Validating RDF data tutorial
ShEx/SHACL by example

Presenters:

Jose Emilio Labra Gayo
WESO Research group, Spain

Iovka Boneva
LINKS, INRIA & CNRS, France

Collaborators:

Eric Prud'hommeaux
World Wide Web Consortium
MIT, Cambridge, MA, USA

Dimitris Kontokostas
GeoPhy
http://kontokostas.com/
More info

HTML version: http://book.validatingrdf.com
Examples: https://github.com/labra/validatingRDFBookExamples
Contents

RDF data model (short overview)
Motivation for RDF data Validation
ShEx by example
SHACL by example
Comparing ShEx and SHACL
Applications
RDF Data Model

Overview of RDF Data Model and simple exercise

Link to slides about RDF Data Model
https://figshare.com/articles/RDF_data_model/7159796
RDF, the good parts...

RDF as an integration language
RDF as a *lingua franca* for semantic web and linked data
RDF data stores & SPARQL
RDF flexibility
  - Data can be adapted to multiple environments
  - Open and reusable data by default
RDF, the other parts

Inference & knowledge representation

RDF should combine well with KR vocabularies (RDF Schema, OWL...)
Performance of RDF based systems with inference = challenging

Consuming & producing RDF

Multiple serializations: Turtle, RDF/XML, JSON-LD, ...
Embedding RDF in HTML
Describing and validating RDF content
Why describe & validate RDF?

For producers
   Developers can understand the contents they are going to produce
   They can ensure they produce the expected structure
   Advertise and document the structure
   Generate interfaces

For consumers
   Understand the contents
   Verify the structure before processing it
   Query generation & optimization
### Similar technologies

<table>
<thead>
<tr>
<th>Technology</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relational Databases</td>
<td>DDL</td>
</tr>
<tr>
<td>XML</td>
<td>DTD, XML Schema, RelaxNG, Schematron</td>
</tr>
<tr>
<td>Json</td>
<td>Json Schema</td>
</tr>
<tr>
<td>RDF</td>
<td>?</td>
</tr>
</tbody>
</table>

Fill that gap
Understanding the problem

RDF is composed by nodes and arcs between nodes

We can describe/check

The form of the node itself (node constraint)

The number of possible arcs incoming/outgoing from a node

The possible values associated with those arcs

RDF Node

:alice schema:name "Alice";
  schema:knows :bob .

IRI schema:name string 1
  schema:knows IRI 0, 1,...

Shape

RDF Node that represents a User

<UserShape> IRI {
  schema:name xsd:string ;
  schema:knows IRI *
}

ShEx
Understanding the problem

RDF flexibility

Mixed use of objects & literals

Example:

Values of schema:creator can be:

string or

schema:Person

in the same data

Lots of examples at http://schema.org
Understanding the problem

Repeated properties
The same property can be used for different purposes in the same data
Example: A product must have 2 codes with different structure

```turtle
schema:productID "code456" .
```

A practical example from FHIR
See: http://hl7-fhir.github.io/observation-example-bloodpressure.ttl.html
Understanding the problem

Shapes ≠ types

Nodes in RDF graphs can have zero, one or many `rdf:type` declarations

One type can be used for multiple purposes (`foaf:Person`)

Data doesn't need to be annotated with fully discriminating types

`foaf:Person` can represent friend, invitee, patient, ...

Different meanings and different structure depending on the context

Specific validation constraints for different contexts
Understanding the problem

RDF validation ≠ ontology definition ≠ instance data

Ontologies are usually focused on real world entities

RDF validation is focused on RDF graph features (lower level)

Different levels

- **Ontology**
  ```
  schema:knows a owl:ObjectProperty ;
  rdfs:domain schema:Person ;
  rdfs:range schema:Person .
  ```

- **Constraints**
  A user must have only two properties:
  ```
  schema:name of value xsd:string
  schema:knows with an IRI value
  ```

- **RDF Validation**
  ```
  <User> IRI {
    schema:name xsd:string ;
    schema:knows IRI
  }
  ```

- **Instance data**
  ```
  :alice schema:name "Alice";
  schema:knows :bob .
  ```
Previous RDF validation approaches

SPARQL based
  Plain SPARQL
  SPIN: http://spinrdf.org/

OWL based
  Stardog ICV
    http://docs.stardog.com/icv/icv-specification.html

Grammar based
  OSLC Resource Shapes
    https://www.w3.org/Submission/2014/SUBM-shapes-20140211/
Define SPARQL queries that detect errors

Pros:
Expressive
Ubiquitous

Cons
Expressive
Idiomatic - many ways to encode the same constraint

Example: SPARQL query to check that...
There is one schema:name which must be a xsd:string and one schema:gender must be schema:Male or schema:Female

```sparql
ASK {{
  SELECT ?Person {
    ?Person schema:name ?o .
  } GROUP BY ?Person HAVING (COUNT(*)=1)
}

SELECT ?Person {
  ?Person schema:name ?o .
  FILTER (isLiteral(?o) &&
          datatype(?o) = xsd:string)
} GROUP BY ?Person HAVING (COUNT(*)=1)

SELECT ?Person (COUNT(*) AS ?c1) {
} GROUP BY ?Person HAVING (COUNT(*)=1)}

SELECT ?Person (COUNT(*) AS ?c2) {
  FILTER ((?o = schema:Female ||
           ?o = schema:Male))
} GROUP BY ?Person HAVING (COUNT(*)=1)}
FILTER (?c1 = ?c2)
```
SPARQL inferencing notation [http://spinrdf.org/](http://spinrdf.org/)

Developed by TopQuadrant

Commercial product

Vocabulary associated with user-defined functions in SPARQL

SPIN has influenced SHACL (see later)
Stardog ICV

ICV - Integrity Constraint Validation

Commercial product

OWL with unique name assumption and closed world

Compiled to SPARQL

More info: http://docs.stardog.com/icv/icv-specification.html
OSLC Resource Shapes

Grammar based approach
Language for RDF validation
Input for ShEx and SHACL

```r
:user a rs:ResourceShape ;
rs:property [ 
  rs:name "name" ;
  rs:propertyDefinition schema:name ;
  rs:valueType xsd:string ;
  rs:occurs rs:Exactly-one ;
] ;
rs:property [ 
  rs:name "gender" ;
  rs:propertyDefinition schema:gender ;
  rs:allowedValue schema:Male, schema:Female ;
  rs:occurs rs:Zero-or-one ;
].
```
Other approaches

Dublin Core Application profiles (K. Coyle, T. Baker)
  http://dublincore.org/documents/dc-dsp/

RDF Data Descriptions (Fischer et al)

RDFUnit (D. Kontokostas)
  http://aksw.org/Projects/RDFUnit.html

...
ShEx and SHACL

2013 RDF Validation Workshop

   Conclusions of the workshop:
       *There is a need of a higher level, concise language for RDF Validation*  
       ShEx initially proposed (v 1.0)

2014 W3c Data Shapes WG chartered

2017 SHACL accepted as W3C recommendation

2017 ShEx 2.0 released as Community group draft
Continue this tutorial with...

ShEx by example

SHACL by example

ShEx and SHACL compared

Applications and future work

https://figshare.com/articles/ShExByExample_pptx/6291464

https://figshare.com/articles/SHACL_by_example/6449645

https://figshare.com/articles/ShEx_and_SHACL_compared/6449648

https://figshare.com/articles/Applications_and_future_work_validating_RDF_data/7159835