Validating Semantic Knowledge Graphs using SHACL

By Veronika Heimsbakk

Validating Semantic Knowledge Graphs using SHACL

Veronika Heimsbakk

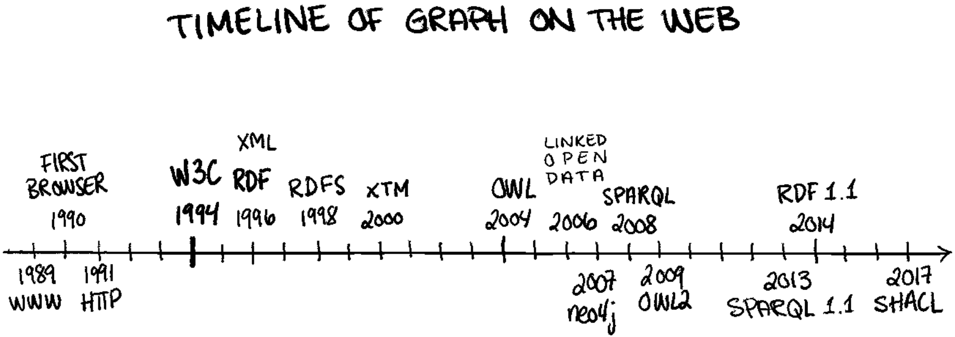
Capgemini

- vheimsbakk
- veleda
- veronikaheim
- veronahe.no

Agenda

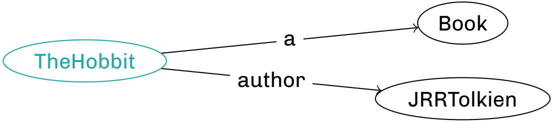
- Lecture
- Live coding

Once upon a time...



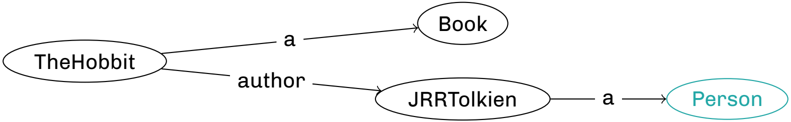
domain & range

domain



```
:author a rdf:Property ;
  rdfs:domain :Book .
```

range



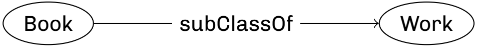
```
:author a rdf:Property ;
  rdfs:domain :Book ;
  rdfs:range  :Person .
```

TBox & ABox

TBox

TBox (terminological component)

- Sets of individuals.
- Classes and concepts.



Book

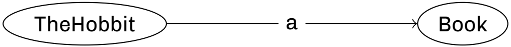
subClassOf

Work

ABox

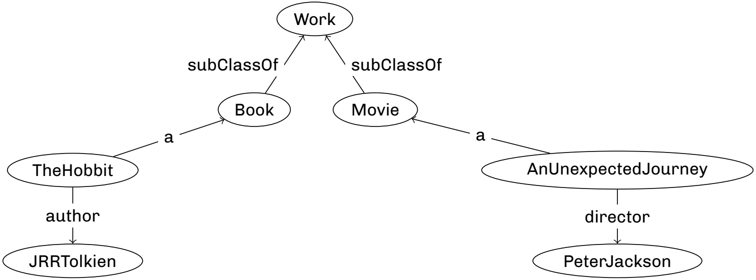
ABox (assertion component)

- Individuals and instances.



Knowledge graph

TBox + ABox = Knowledge graph ♥



World assumptions

Open world assumption (OWA)

- Admits incomplete knowledge.
- Ontologies with Web Ontology Language (OWL).

The assumption that the truth value of a statement may be true irrespective of whether or not it is known to be true.

Example

Statement: In a hole in the ground there lived a hobbit.
Question: Do Gandalf live in a hole in the ground?
OWA: Unknown

Closed world assumption (CWA)

- Shape constraints with Shape Constraint Language (SHACL).

Any statement that is true is known to be true. What is not currently known to be true is false.

