xmlprague

Schematron QuickFix

Nico Kutscherauer

contact@schematron-quickfix.com
@nkutsche

Octavian Nadolu

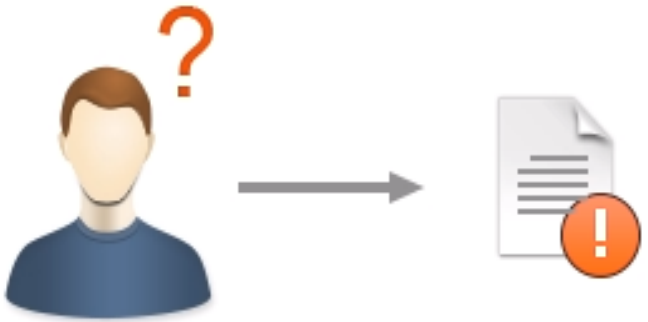
octavian_nadolu@oxygenxml.com
@OctavianNadolu

SQF

<oXygen/> XML Editor

Error Fixes

- Fixing errors has always been a challenge
- Solutions offered by IDEs

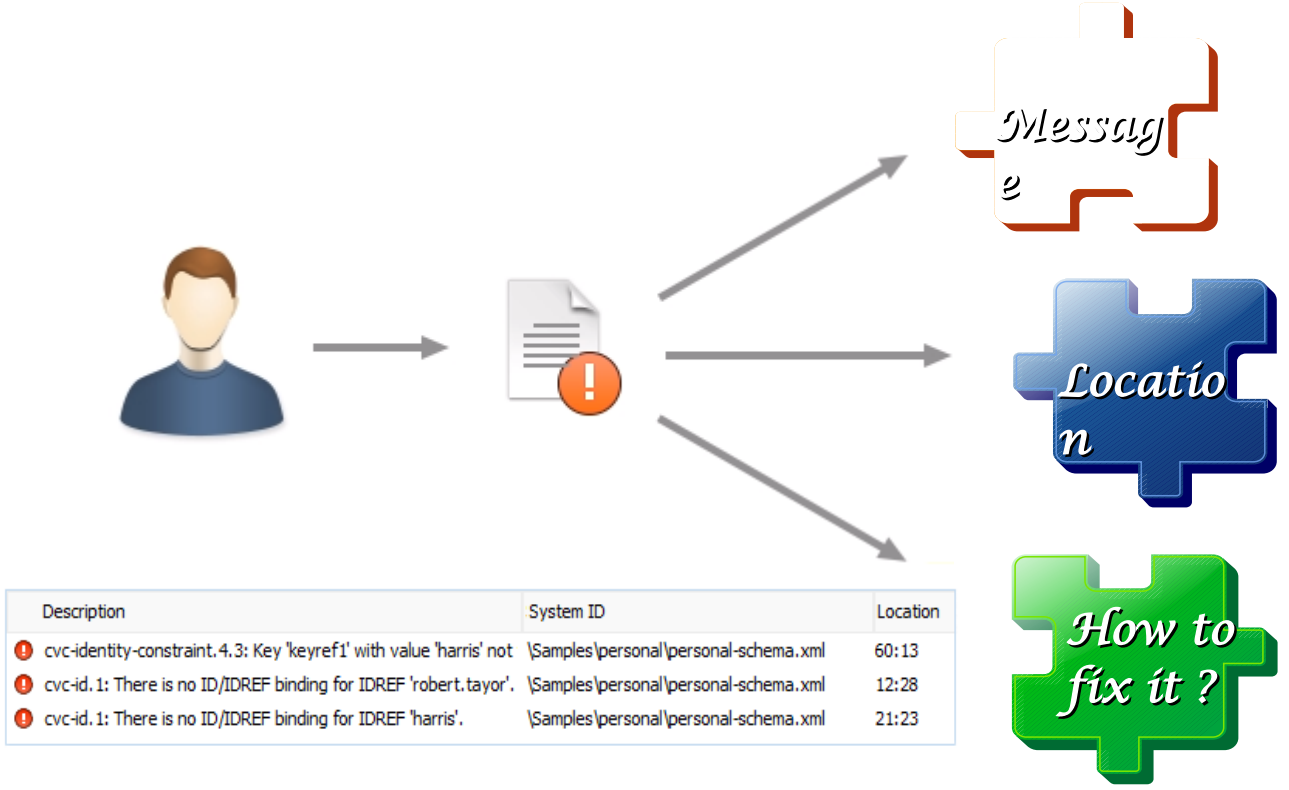


Fix Proposals

- Using fix proposals to solve errors:
  - Better understanding of the problem
  - Fewer (no) mistakes
  - Saves time (money)

