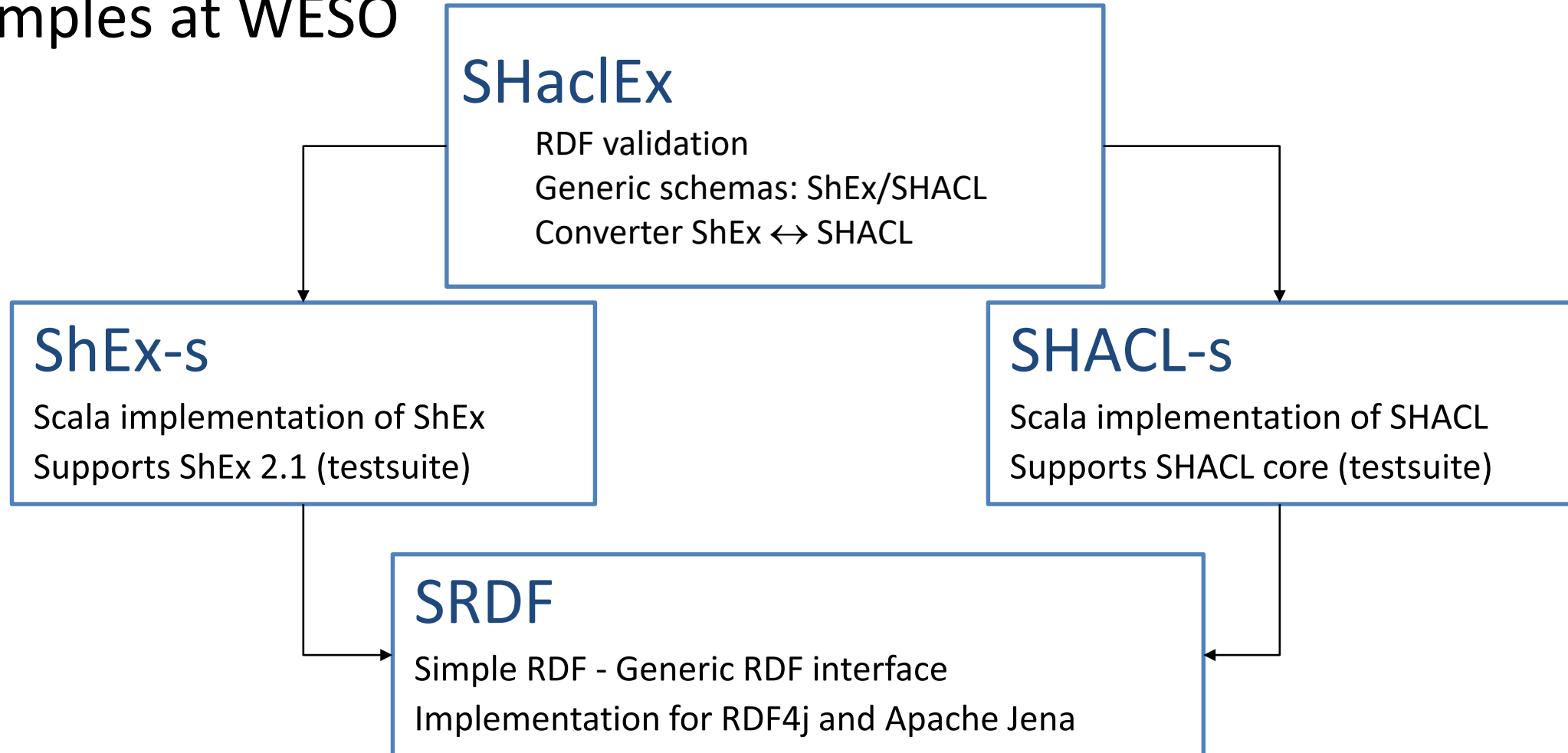
Integrated in ontology editors

Continuous integration with Shapes



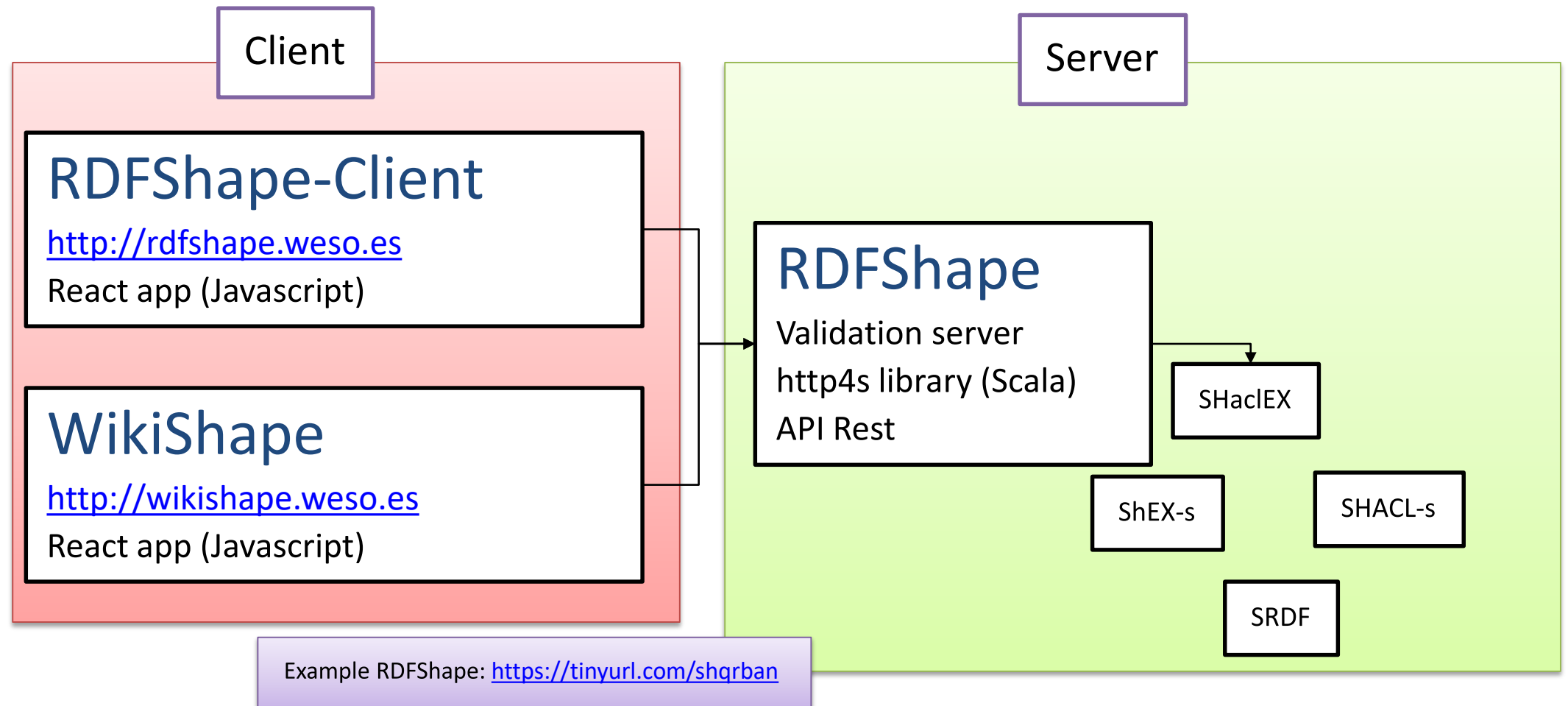
Libraries and command line validators

Examples at WESO



Online demos

Web Demos and playgrounds



Continuous integration with Shapes

Coexistence between ontologies/shapes

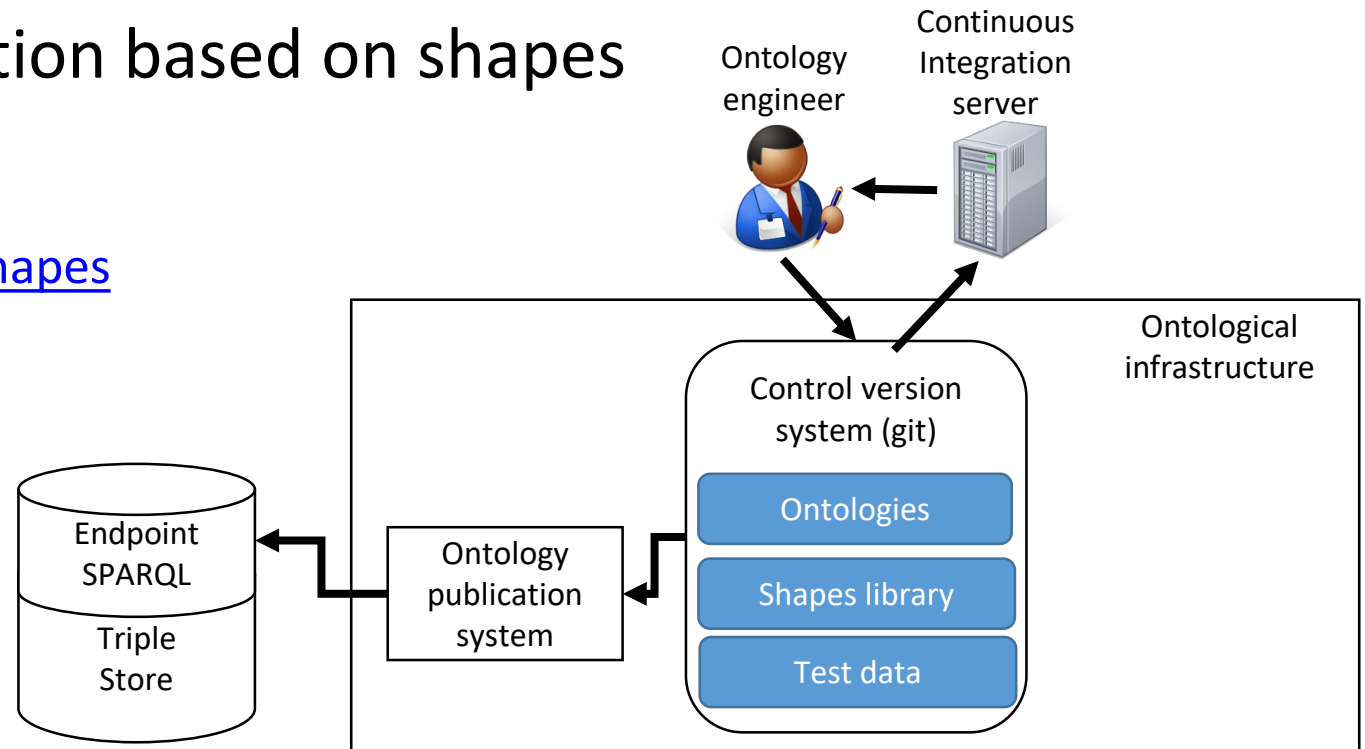
Shapes can validate the behaviour of inference systems

Shapes pre- and post- inference

TDD and continuous integration based on shapes