Example

Statement: In a hole in the ground there lived a hobbit.
Question: Do Gandalf live in a hole in the ground?
CWA: No

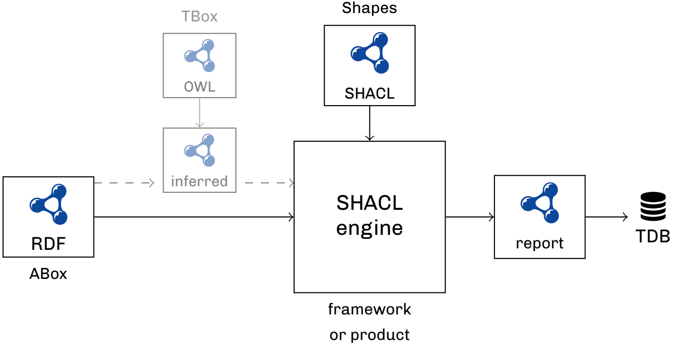
Shape Constraint Language

A language for describing and validating RDF graphs

Validation of RDF – a brief history

- Prior to SHACL; no W3C standard for validating RDF.
- SPARQL Inferencing Notation (SPIN), IBM Resource Shapes, Shape Expressions (ShEx)
- W3C recommendation in July 2017.

Workflow


TBox
OWL
inferred
Shapes
SHACL
RDF
ABox
SHACL engine
framework
or product
report
TDB

Comparing SHACL and OWL

Common

- RDF & URIs
- Infer new triples
- Rely on RDF Schema (RDFS)

Difference

OWL	**SHACL**
Designed for inference	Designed for validation
Open world assumption	Closed world assumption
Limited vocabulary	Backed by SPARQL → extensible
	SHACL resourses are distinct by default
Logical contradictions	Conforms to given schema

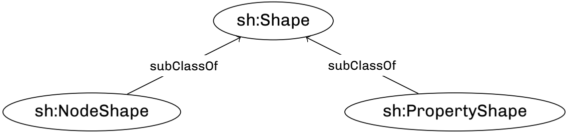
When to use SHACL?

Concept modelling (TBox)	OWL or SHACL
Instance data (ABox)	SHACL constraints
Need for inference?	OWL (or SHACL)
Knowledge graph (TBox + ABox)	OWL, then SHACL

SHACL Shape

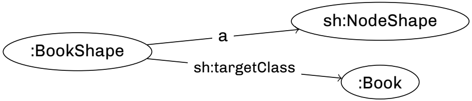
A collection of constraints for given RDF resource.

- Shapes about focus nodes (**sh:NodeShape**).
- Shapes about values of a property or path for the focus node (**sh:PropertyShape**).



sh:NodeShape

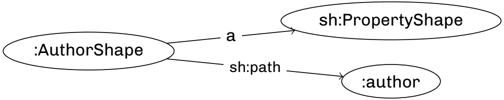
A *node shape* is a shape that is **not** the subject of a triple with *sh:path* as its predicate.



```
:BookShape
  a sh:NodeShape ;
  sh:targetClass :Book .
```

sh:PropertyShape

A *property shape* is a shape that is the subject of a triple that has *sh:path* as its predicate.



```
:AuthorShape
  a sh:PropertyShape ;
  sh:path :author .
```

BookShape

```
:BookShape
  a sh:NodeShape ;
  sh:targetClass :Book ;
  sh:property :AuthorShape .

:AuthorShape
  a sh:PropertyShape ;
  sh:path :author .
```

SHACL Core Constraint Components

SHACL Core Constraint Components

Value type

sh:class	Each value node is an instance of a given type.
sh:datatype	Datatype of each value node.
sh:nodeKind	Node kind (IRI, blank node etc.) of each value node.

```
:BookShape
  a sh:NodeShape ;
  sh:targetClass :Book ;
  sh:property [
    sh:path :author ;
    sh:class :Person ;
  ] .
```

SHACL Core Constraint Components

Cardinality

sh:minCount	Minimum cardinality as xsd:integer.
sh:maxCount	Maximum cardinality as xsd:integer.