XML Validation Errors

- From the user's perspective the validation error is composed by three things



Fixing XML Validation Errors

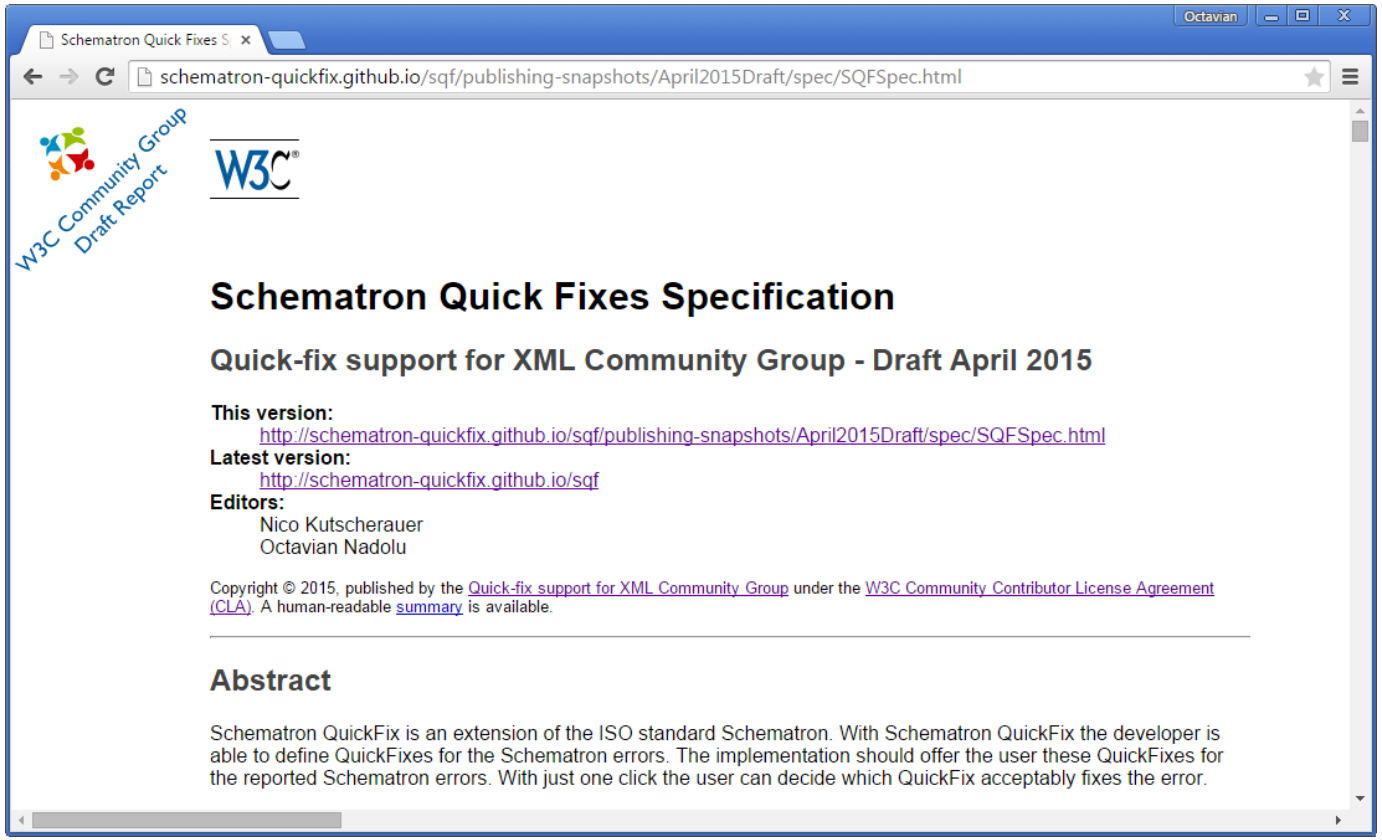
- Predefined Errors - defined in the validation engine
  - Fixes generated from the validation engine
  - Fixes based on the message (error code) and location
- Custom Errors - defined by the user
  - Difficult to generate fixes based on message and location
  - A language to create fixes is more appropriate