Gene Ontology Shapes:

<https://github.com/geneontology/go-shapes>

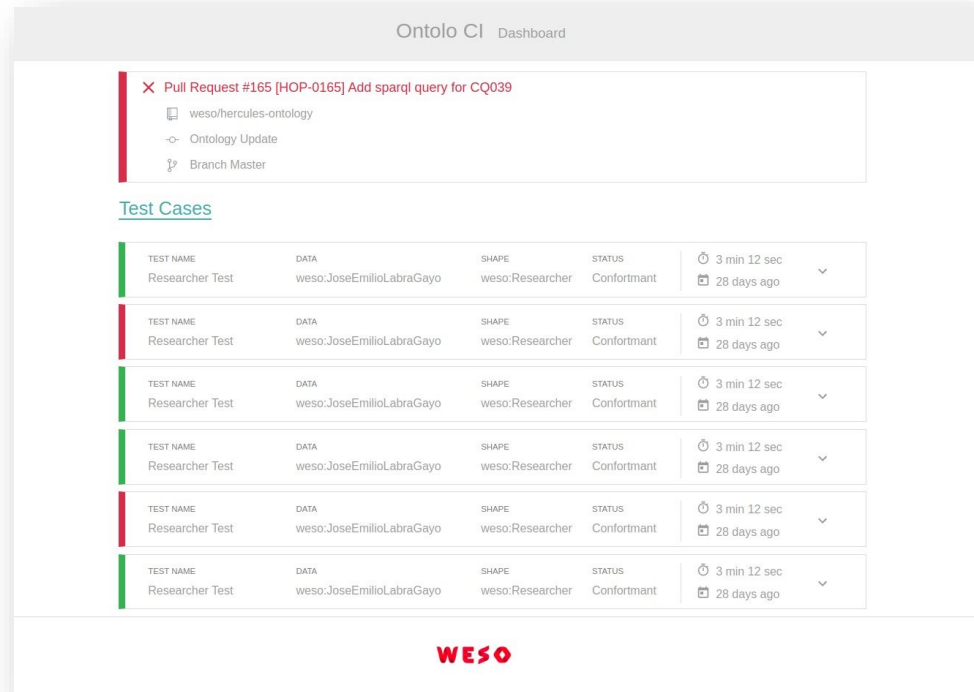


Developed as part of HERCULES-Ontology

Test-Driven-Development applied to Ontologies

Input:

- Ontologies
- Shapes
- Test data
- Input shape map (SPARQL competency question)
- Expected result shape map



Obtaining shapes

Shapes editors

- Text-based editors

- Visual editors and visualizers

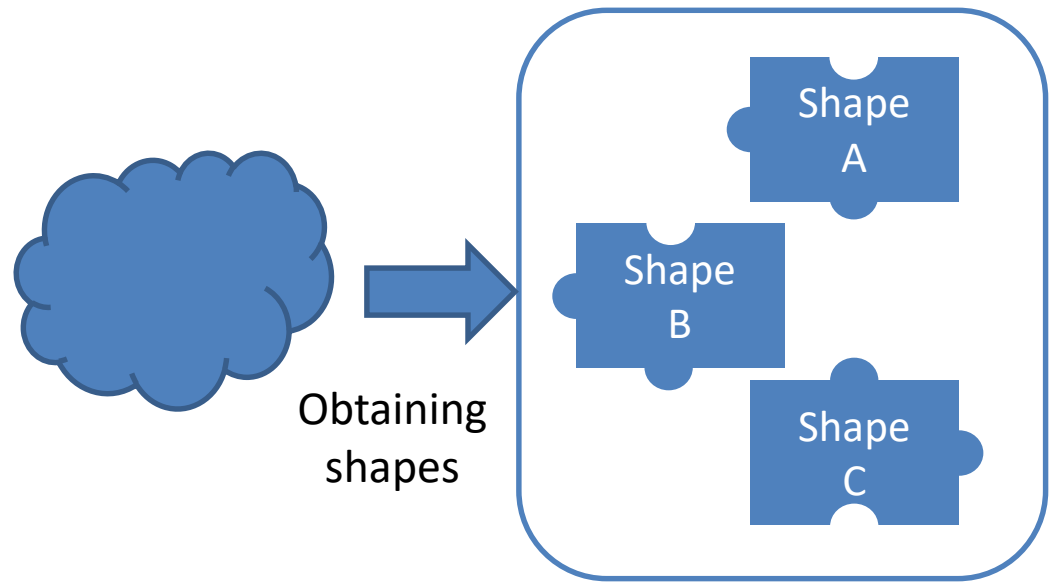
Obtaining shapes from...