Value range

sh:minExclusive	$x <$ \$value
sh:minInclusive	$x <=$ \$value
sh:maxExclusive	$x >$ \$value
sh:maxInclusive	$x >=$ \$value

```
:BookShape
  a sh:NodeShape ;
  sh:targetClass :Book ;
  sh:property [
    sh:path :pages ;
    sh:minInclusive 10 ;
  ] .
```

SHACL Core Constraint Components

String-based	
sh:minLength	Minimum length as xsd:integer.
sh:maxLength	Maximum length as xsd:integer.
sh:pattern	Regular expression.
sh:languageIn	A list of languages as per RFC5646.
sh:uniqueLang	One unique tag per language.

```
:BookShape
  a sh:NodeShape ;
  sh:targetClass :Book ;
  sh:property [
    sh:path :ISBN ;
    sh:pattern "^(?=(?:\D*\d){10}(?:(?:\D*\d){3})?$)[\d-]+$" ;
  ] .
```

SHACL Core Constraint Components

Property pair	Compare two IRIs where,
sh:equals	$x \equiv y$
sh:disjoint	$x \cap y = \emptyset$
sh:lessThan	$x < y$
sh:lessThanOrEquals	$x <= y$

```
:PersonShape
  a sh:NodeShape ;
  sh:targetClass :Person ;
  sh:property [
    sh:path :birth ;
    sh:lessThanOrEquals :death ;
  ] .
```

SHACL Core Constraint Components

Logical	List of value nodes that,
sh:not	Cannot conform to given shape.
sh:and	Conforms to all provided shapes.
sh:or	Conforms to at least one of the provided shapes.
sh:xone	Conforms to exactly one of the provided shapes.

```
:PersonShape
  a sh:NodeShape ;
  sh:targetClass :Person ;
  sh:or (
    [ sh:path :firstName ; sh:minCount 1 ; ]
    [ sh:path :lastName ; sh:minCount 1 ; ]
  ) .
```

SHACL & OWL

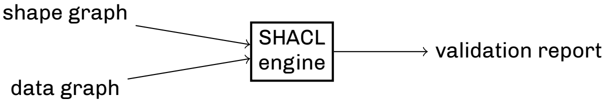
```
:BookShape
  a sh:NodeShape, owl:Class ;
  sh:property [
    sh:path :author ;
    sh:or (
      [ sh:class :Author ]
      [ sh:datatype xsd:string ]
    )
  ] .
```

SHACL Core Constraint Components

Shape-based	Each value node,
sh:node	Conforms to the given node shape.
sh:property	Has a given property shape.
Other	
sh:closed	Boolean signalising a complete shape.
sh:ignoredProperties	List of properties to ignore.
sh:hasValue	At least one value node is equal to the given term.
sh:in	Value node is member of given list.

```
:BookShape
  a sh:NodeShape ;
  sh:targetClass :Book ;
  sh:closed true ;
  sh:ignoredProperties (rdf:type) .
```

SHACL engine



Validation report

Each instance of *sh:ValidationReport* has exactly one value of *sh:conforms*.

sh:conforms is true iff the validation did not produce any **validation results**, and false otherwise.

Iff validation conforms false, the report will contain an instance of **sh:ValidationResult**.

```
[
  a sh:ValidationReport ;
  sh:conforms true ;
] .
```

Validation result

All properties described can be specified in a validation result.

sh:focusNode	Node that caused the result.
sh:resultPath	Pointing to value of **sh:path**
sh:value	Value node that violated constraint.
sh:sourceShape	Shape that given focus node validated against.
sh:sourceConstraintComponent	Constraint component that caused the result.
sh:detail	Parent result containing more details about the violation.
sh:message	Annotation property with textual details.
sh:severity	Default **sh:Violation**.

