Europeana and RDF data validation

Antoine Isaac

RDF Validation Workshop
10-11 September 2013
Data validation on the Europeana Data Model

EDM is RDF, but Europeana needs to enforce constraints on the datasets sent by its providers

Matching basic Europeana functional requirements, e.g.:
- at most one `edm:isShownBy`
- at most one `edm:isShownAt`
- either `edm:isShownBy` or `edm:isShownAt` is mandatory

General data quality, e.g.:
- at least a `dc:title` or a `dc:description`

http://pro.europeana.eu/edm-documentation
**EDM “Mapping Guidelines”**

Template-based instructions for Europeana providers

<table>
<thead>
<tr>
<th>property</th>
<th>value type</th>
<th>cardinality</th>
</tr>
</thead>
<tbody>
<tr>
<td>edm:aggregatedCHO</td>
<td>reference (of an item)</td>
<td>min 1, max 1</td>
</tr>
<tr>
<td>edm:dataProvider</td>
<td>literal or reference</td>
<td>min 1, max 1</td>
</tr>
<tr>
<td>edm:isShownAt</td>
<td>reference</td>
<td>min 0, max 1 --- Either isShownBy OR isShownAt is Mandatory</td>
</tr>
<tr>
<td>edm:isShownBy</td>
<td>reference</td>
<td>min 0, max 1 --- Either isShownBy OR isShownAt is Mandatory</td>
</tr>
<tr>
<td>edm:object</td>
<td>reference</td>
<td>min 0, max 1</td>
</tr>
<tr>
<td>edm:provider</td>
<td>literal or reference</td>
<td>min 1, max 1</td>
</tr>
<tr>
<td>dc:rights</td>
<td>reference or literal</td>
<td>min 0, max unbounded</td>
</tr>
<tr>
<td>edm:rights</td>
<td>reference</td>
<td>min 1, max 1</td>
</tr>
<tr>
<td>edm:ugc</td>
<td>literal (true)</td>
<td>min 0, max 1</td>
</tr>
</tbody>
</table>
Machine-readable specs by OWL ontology?

→ We have an OWL version of EDM


→ But as we know: OWL is good for writing down constraints, not for validation

→ And in OWL some EDM constraints amount to adding semantics to classes and properties that already exist

an ore:Aggregation should have at least 1 edm:isShownAt or 1 edm:isShownBy

(let’s be honest: we were not ready for full RDF/OWL compatibility anyway…)
Falling back to XML Schema

EDM is implemented as XML Schema (for RDF data!)

```
<sequence>
    [...]
    <element ref="edm:dataProvider" maxOccurs="1" minOccurs="1"/>
    <element ref="edm:isShownAt" maxOccurs="1" minOccurs="0"/>
    <element ref="edm:isShownBy" maxOccurs="1" minOccurs="0"/>
    [...]
</sequence>
```

With Schematron rules:

```
<sch:pattern>
    <sch:rule context="ore:Aggregation">
        <sch:assert test="edm:isShownAt or edm:isShownBy">
            [Error message]
        </sch:assert>
    </sch:rule>
</sch:pattern>
```

Not ideal of course

- Document-centric approach to validation
- Extra constraints, especially order of elements
- 2 constraint systems co-existing
EDM as a Dublin Core application profile?

[Cf. Karen and Tom tomorrow]

An example in the “Description Set Profiles” constraint language:

```
DescriptionSet [EDM-Providers]
  Description [Aggregation-Providers]
    Resource Class
      ore:Aggregation
    Statement
      Property
        edm:isShownBy
        edm:isShownAt
    Min Occurs
      1
```

http://dublincore.org/documents/dc-dsp/
Could be converted to other formalisms

SPIN:

```xml
could be converted to other formalisms

ore:Aggregation
  spin:constraint
    [ a sp:Ask ;
      sp:text """"
      # either isShownBy or isShownAt must be present
      ASK WHERE {
        {?this isShownBy ?image } UNION {?this isShownBy ?page }
      }"""
    ] .
```

Stardog ICV:

```
Class: ore:Aggregation
  SubClassOf: min 1 edm:isShownBy or min 1 edm:isShownAt
```

**Issue:** *still looks like adding general semantics to ore:Aggregation*…
Making our requirements clearer

Level 1: Enabling basic validation

→ Expressivity for individual constraints
  
  Needs further testing, but DC AP, “OWL-inspired” and SPARQL seem good
  
  OWL would probably force us to introduce many “technical” classes & properties

→ Scalability
  
  ?

Level 2: “Packaging data” expressing scope of constraints – datasets!

→ Side requirement: constraints should read less like messing up with the original semantics of classes and properties

  DC AP approach provides better hooks for tying constraints to groups of datasets
Making our requirements clearer

Level 3: sharing and re-use of constraints

→ For humans: relative ease of understanding. Europeana has a wide network of partners, not always tech-savvy.

*OWL terms are hard, SPARQL seems low-level (even though it’s not)*

→ For machines: higher-level expressions of all constraint will help implementation in different frameworks

*XML/Schematron bad at making different levels of expression/implementation clear*

Level 2: “Packaging data” expressing scope of constraints – datasets!

→ Other organizations (esp. cultural aggregators) could make their own profiles of EDM, with some constraints in common but not all

*Importance of “packaging” data*
Thank you!

aisaac@few.vu.nl