- Spreadsheets

- RDF data

- Ontologies

- Other schemas (XML Schema)

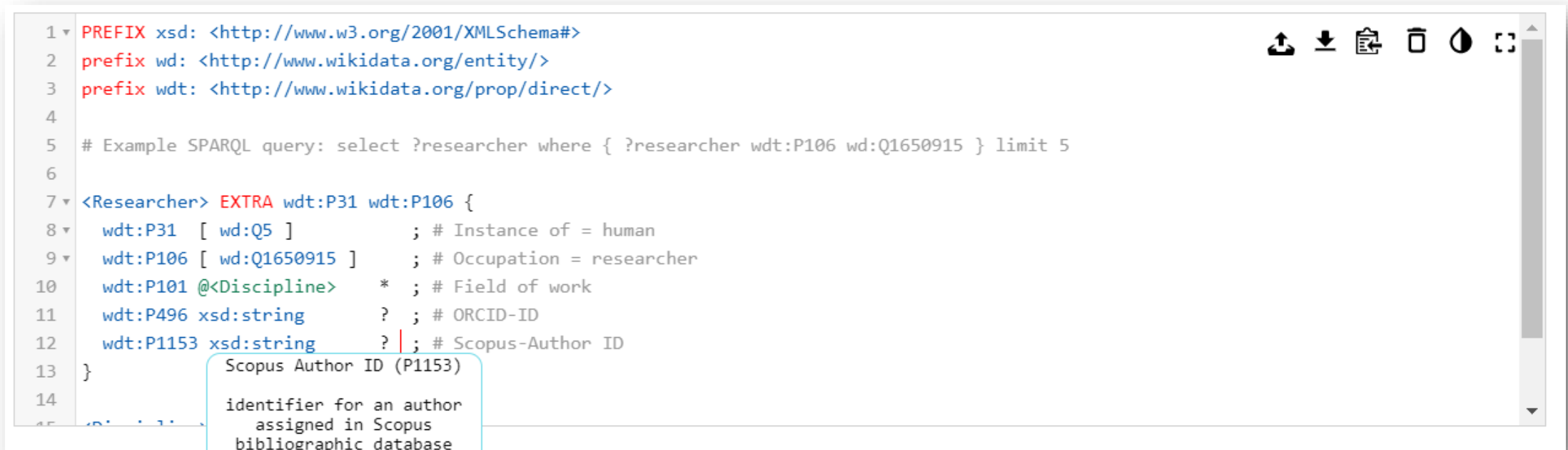


Text-based editors

YaSHE: Forked from YASGUI: <http://www.weso.es/YASHE/>

Syntax highlighting

Auto-completion



```
1 PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
2 prefix wd: <http://www.wikidata.org/entity/>
3 prefix wdt: <http://www.wikidata.org/prop/direct/>
4
5 # Example SPARQL query: select ?researcher where { ?researcher wdt:P106 wd:Q1650915 } limit 5
6
7 <Researcher> EXTRA wdt:P31 wdt:P106 {
8   wdt:P31 [ wd:Q5 ] ; # Instance of = human
9   wdt:P106 [ wd:Q1650915 ] ; # Occupation = researcher
10  wdt:P101 @<Discipline> * ; # Field of work
11  wdt:P496 xsd:string ? ; # ORCID-ID
12  wdt:P1153 xsd:string ? ; # Scopus-Author ID
13 }
14
```

Scopus Author ID (P1153)
identifier for an author
assigned in Scopus
bibliographic database

Shapes author tools: ShEx Author

ShEx-Author: Inspired by Wikidata Query Service

2 column: Visual one synchronized with text based

The screenshot displays the ShExAuthor web interface, which is divided into two main columns. The left column contains a visual editor, and the right column contains a text editor.

Visual Editor (Left Column):

- Assistant:** A sidebar on the far left with icons for information, eye, prefix, undo, redo, save, download, and other functions.
- Shape:** A section for defining shapes. It includes a dropdown for "IriRef" and a text input for "Researcher".
- Triple:** A table for defining triples. Each row has a "Prefix" dropdown, a "Property" dropdown, and a "Value" text input. The table is synchronized with the text editor on the right.
- + Triple:** A button to add a new triple.
- Shape:** A section for defining shapes. It includes a dropdown for "IriRef" and a text input for "Discipline".
- Triple:** A table for defining triples. Each row has a "Prefix" dropdown, a "Property" dropdown, and a "Value" text input. The table is synchronized with the text editor on the right.
- + Triple:** A button to add a new triple.

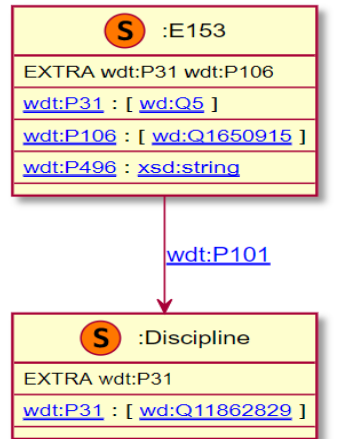
Text Editor (Right Column):

```
1 PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
2 PREFIX wd: <http://www.wikidata.org/entity/>
3 PREFIX wdt: <http://www.wikidata.org/prop/direct/>
4
5 <Researcher> {
6   wdt:P31 IRI ;
7   wdt:P106 IRI ? ;
8   wdt:P101 @<Discipline>? ;
9   wdt:P496 xsd:string ? ;
10  wdt:P1153 xsd:string * ;
11 }
12
13 <Discipline> {
14   wdt:P31 IRI * ;
15 }
16
17
```

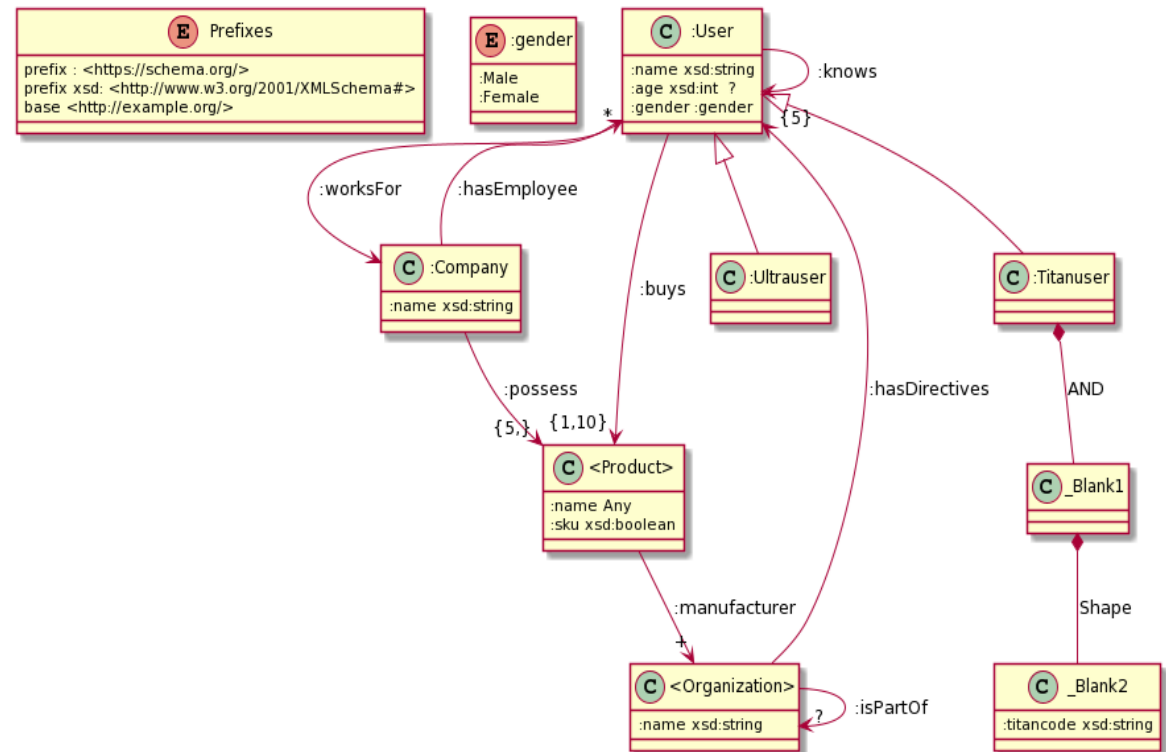

Shapes visualization

Integrated in RDFShape/Wikishape

- [UMLSHacLEX](#) UML diagrams for ShEx
- [ShUMLex](#): Conversion to UML through XMI