Schematron Fix Proposals

- User-defined fixes for Schematron errors
- Schematron QuickFix (SQF) language
  - Extends the Schematron language
  - SQF initiated by Nico Kutscherauer

www.schematron-quickfix.com
github.com/schematron-quickfix/sqf

Schematron Quick Fixes Spec



www.w3.org/community/quickfix

schematron-quickfix.github.io/sqf

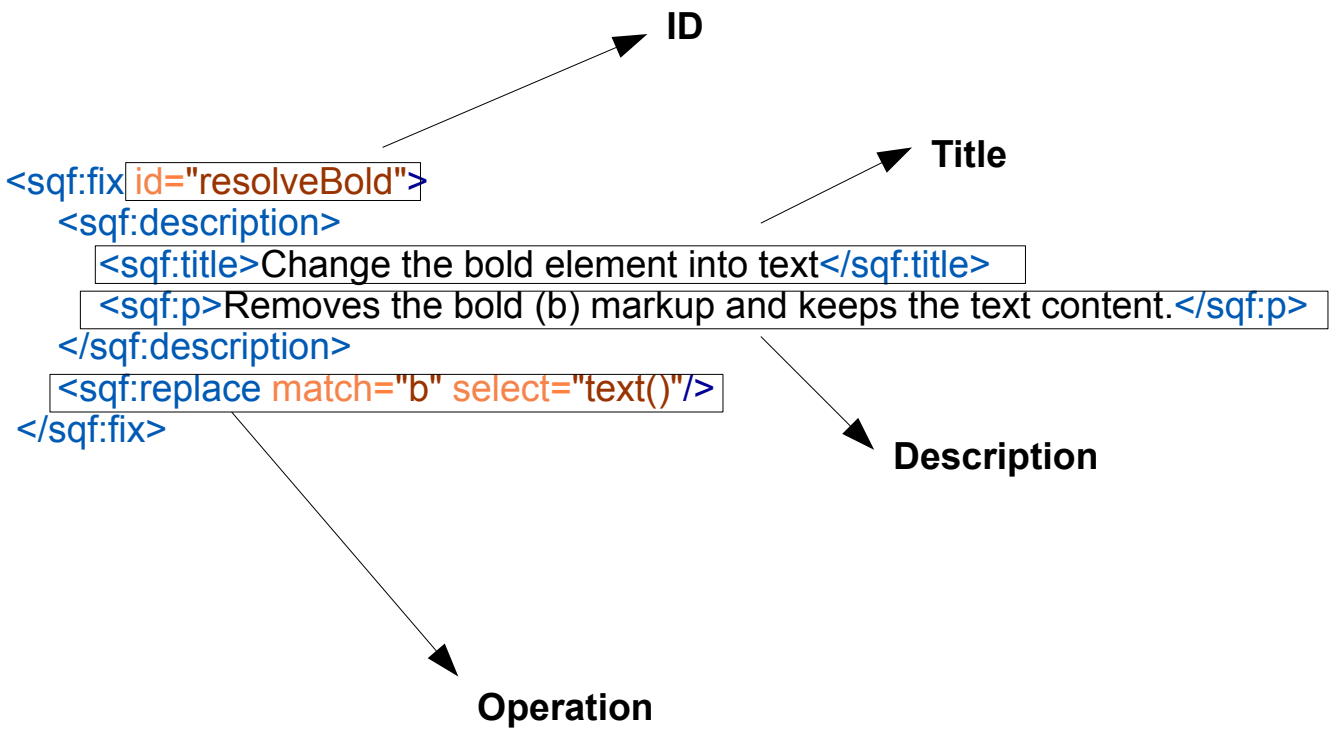
SQF Extension of the Schematron

- Associated with assert and report elements
- Added as Schematron annotations

```
<rule context="html">
    <report test="//comment()" sqf:fix="removeComments">
       Comments are not allowed in document.</report>

    <sqf:fix id="removeComments" role="delete">
       <sqf:description>
          <sqf:title>Remove all comments</sqf:title>
          <sqf:p>Remove all comment nodes from the current document</sqf:p>
       </sqf:description>
       <sqf:delete match="//comment()"/>
    </sqf:fix>
 </rule>
```

