RDF Validation tutorial

ShEx/SHACL by example

Jose Emilio Labra Gayo
WESO Research group
Spain

Eric Prud'hommeaux
World Wide Web, USA

Harold Solbrig
Mayo Clinic, USA

Iovka Boneva
LINKS, INRIA & CNRS, France

Contents

Overview of RDF data model

Motivation for RDF Validation and previous approaches

ShEx by example

SHACL by example

ShEx vs SHACL

RDF Data Model

Overview of RDF Data Model and simple exercise

Link to slides about
RDF Data Model → http://www.slideshare.net/jelabra/rdf-data-model

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 RDF producers
  - Developers can understand the contents they are going to produce
  - They can ensure they produce the expected structure
  - Advertise the structure
  - Generate interfaces
- For RDF consumers
  - Understand the contents
  - Verify the structure before processing it
  - Query generation & optimization

Similar technologies

Technology	Schema
Relational Databases	DDL
XML	DTD, XML Schema, RelaxNG
Json	Json Schema
RDF	?

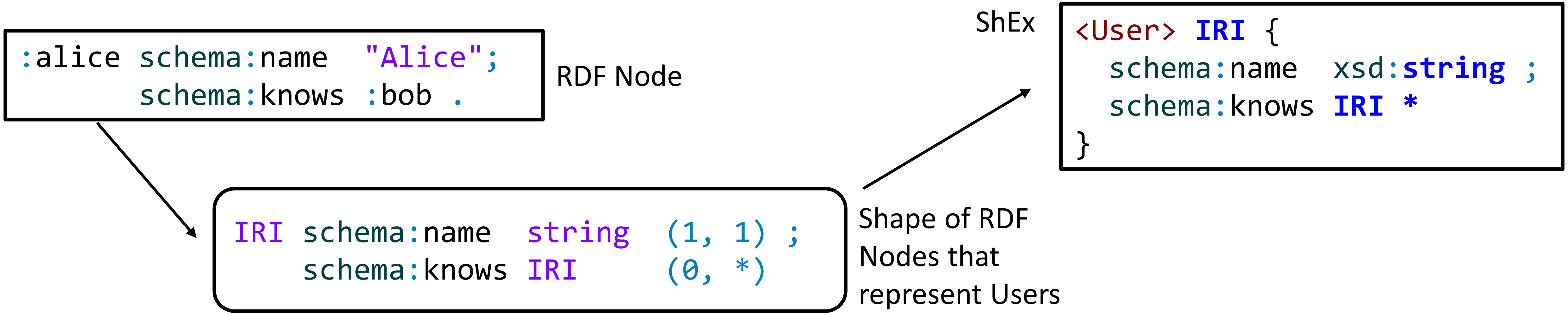
Our goal is to fill that gap

Understanding the problem

RDF is composed by nodes and arcs between nodes

We can describe/check

- form of the node itself (node constraint)
- number of possible arcs incoming/outgoing from a node
- possible values associated with those arcs



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

RDF Node

```
IRI schema:name   string   (1, 1) ;
    schema:knows  IRI      (0, *)
```