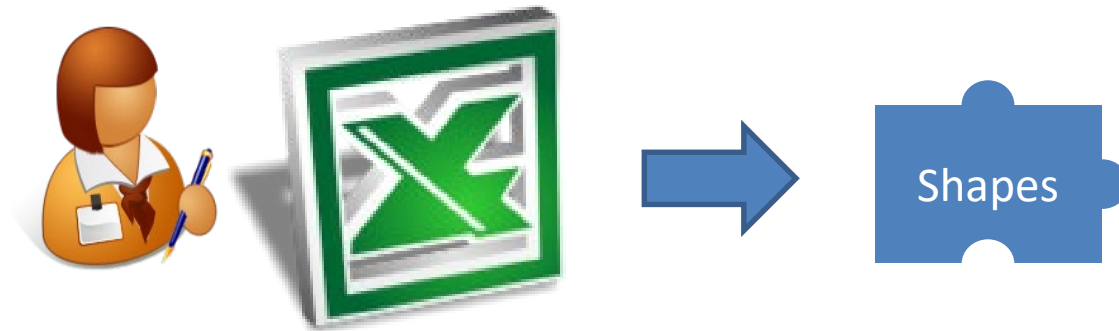
Generated by [rdfshape](#)



Shapes from spreadsheets

ShExCSV: CSV representation of Shapes

Hermes: ShExCSV processor, <https://github.com/weso/hermes>

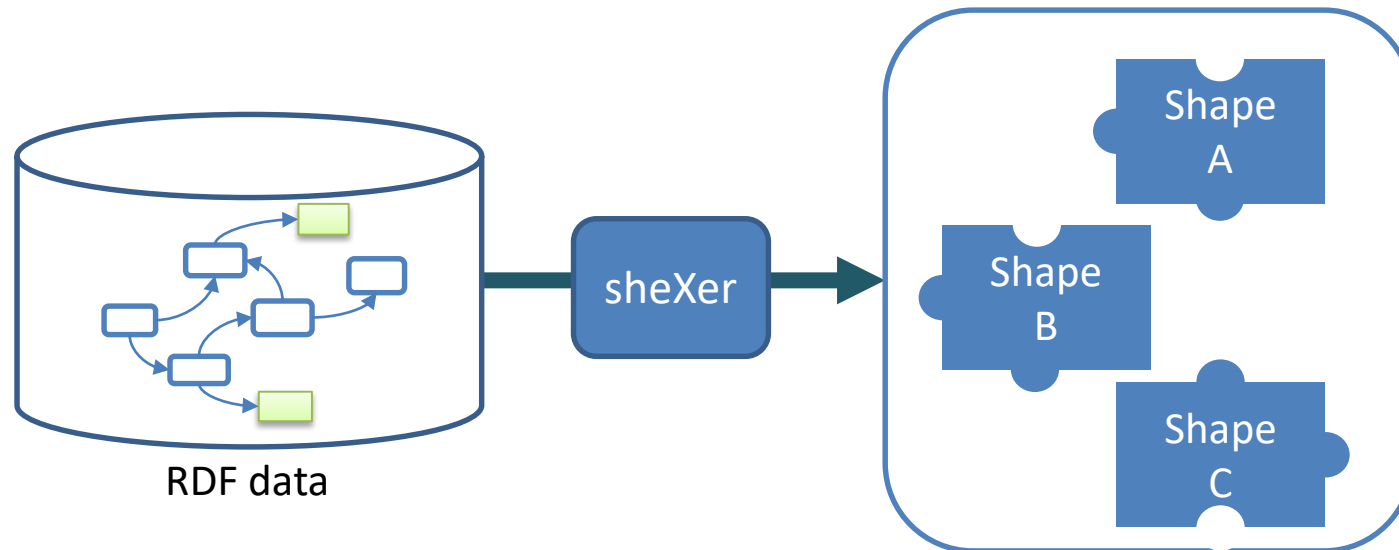


Generating Shapes from RDF data

Useful use case in practice

Some prototypes

sheXer: <http://shexer.weso.es/>

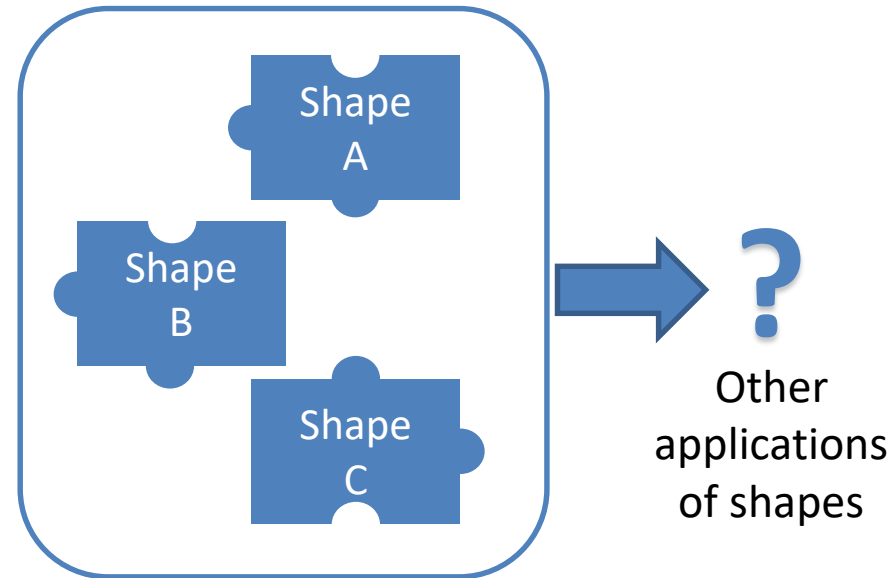


Other applications of Shapes

UIs and shapes

Generating code from Shapes

Generate subsettings



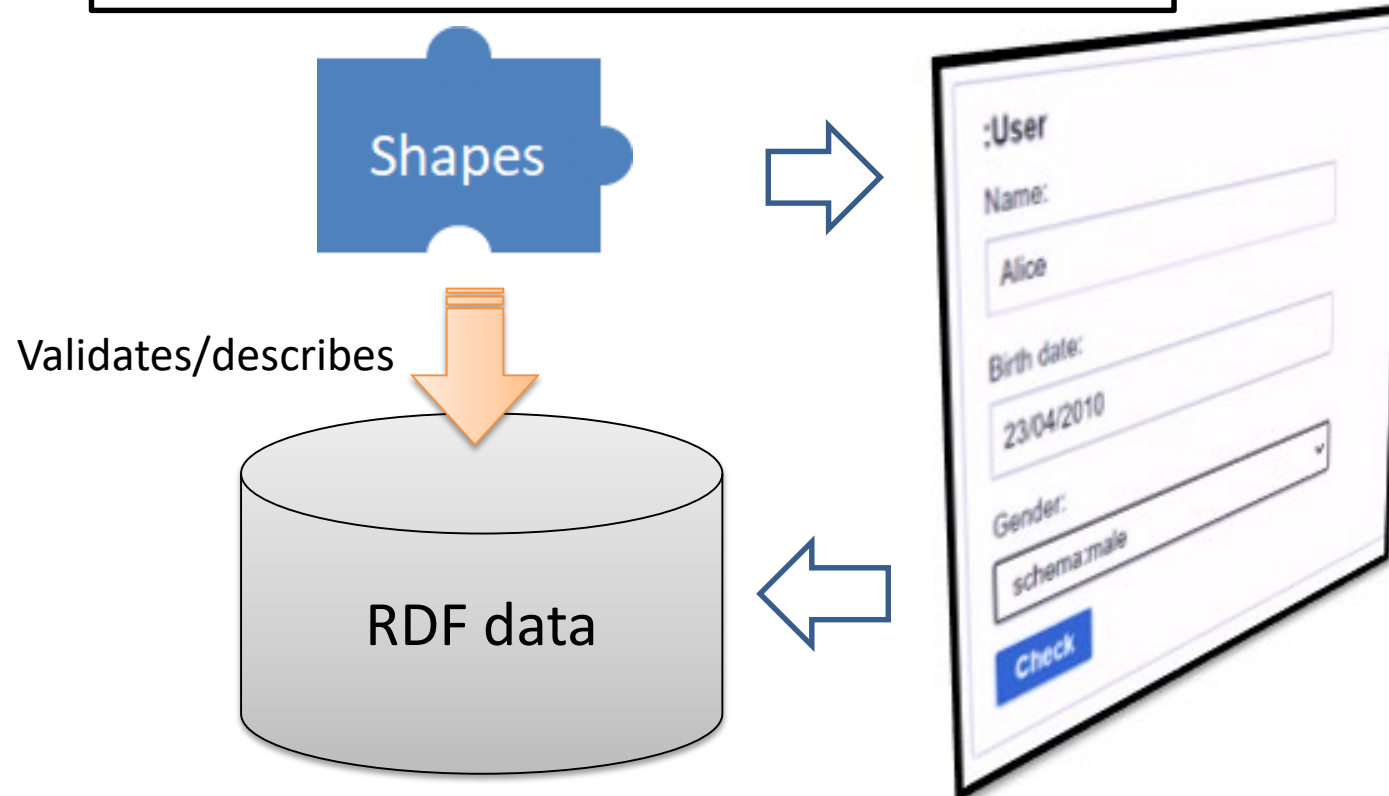
UI and shapes: ShapeForms