Schematron QuickFix (SQF)



SQF Benefits

- ✓ Create custom quick fixes for errors

SQF Benefits

- Create custom quick fixes for errors
- ✓ Use the power of Schematron and XSLT

SQF Benefits

- Create custom quick fixes for errors
- Use the power of Schematron and XSLT
- ✓ Create refactoring actions using SQF

SQF Benefits

- Create custom quick fixes for errors
- Use the power of Schematron and XSLT
- Create refactoring actions using SQF
- ✓ Fix problems in external documents

SQF Benefits

- Create custom quick fixes for errors
- Use the power of Schematron and XSLT
- Create refactoring actions using SQF
- Fix problems in external documents

✓ Fixes for any XML documents

Language Overview

- Small language
  - Keep it easy, but extensible
  - No knowledge of XSLT is needed for simple cases
  - Integrate well-known standards for more complex things

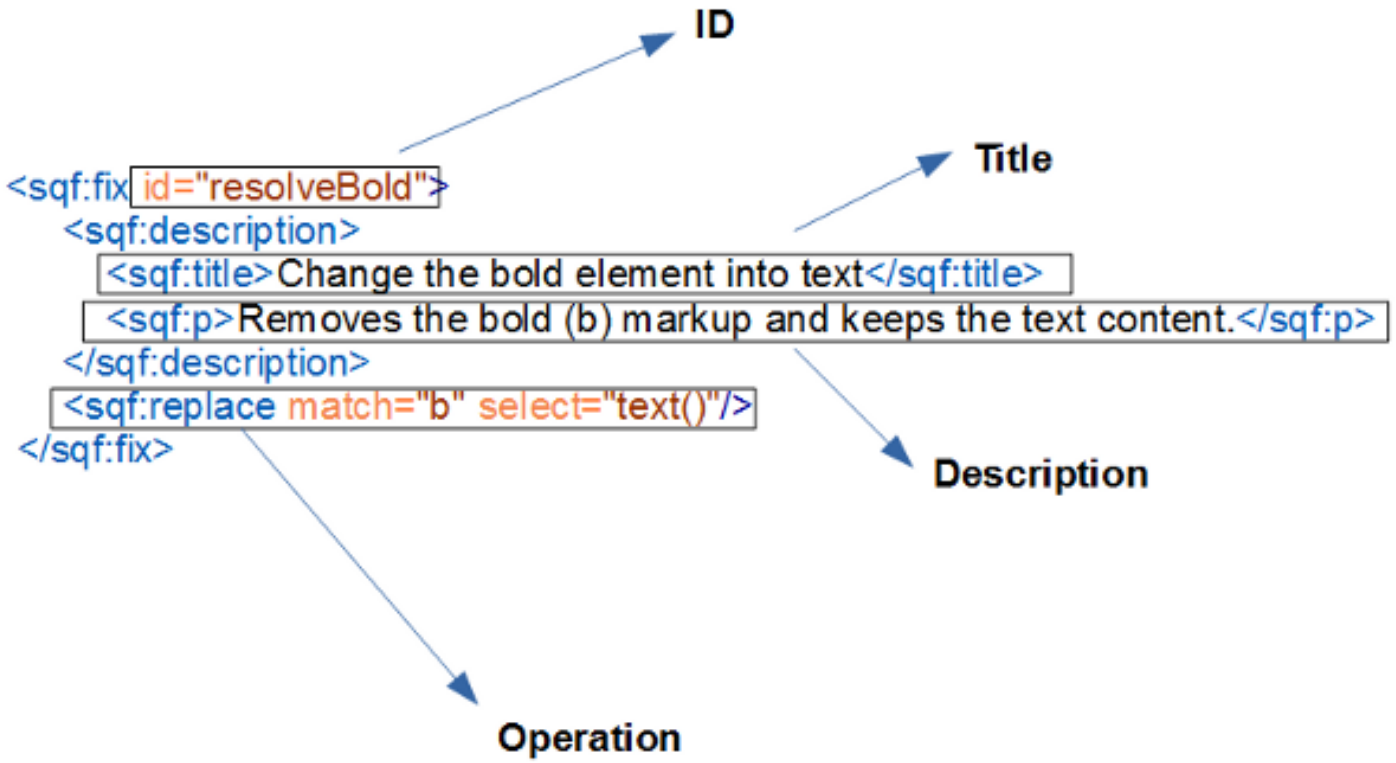
Language Overview

- Small language
  - Keep it easy, but extensible
  - No knowledge of XSLT is needed for simple cases
  - Integrate well-known standards for more complex things
- Own Namespace with prefix sqf:

http://www.schematron-quickfix.com/validator/process

Language Structure

- Reference / structure
- User interface
- Activity Elements (operations)



Language Structure

- Reference / structure
- User interface
- Activity Elements (operations)
- Generic features

Learning By Examples

- Prepared 6 examples
- All to find on

  https://github.com/octavianN/SQFPresentation/tree/master/Samples
- Additional Examples
  http://www.schematron-quickfix.com/examples.html
- Tutorial
  http://www.schematron-quickfix.com/quickFix/guide.html

1. Simple Replace

- Replace or unwrap a node

```
<sch:rule context="b">
   <sch:report test="ancestor::b"
      sqf:fix="italic unwrap">
      Bold in bold is not allowed.</sch:report>
   <sqf:fix id="italic">
      <sqf:description>
         <sqf:title>Change it to italic.</sqf:title>
      </sqf:description>
      <sqf:replace match="." node-type="element" target="i" select="node()"/>
