Towards a Uniform User Interface for Editing Data Shapes

Ben De Meester, ben.demeester@ugent.be, @Ben__DM
Pieter Heyvaert, Anastasia Dimou, and Ruben Verborgh

IDLab, Department of Electronics and Information Systems, Ghent University – imec, Ghent, Belgium
Garbage In, Garbage Out?

Data shapes
Easy editing is important

“Fitness for use”

Constraint languages: **declaration and implementation are decoupled**
SHACL (W3C Recommendation)
ShEx
...

**Machine-processability in mind**
What are the necessary features for visually editing data shapes?
Outline

1. SOTA
2. Features
3. PoC: unSHACLed
   - UI
   - Features
4. Conclusions
Outline

1 SOTA
2 Features
3 PoC: unSHACLed
   UI
   Features
4 Conclusions
State of the Art

Data shapes
Validation based on: OWL | SPARQL | SHACL, ShEx

Data shape editors
Depend on the (constraint) language or enforce a linear workflow

Editors
Data editors: text-based, form-based, use-case specific
Ontology editors: graph-based, indented-tree-based, UML-based
SPARQL editors: text-based
Linked Data generation rule editors: form-based, graph-based
Outline

1 SOTA

2 Features

3 PoC: unSHACLed
   UI
   Features

4 Conclusions
Desired Features for Data Shape Editing

1. Independence of constraint language
2. Support multiple data sources
3. Support different serializations
4. Support multiple ontologies
5. Multiple alternative editing approaches
6. Non-linear workflows
7. Independence of execution
Outline

1 SOTA
2 Features
3 **PoC: unSHACLed**
   - UI
   - Features
4 Conclusions
unSHACLed

Visual Data Shapes editor as Web application
Drag-and-drop loaded data graphs and data shapes
Add data shapes using templates
Get visual feedback on conformance
Export shape

https://w3id.org/imec/unshacled/app
The unSHACLED UI, consisting of an **Overview Sidebar** (left), an **Action Toolbar** (top), and an **Editing Area** (middle-right).
## Different elements enable the different features

<table>
<thead>
<tr>
<th>Features</th>
<th>UI Element</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Overview</td>
</tr>
<tr>
<td></td>
<td>Sidebar</td>
</tr>
<tr>
<td>F1. Independence of constraint language</td>
<td>✓</td>
</tr>
<tr>
<td>F2. Support multiple data sources</td>
<td></td>
</tr>
<tr>
<td>F3. Support different serializations</td>
<td></td>
</tr>
<tr>
<td>F4. Support multiple ontologies</td>
<td></td>
</tr>
<tr>
<td>F5. Multiple alternative modeling approaches</td>
<td></td>
</tr>
<tr>
<td>F6. Non-linear workflows</td>
<td>✓</td>
</tr>
<tr>
<td>F7. Independence of execution</td>
<td></td>
</tr>
</tbody>
</table>
Outline

1 SOTA

2 Features

3 PoC: unSHACLed
   UI
   Features

4 Conclusions
Conclusions

No need to write RDF / SHACL / ShEx

Use-case independent

Open Issues
  (Map)VOWL or UML or ... ?
  User evaluation graphical representation
  Representation large data shapes
    Workspaces
    Detail levels

Features as starting point for visual data shape editors
Towards a Uniform User Interface for Editing Data Shapes

Ben De Meester, ben.demeester@ugent.be, @Ben__DM
Pieter Heyvaert, Anastasia Dimou, and Ruben Verborgh

IDLab, Department of Electronics and Information Systems, Ghent University – imec, Ghent, Belgium