ShapeForms

```
prefix schema: <http://schema.org/>
prefix : <http://example.org/>
prefix xsd: <http://www.w3.org/2001/XMLSchema#>
prefix ui: <http://www.w3.org/ns/ui#>

start = @:User

:User {
  schema:name xsd:string // ui:label "name" ;
  schema:birthDate xsd:dateTime // ui:label "Birth date" ;
  schema:gender [ schema:Male schema:Female ] // ui:label "Gender" ;
}
```



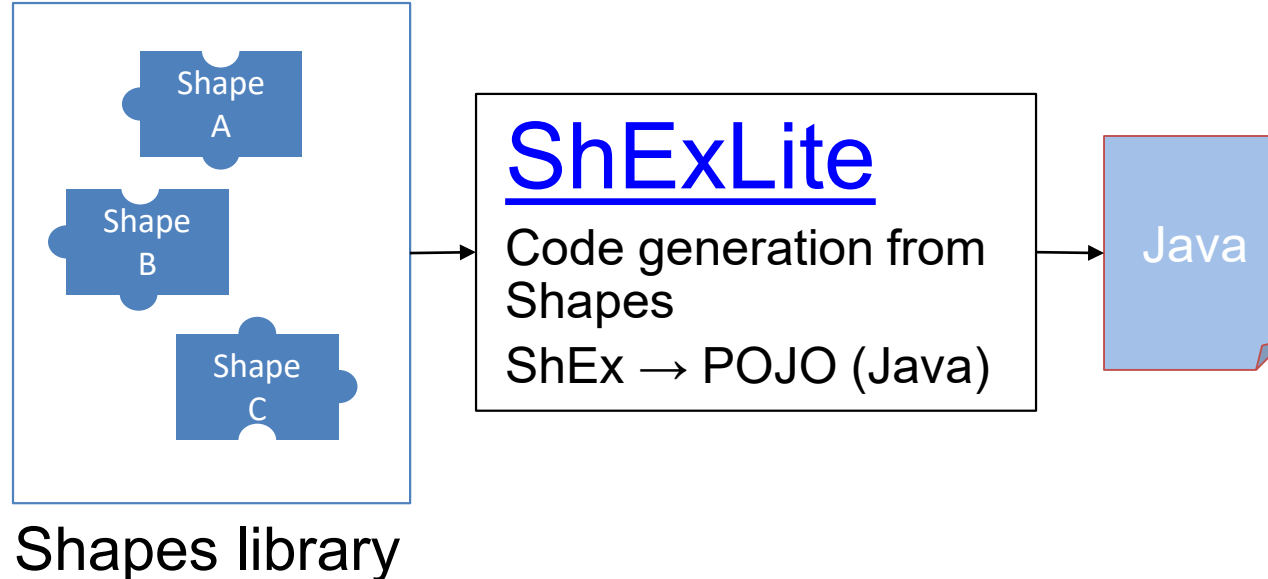
Generating code from shapes

Generate domain model from shapes

Entities (pseudo-shapes) defined with Excel (Google spreadsheets)