   </sqf:fix>
   <sqf:fix id="unwrap">
      <sqf:description>
         <sqf:title>Unwrap <sch:name/> element</sqf:title>
      </sqf:description>
      <sqf:replace select="node()"/>
   </sqf:fix>
</sch:rule>
```

1. Simple Replace

- Replace or unwrap a node

```
<sch:rule context="b">
   <sch:report test="ancestor::b"
      sqf:fix="italic unwrap">
      Bold in bold is not allowed.</sch:report>
   <sqf:fix id="italic">
      <sqf:description>
         <sqf:title>Change it to italic.</sqf:title>
      </sqf:description>
      <sqf:replace match="." node-type="element" target="i" select="node()"/>
   </sqf:fix>
   <sqf:fix id="unwrap">
      <sqf:description>
         <sqf:title>Unwrap <sch:name/> element</sqf:title>
      </sqf:description>
      <sqf:replace select="node()"/>
   </sqf:fix>
</sch:rule>
```

1. Simple Replace

- Replace or unwrap a node

```
<sch:rule context="b">
   <sch:report test="ancestor::b"
      sqf:fix="italic unwrap">
      Bold in bold is not allowed.</sch:report>
   <sqf:fix id="italic">
      <sqf:description>
         <sqf:title>Change it to italic.</sqf:title>
      </sqf:description>
      <sqf:replace match="." node-type="element" target="i" select="node()"/>
   </sqf:fix>
   <sqf:fix id="unwrap">
      <sqf:description>
         <sqf:title>Unwrap <sch:name/> element</sqf:title>
      </sqf:description>
      <sqf:replace select="node()"/>
   </sqf:fix>
</sch:rule>
```

1. Simple Replace

- Replace or unwrap a node

```
<sch:rule context="b">
   <sch:report test="ancestor::b"
      sqf:fix="italic unwrap">
      Bold in bold is not allowed.</sch:report>
   <sqf:fix id="italic">
      <sqf:description>
         <sqf:title>Change it to italic.</sqf:title>
      </sqf:description>
      <sqf:replace match="." node-type="element" target="i" select="node()"/>
   </sqf:fix>
   <sqf:fix id="unwrap">
      <sqf:description>
         <sqf:title>Unwrap <sch:name/> element</sqf:title>
      </sqf:description>
      <sqf:replace select="node()"/>
   </sqf:fix>
</sch:rule>
```