Other nice to knows about SHACL

- Deactivating shapes
- Non-validating property shape characteristics
  - sh:name & sh:description
  - sh:order & sh:group
  - sh:defaultValue
- Syntax checking of shapes graph

SHACL Implementations

Framework	
ruby-rdf/shacl	https://github.com/ruby-rdf/shacl
dotNetRDF	https://dotnetrdf.org/docs/stable/api/VDS.RDF.Shacl.html
pySHACL	https://github.com/RDFLib/pySHACL
RDF4J	https://rdf4j.org/
Jena	https://jena.apache.org/
Vendors	
TopQuadrant	https://www.topquadrant.com/
Stardog	https://www.stardog.com/
Cambridge Semantics	https://cambridgesemantics.com/anzograph/
Franz	https://allegrograph.com
Web playground	
SHACL Playground	https://shacl.org/playground/

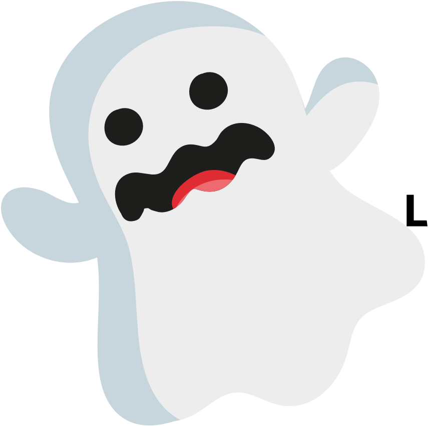
Not covered in this masterclass

Variants of property paths.

- sh:inversePath
- sequence path
- sh:alternativePath
- sh:zeroOrMorePath

Beyond SHACL Core

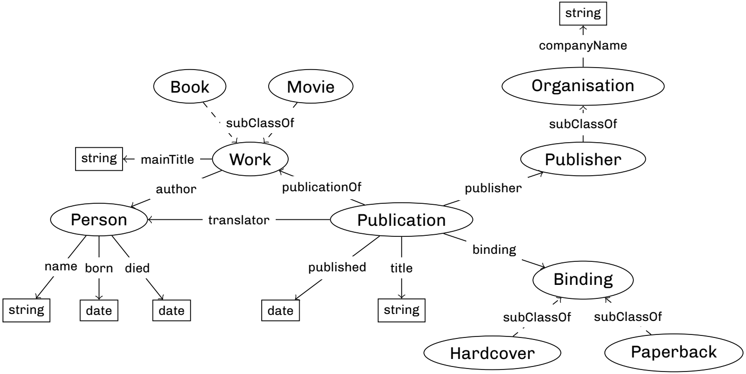
- SHACL-SPARQL
- DASH Data Shapes Vocabulary
- SHACL Advanced Features (SHACL-AF)



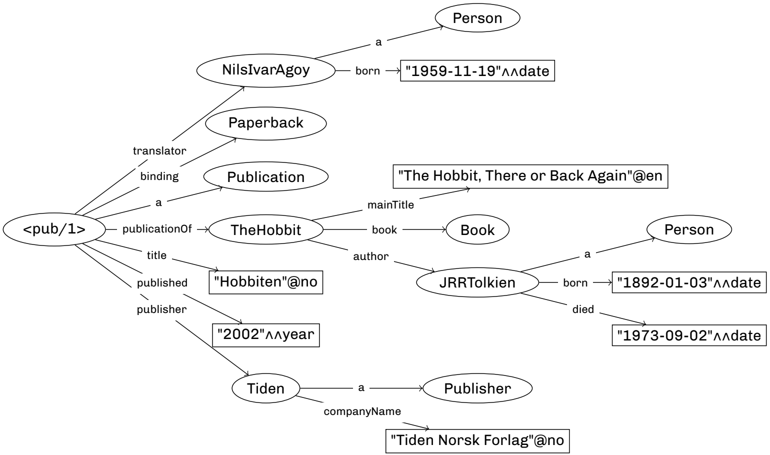
Live coding!

- Publications of books
- Persons and affiliations

Example Publication Ontology



Example data graph



References & resources

Images

My toy box & bookshelf	freepik.com

Around the web

W3C Recommendation	*Shape Constraint Language*	`https://www.w3.org/TR/shacl/`
Holger Knublauch	*SHACL and OWL Compared*	`https://spinrdf.org/shacl-and-owl.html`
W3C Working Group Note	*SHACL Advanced Features*	`https://w3c.github.io/shacl/shacl-af/`
TopQuadrant	*DASH Data Shapes*	`http://datashapes.org/`

Book

Jose Emilio labra Gayo, Eric Prud'hommeaux, Iovka Boneva, Dimitris Kontokostas, *Validating RDF Data*, 2018.