Shapes generation from those templates

Java code generation (POJOs) from those shapes

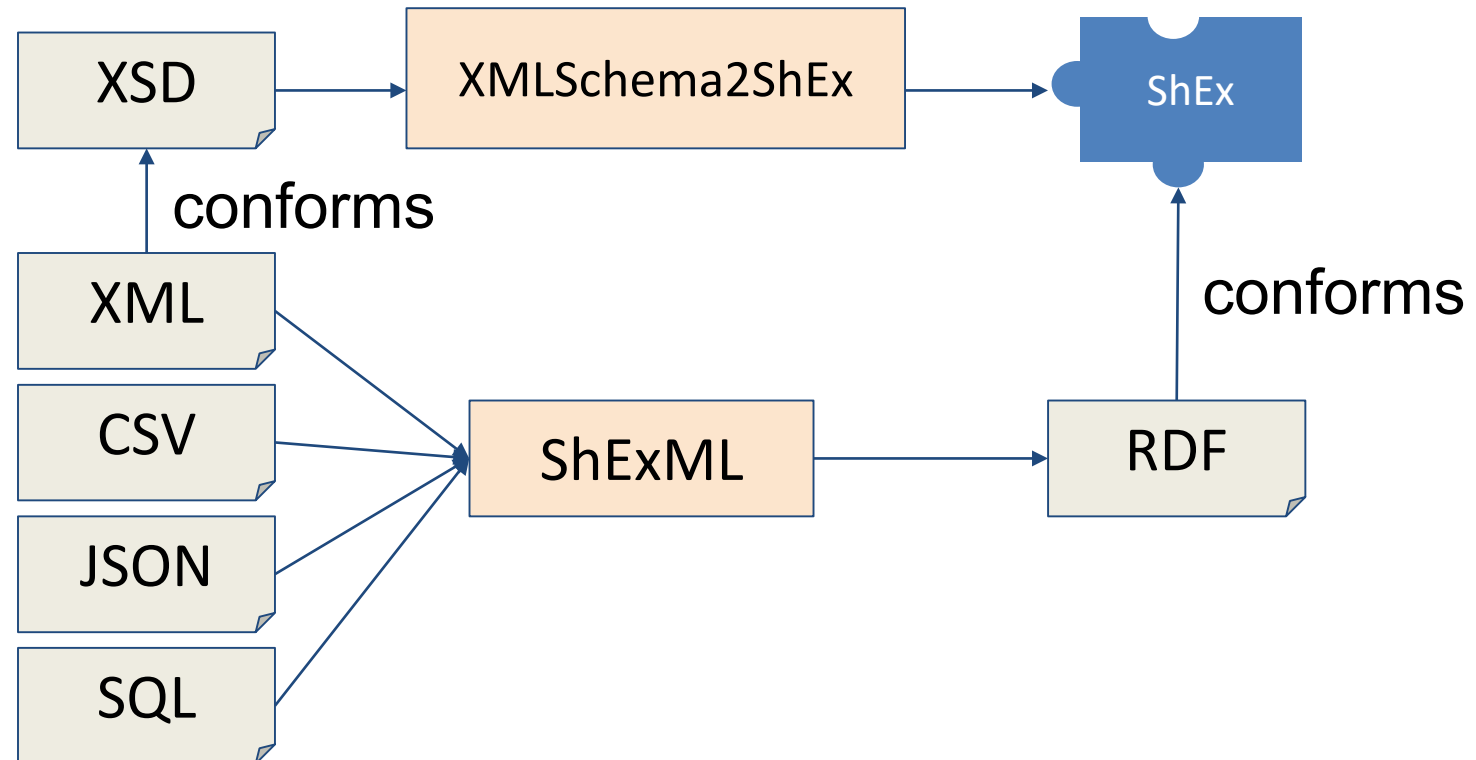


Shapes for data integration

[XMLSchema2ShEx](#): Convert XML Schemas to shapes

[ShExML](#): Domain specific language to convert data to RDF

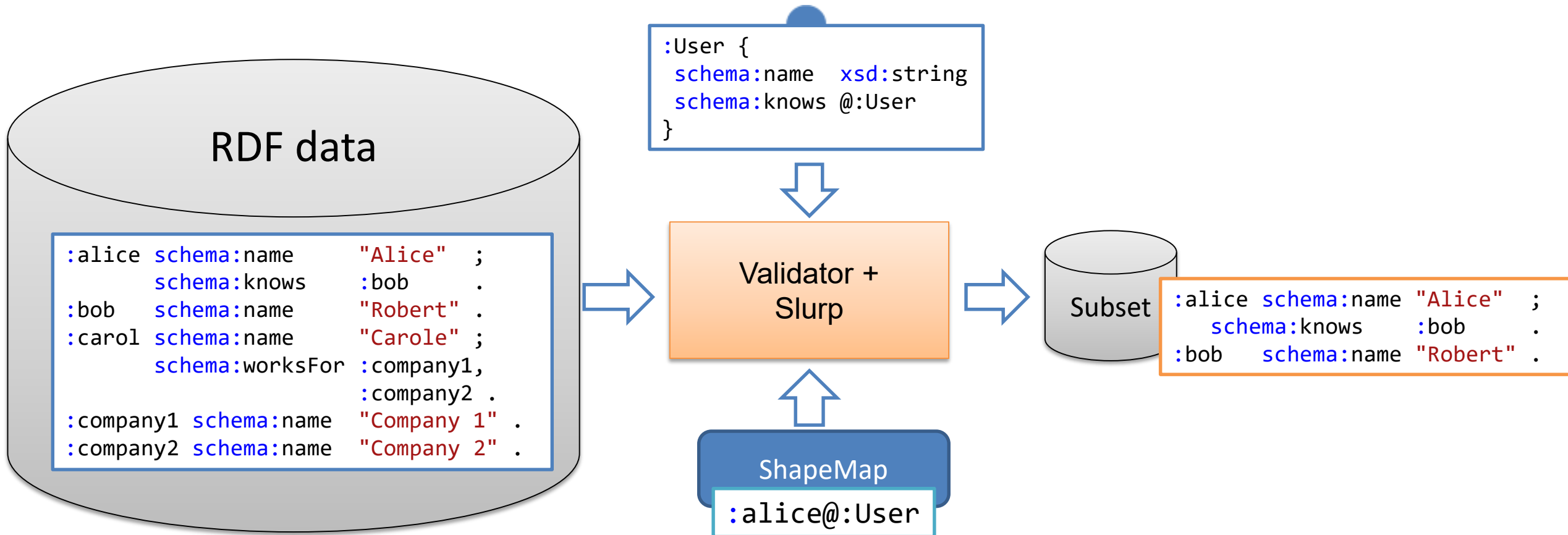
Input formats: CSV, XML, JSON, SQL



Subsetting based on Shapes

Generate subsets from ShEx

Slurp option: when validating, collect the affected nodes/triples



Shapes ecosystems

Wikidata provides a whole ShEx ecosystem

Entity schemas can evolve and relate between each other

Directory: https://www.wikidata.org/wiki/Wikidata:Database_reports/EntitySchema_directory

Different schemas for the same entities?

Some schemas stress some aspects while others stress others

Evolution of schemas

Searching entity schemas

Reasoning

RDFS

RDFS (RDF Schema) was already created in 1999

A language that can be used to describe vocabularies

It can declare

Classes (`rdfs:Class`)

Properties (`rdfs:Property`)

Resources (`rdfs:Resource`): anything

Relationships between classes and properties

`rdfs:subClassOf`, `rdfs:subPropertyOf`

`rdfs:domain`, `rdfs:range`, ...

Documentation about vocabularies

`rdfs:label`, `rdfs:comment`

RDFS: Instances, classes and properties

Difference between

Instances/individuals: `:alice`, `:cs101`, `:Oviedo`, ...