1. Simple Replace

- Replace or unwrap a node

```
<sch:rule context="b">
   <sch:report test="ancestor::b"
      sqf:fix="italic unwrap">
      Bold in bold is not allowed.</sch:report>
   <sqf:fix id="italic">
      <sqf:description>
         <sqf:title>Change it to italic.</sqf:title>
      </sqf:description>
      <sqf:replace match="." node-type="element" target="i" select="node()"/>
   </sqf:fix>
   <sqf:fix id="unw
      <sqf:descript
         <sqf:title>U
      </sqf:descrip
      <sqf:replace
   </sqf:fix>
</sch:rule>
```

Defines the Anchor nodes.

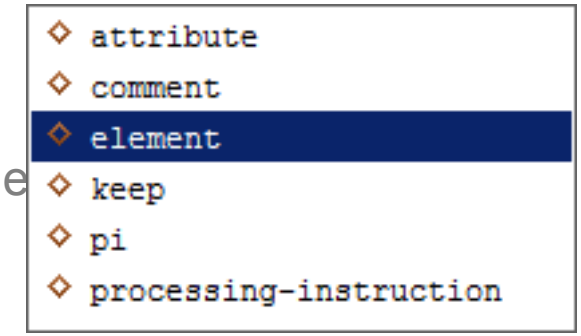
The operation will be executed relative to the Anchor node

Value: XPath expression relative to the context of the error.

1. Simple Replace

- Replace or unwrap a node

```
<sch:rule context="b">
   <sch:report test="ancestor::b"
      sqf:fix="italic unwrap">
      Bold in bold is not allowed.</sch:report>
   <sqf:fix id="italic">
      <sqf:description>
         <sqf:title>Change it to italic.</sqf:title>
      </sqf:description>
      <sqf:replace match="." node-type="element" target="i" select="node()"/>
   </sqf:fix>
   <sqf:fix id="unwrap">
      <sqf:description>
         <sqf:title>Unwrap <sch:name/> e
      </sqf:description>
      <sqf:replace select="node()"/>
   </sqf:fix>
</sch:rule>
```



1. Simple Replace

- Replace or unwrap a node

```
<sch:rule context="b">
   <sch:report test="ancestor::b"
      sqf:fix="italic unwrap">
      Bold in bold is not allowed.</sch:report>
   <sqf:fix id="italic">
      <sqf:description>
         <sqf:title>Change it to italic.</sqf:title>
      </sqf:description>
      <sqf:replace match="." node-type="element" target="i" select="node()"/>
   </sqf:fix>
   <sqf:fix id="unwrap">
      <sqf:description>
         <sqf:title>Unwrap <sch:name/> element</sqf:title>
      </sqf:description>
      <sqf:replace select="node()"/>
   </sqf:fix>
</sch:rule>
```

2. Move Nodes

- Move a node – delete and add

```
<sch:rule context="title">
   <sch:report test="preceding-sibling::abstract" sqf:fix="move">
      The abstract element shouldn't be placed before the <sch:name/> element.
   </sch:report>
      <sqf:fix id="move">
         <sqf:description>
            <sqf:title>Move the abstract element after the title</sqf:title>
            <sqf:p>The missplaced abstract element will be deleted and
                     inserted direct after the title element</sqf:p>
         </sqf:description>
         <sqf:delete match="preceding-sibling::abstract"/>
         <sqf:add match="." select="preceding-sibling::abstract" position="after"/>
      </sqf:fix>
</sch:rule>
```

2. Move Nodes

- Move a node – delete and add

```
<sch:rule context="title">
   <sch:report test="preceding-sibling::abstract" sqf:fix="move">
      The abstract element shouldn't be placed before the <sch:name/> element.
   </sch:report>
      <sqf:fix id="move">
         <sqf:description>
            <sqf:title>Move the abstract element after the title</sqf:title>
            <sqf:p>The missplaced abstract element will be deleted and
                     inserted direct after the title element</sqf:p>
         </sqf:description>
         <sqf:delete match="preceding-sibling::abstract"/>
         <sqf:add match="." select="preceding-sibling::abstract" position="after"/>
      </sqf:fix>
</sch:rule>
```