Shape of RDF
Nodes that
represent Users

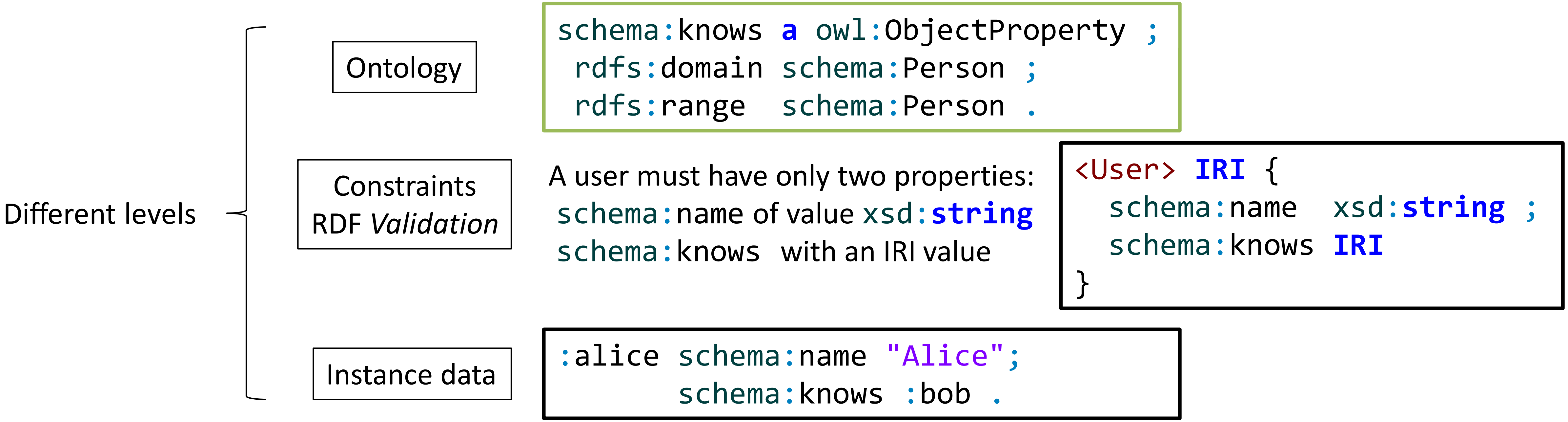
ShEx

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

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
RDF *Validation*

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

```
<User> IRI {
  schema:name   xsd:string ;
  schema:knows  IRI
}
```

Instance data

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

Understanding the problem

Shapes ≠ types

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

- 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
    - We should be able to define specific validation constraints in different contexts

Understanding the problem

RDF flexibility

- Mixed use of objects & literals
  - schema:creator can be a **string** or schema:Person in the same data

```
:angie schema:creator "Keith Richards" ,
       [ a schema:Person ;
           schema:singleName "Mick" ;
           schema:lastName "Jagger"
       ] .
```

See other examples from http://schema.org

Understanding the problem

Repeated properties

Sometimes, the same property is used for different purposes in the same data

Example: A book record must have 2 codes with different structure

```
:book schema:productID "isbn:123-456-789";
      schema:productID "code456" .
```

A practical example from FHIR

See: http://hl7-fhir.github.io/observation-example-bloodpressure.ttl.html

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/

Use SPARQL queries to detect errors

Pros:

- Expressive
- Ubiquitous

Cons

- Expressive
- Idiomatic - many ways to encode the same constraint

Example:

schema:name must be a xsd:string

schema:gender must be schema:Male or schema:Female

```
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) {
      ?Person schema:gender ?o .
    } GROUP BY ?Person HAVING (COUNT(*)=1)}
    { SELECT ?Person (COUNT(*) AS ?c2) {
      ?S schema:gender ?o .
      FILTER ((?o = schema:Female ||
               ?o = schema:Male))
    } GROUP BY ?Person HAVING (COUNT(*)=1)}
    FILTER (?c1 = ?c2)
}
```

SPIN

SPARQL inferencing notation 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

OSLC Resource Shapes

- OSLC Resource Shapes
  - Grammar based approach
  - Language for RDF validation
  - Less expressive than ShEx

```
: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/profile-guidelines/

RDF Data Descriptions (Fischer et al)

RDFUnit (D. Kontokostas)

...

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 by Eric Prud'hommeaux
- 2014 W3c Data Shapes WG chartered
- 2015 SHACL as a deliverable from the WG

Continue this tutorial with...

ShEx by example	http://www.slideshare.net/jelabra/shex-by-example
SHACL by example	http://www.slideshare.net/jelabra/shacl-by-example
ShEx vs SHACL	http://www.slideshare.net/jelabra/shex-vs-shacl
Future work and applications	http://www.slideshare.net/jelabra/rdf-validation-future-work-and-applications