Classes/concepts: `:Person`, `:Course`, `:City`, `:Place`

Properties/relationships: `:teaches`, `:hasCountry`

`rdf:type` declares that an individual has a given type

`rdfs:subClassOf` declares that a Class is a subclass of another

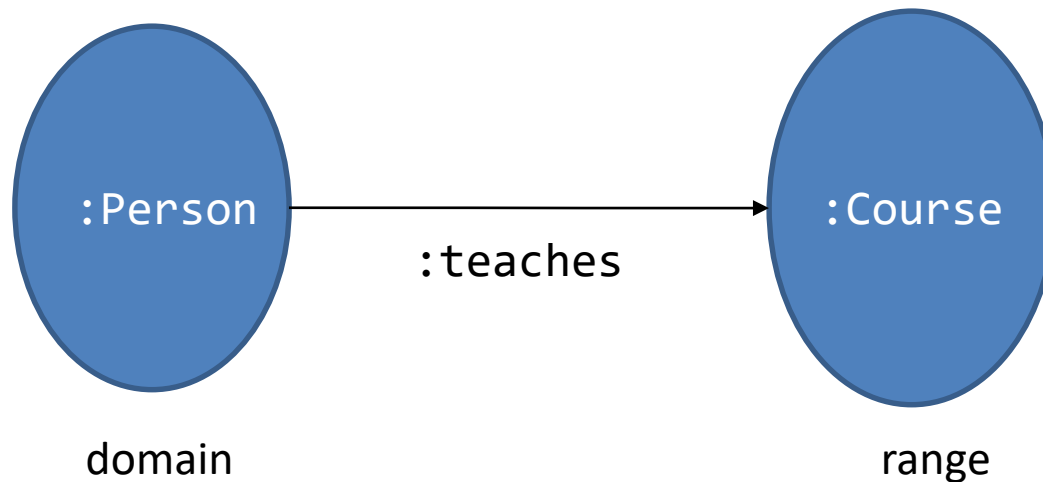
RDFS: Domain and range

It is possible to declare domain and range constraints

Example: `:teaches`

Domain: `:Person`

Range: `:Course`



RDFS: Class hierarchies

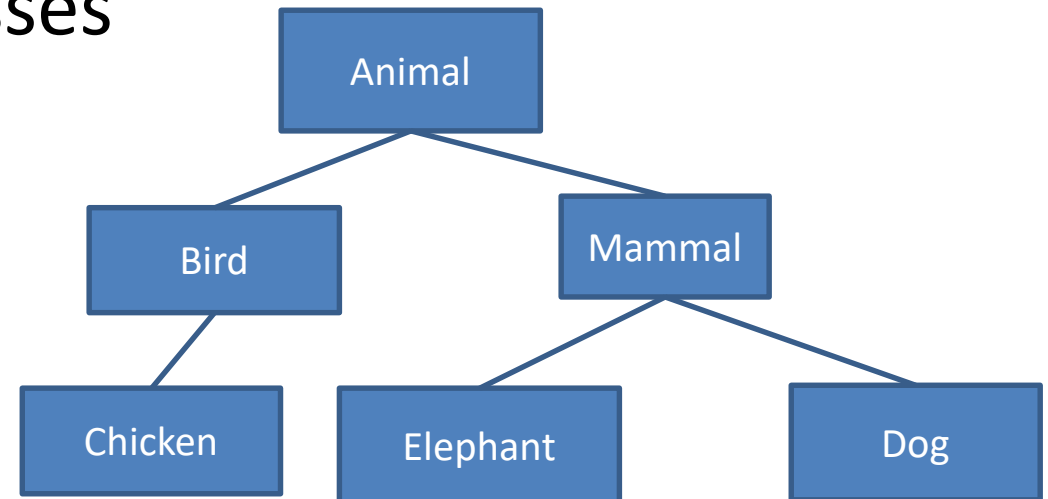
Classes can be organized in hierarchies

rdfs:subClassOf defines that a class is a subclass of another

A is a subclass of B if every individual from A is in B

In that case, B is a superclass of A

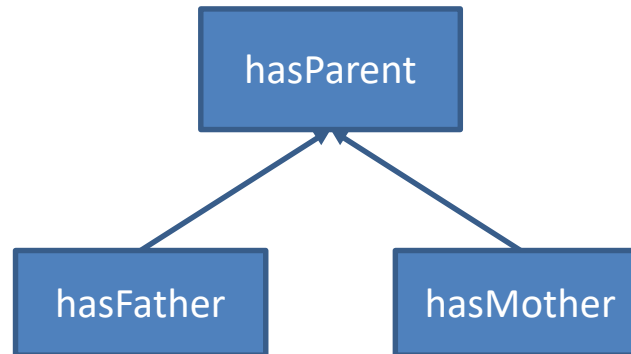
A class can have multiple superclasses



RDFS: Property hierarchies

Properties can also be organized in hierarchies

rdfs:subPropertyOf defines that a property is a property of another
p is a subProperty of q if for all x,y such that (x,p,y) then (x,q,y)



`:alice :hasFather :bob ⇒ :alice :hasParent :bob`

RDFS: inferences

RDFS has a built-in semantics which allows to infer new triples

Example rules:

$$X \text{ rdf:type } A \wedge A \text{ subClassOf } B \rightarrow X \text{ rdf:type } B$$

$$A \text{ rdfs:subClassOf } B \wedge B \text{ rdfs:subClassOf } C \rightarrow A \text{ rdfs:subClassOf } C$$

$$P \text{ rdfs:domain } A \wedge X P Y \rightarrow X \text{ rdf:type } A$$

$$P \text{ rdfs:range } B \wedge X P Y \rightarrow Y \text{ rdf:type } B$$

$$P \text{ rdfs:subPropertyOf } Q \wedge Q \text{ rdfs:subPropertyOf } R \rightarrow P \text{ rdfs:SubpropertyOf } R$$

$$P \text{ rdfs:subPropertyOf } Q \wedge X P Y \rightarrow X Q Y$$

RDFS expressivity

It can be used as a first step towards vocabulary definitions

But...

It has limited expressivity to define, for example:

Negative information: *A Person is not a Car*

Quantifiers: *Every parent must have at least one children*

Cardinality: *A good student must have passed at least 3 courses*

Attributes about properties

Example: Inverse property: If x hasParent y, then y hasChild x

Other attributes: symmetry, transitive, etc.

It can also be too liberal

You can declare, in RDFS:

```
:Person rdfs:type :Person
```

OWL

OWL = Web Ontology Language

Developed as part of W3C initiative

Antecedents: DAML, OIL

Based on Description Logics

2004 - OWL 1.0 W3C recommendation

2009 - OWL 2.0 W3C recommendation



OWL expressivity

OWL adds a lot of expressivity

More expressivity \Rightarrow More complexity

Several levels

OWL Full: No restrictions

It can be undecidable

OWL DL: Decidable but has NPTIME Complexity

Several profiles

OWL EL, OWL QL, OWL RL, ...

More efficient, less expressive

<http://www.cs.man.ac.uk/~ezolin/dl/>

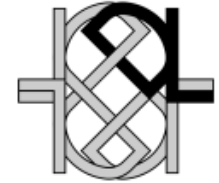
Expressiveness vs Complexity



OWL: description logics

OWL DL is based on description logics

It corresponds to the family: $\mathcal{SHOIN}(\mathcal{D}_n)$



<http://dl.kr.org/>

Description logics is a subset of first order predicate logic

Goal: restrict expressivity to obtain decidability

Predicates with one or two arguments

Person(alice)

Teaches(alice,cs101)

The use of variables is restricted

OWL: TBox and ABox

TBox (Terminological) definitions

$\text{Parent} \equiv \text{Person} \cap \exists \text{ hasChild Person}$

$\text{Proud} \equiv \text{Person} \cap \exists \text{ hasChild NewBorn}$

$\text{NewBorn} \subseteq \text{Person}$

ABox (Assertional) definitions

$\text{NewBorn}(\text{bob})$

$\text{hasChild}(\text{alice}, \text{bob})$

$\text{Person}(\text{alice})$

OWL and predicate logic

Description logic definitions can be translated for predicate logic

Parent \equiv Person $\cap \exists$ hasChild Person

$$\forall x(\text{Padre}(x) \leftrightarrow (\text{Person}(x) \wedge \exists y(\text{hasChild}(x,y) \wedge \text{Person}(y))))$$

Proud \equiv Person $\cap \exists$ hasChild NewBorn

$$\forall x(\text{Proud}(x) \leftrightarrow (\text{Person}(x) \wedge \exists y(\text{hasChild}(x,y) \wedge \text{NewBorn}(y))))$$

NewBorn \subseteq Person

$$\forall x(\text{NewBorn}(x) \rightarrow \text{Person}(x))$$

OWL: concept definition

Equivalence: $C \equiv D$

Example: $\text{Spaniard} \equiv \text{SpanishNationality}$

Subclass: $C \subseteq D$ (C is a subset of D)

Example: $\text{Spaniard} \subseteq \text{European}$

Intersection: $C \cap D$

Example: $\text{Woman} \equiv \text{Person} \cap \text{Female}$

Union: $C \cup D$

Example: $\text{Person} \equiv \text{Man} \cup \text{Woman}$

Complement: $\neg C$

Example: $\text{Male} \equiv \neg \text{Female}$

Empty concept: \perp

Disjoint classes $C \cap D \equiv \perp$

OWL: quantifiers

Existential (\exists R C)

x belongs to \exists R C if there's some $y \in C$ such that $R(x,y)$

Example: $\text{Mother} \equiv \text{Woman} \cap \exists \text{ hasChild Person}$

Universal (\forall R C)

x belongs to \forall R C if for all y, if $R(x,y)$ then $y \in C$

Example: $\text{HappyWoman} \equiv \text{Mother} \cap \forall \text{ hasChild Healthy}$

A mother is happy if all her children are healthy

OWL: cardinalities

Cardinality ($P = n$)

x belongs to ($P = n$) if there are n $y \in C$ such that $R(x,y)$

Example: $\text{Elephant} \subseteq \text{Animal} \cap \text{hasLegs} = 4$

Max. cardinality ($P \leq n$)

x belongs to ($P \leq n$) if there are n or less $y \in C$ such that $R(x,y)$

Example: $\text{BadStudent} \equiv \text{Student} \cap \text{passesCourse} \leq 3$

Min. cardinality ($P \geq n$)

x belongs to ($P \geq n$) if there are n or more $y \in C$ such that $R(x,y)$

Example: $\text{GoodStudent} \equiv \text{Student} \cap \text{passesCourse} \geq 3$

$\forall x(\text{GoodStudent}(x) \leftrightarrow \text{Student}(x) \wedge$
 $\quad \exists y_1 \exists y_2 \exists y_3 (\text{passesCourse}(x, y_1) \wedge \text{passesCourse}(x, y_2) \wedge \text{passesCourse}(x, y_3) \wedge y_1 \neq y_2 \wedge y_2 \neq y_3 \wedge y_1 \neq y_3))$

OWL: property attributes

Reflexive: P is reflexive $\equiv \forall x P(x,x)$

Example: `livesWith` is reflexive

Irreflexive: P is irreflexive $\equiv \forall x \neg P(x,x)$

Example: `hasParent` is irreflexive

Symmetry. If $P(x,y)$ then $P(y,x)$

Example: `sibling`

Asymmetry. If $P(x,y)$ then $\neg P(y,x)$

Example: `hasParent`

Transitivity. If $P(x,y)$ and $P(y,z)$ then $P(x,z)$

Example: `sibling`

OWL: property relationships

Inverse: P is inverse of Q $\equiv P(x,y) \Leftrightarrow Q(y,x)$

Example: `hasParent` is inverse of `hasChild`

Subproperty: P is a subproperty of Q $\equiv P(x,y) \Rightarrow Q(x,y)$

Example: `hasChild` is a subproperty of `hasDescendant`

OWL: Functional properties

Functional property. $P(x,y)$ and $P(x,z) \Rightarrow y = z$

Example: `birthPlace`

Inverse functional property.

$P(x,y)$ and $P(z,y) \Rightarrow x = z$

Example: `ssn`

Keys. similar to inverse functional properties but specific for a class

$P(x,y)$ and $P(z,y) \Rightarrow x = z$.

Example: `ssn`

OWL in practice

Lots of ontologies have been defined

Tools: Protégé, TopBraid Composer, ...

Foundries: OBOFoundry,

Ontology Engineering development

Gradually adopting software engineering techniques like TDD

OWL and reasoning

OWL reasoning can be too complex in practice

Ongoing discussions about use cases for OWL, Rules, Shapes,....

Trends in research data

Share and reuse research data

Reproducibility of experiments?

Represent workflows and pipelines?

Research is more than a PDF paper

Some initiatives:

FAIR principles

Nanopublications

Research Objects

FAIR principles

(F)indable

- F1. (Meta)data are assigned a globally unique and persistent identifier
- F2. Data are described with rich metadata (defined by R1 below)
- F3. Metadata clearly and explicitly include the identifier of the data they describe
- F4. (Meta)data are registered or indexed in a searchable resource

<https://www.go-fair.org/fair-principles/>

(A)ccessible

- A1. (Meta)data are retrievable by their identifier using a standardised communications protocol
 - A1.1 The protocol is open, free, and universally implementable
 - A1.2 The protocol allows for an authentication and authorisation procedure, where necessary
- A2. Metadata are accessible, even when the data are no longer available

(I)nteroperable

- I1. (Meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation.
- I2. (Meta)data use vocabularies that follow FAIR principles
- I3. (Meta)data include qualified references to other (meta)data

(R)eusable

- R1. (Meta)data are richly described with a plurality of accurate and relevant attributes
 - R1.1. (Meta)data are released with a clear and accessible data usage license
 - R1.2. (Meta)data are associated with detailed provenance
 - R1.3. (Meta)data meet domain-relevant community standards

Nanopublications

Smallest unit of publishable information

About any topic

Fully expressible in machine-interpretable way (RDF)

3 parts

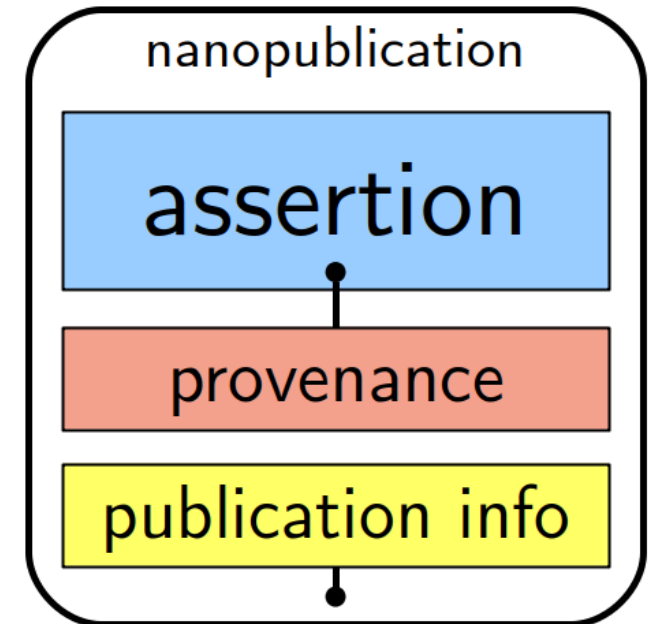
Assertion: main content

Provenance: how the assertion came to be

Example: Scientific methods employed

Publication info: Metadata about nanopublication as a whole

Example: When? By whom?



Nanopublications

Example using TriG

```
@prefix : <http://example.org/pub1/> .
@prefix ex: <http://example.org/> .
@prefix np: <http://www.nanopub.org/nschema#> .
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix dct: <http://purl.org/dc/terms/> .
@prefix orcid: <https://orcid.org/> .

:Head {
  : a np:Nanopublication ;
    np:hasAssertion :assertion ;
    np:hasProvenance :provenance ;
    np:hasPublicationInfo :pubinfo
}
:assertion {
  ex:oviedo ex:hasTemperature 36 .
}
:provenance {
  :assertion prov:wasDerivedFrom :slides
}
:pubinfo {
  : dct:creator orcid: 0000-0001-8907-5348 ;
    dct:created "2021-12-01T07:51:00"^^xsd:dateTime .
}
```

Nanopublications in practice

Nanopub servers: Publish and search nanopublications

Integrity keys can be used to check immutability

Trusty URIs = include hash values in URIs

Example: http://server.nanopubs.lod.labs.vu.nl/RAn15vsPJEVdJvjNKtBPo_oadtjeP9oc3Si-69FiJ4poQ

Nanopub monitor:

<http://app.tkuhn.eculture.labs.vu.nl/nanopub-monitor/>

FAIR principles as a nanopub: <https://github.com/peta-pico/FAIR-nanopubs/blob/master/principles.trig>

Research Objects

Support publication of:

- More than just PDF, example: data, code, etc.

- Collection of resources

- Annotations about resources

Core principles: identity, aggregation, annotation

Specification: RO-Crate (<https://www.researchobject.org/ro-crate>)

- Based on JSON-LD

<https://www.researchobject.org/>

Myths about Semantic Web, linked data, KGs...

Too expensive

It's free

No one will want our data

It is not trendy

Too much openness

Our project will be a success

It is too expensive

Not really

It is possible to learn from previous experiences

Follow the lemma:

Separate content from presentation

Content: Information/data

Presentation: visual/aesthetic aspects

Try not to lose semantics



Content



Presentation

It is free

...well, no

It requires to complement with visualizations

Only data = too restrained

Requires to define data models and URIs

Stable and cool URIs

Challenge of continuous updates

Take care of data pipelines



No one will be interested in our data

Just the opposite...our data = our treasure

Search engines index semantic content

schema.org Project (Google, Bing, Yandex,...)

If we help them \Rightarrow better SEO

Automatic processable data = more valuable data

Promote data culture

New business opportunities and applications

Government = catalyzer: Hackathons y similars...



It is not trendy...

Beware of trends in computer science...

Knowledge graphs seem to be trendy

But...lots of technologies appear/dissapear

Is Semantic web trendy? And linked data?

Hype Cycle for Artificial Intelligence, 2020



Too much open

If we really believe in transparency...

We will pursue reusable data

Nevertheless...

It is useful to distinguish between:

Open data

Linked data

Public/private data

Aggregated data

Partially open data

Linked and closed data

FAIR data

...



Backup slides

OWL: Ontologies

Declare classes, properties and entities in a domain

Focus on the domain model

Knowledge representation

Ontologies enable reasoning and classification

Open World Assumption

We assume there are a lot of knowledge we may not have yet



Rules

Declare conditions IF....THEN....

Focus on the domain model

Knowledge representation

Premises/conclusions

Open/Closed World Assumption

Both assumptions can be used depending on the domain