2. Move Nodes

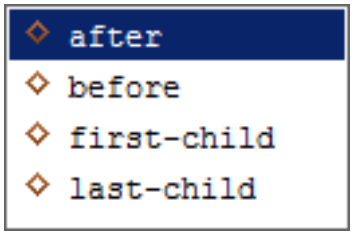
- Move a node – delete and add

```
<sch:rule context="title">
   <sch:report test="preceding-sibling::abstract" sqf:fix="move">
      The abstract element shouldn't be placed before the <sch:name/> element.
   </sch:report>
      <sqf:fix id="move">
         <sqf:description>
            <sqf:title>Move the abstract element after the title</sqf:title>
            <sqf:p>The missplaced abstract element will be deleted and
                     inserted direct after the title element</sqf:p>
         </sqf:description>
         <sqf:delete match="preceding-sibling::abstract"/>
         <sqf:add match="." select="preceding-sibling::abstract" position="after"/>
      </sqf:fix>
</sch:rule>
```

2. Move Nodes

- Move a node – delete and add

```
<sch:rule context="title">
   <sch:report test="preceding-sibling::abstract" sqf:fix="move">
      The abstract element shouldn't be placed before the <sch:name/> element.
   </sch:report>
      <sqf:fix id="move">
         <sqf:description>
            <sqf:title>Move the abstract element after the title</sqf:title>
            <sqf:p>The missplaced abstract element will be deleted and
                     inserted direct after the title element</sqf:p>
         </sqf:description>
         <sqf:delete match="preceding-sibling::abstract"/>
         <sqf:add match="." select="preceding-sibling::abstract" position="after"/>
      </sqf:fix>
</sch:rule>
```



3. User Entries

- User Entry – parameter of the QuickFix

```
<sch:rule context="title">
   <sch:assert test="normalize-space(.) != '' " sqf:fix="title"
      >A title shouldn't be empty.</sch:assert>
   <sqf:fix id="title">
      <sqf:description>
         <sqf:title>Set a title</sqf:title>
         <sqf:p>This QuickFix will set a title by using a User Entry.</sqf:p>
      </sqf:description>
      <sqf:user-entry name="title" type="xs:string">
         <sqf:description>
            <sqf:title>Please enter the new title.</sqf:title>
         </sqf:description>
      </sqf:user-entry>
      <sqf:replace target="{name()}" node-type="element"
         select="$title" />
   </sqf:fix>
</sch:rule>
```

3. User Entries

- User Entry – parameter of the QuickFix

```
<sch:rule context="title">
   <sch:assert test="normalize-space(.) != '' " sqf:fix="title"
      >A title shouldn't be empty.</sch:assert>
   <sqf:fix id="title">
      <sqf:description>
         <sqf:title>Set a title</sqf:title>
         <sqf:p>This QuickFix will set a title by using a User Entry.</sqf:p>
      </sqf:description>
      <sqf:user-entry name="title" type="xs:string">
         <sqf:description>
            <sqf:title>Please enter the new title.</sqf:title>
         </sqf:description>
      </sqf:user-entry>
      <sqf:replace target="{name()}" node-type="element"
         select="$title" />
   </sqf:fix>
</sch:rule>
```

4. QuickFix Conditions

- Conditions – provide a QuickFix only if it makes sense

```
<sch:rule context="head/title">
   <sch:assert test="string-length(normalize-space(.)) le 20 "
      sqf:fix="title">
      A title shouldn't have more than 20 characters.</sch:assert>
   <sqf:fix id="title" use-when="//h1[1][string-length(.) le 20]">
      <sqf:description>
         <sqf:title>
            Set the title to "<sch:value-of select="//h1[1]"/>".
         </sqf:title>
      </sqf:description>
      <sqf:replace target="title" node-type="element">
         <sch:value-of select="//h1[1]"/>
      </sqf:replace>
   </sqf:fix>
</sch:rule>
```

4. QuickFix Conditions

- Conditions – provide a QuickFix only if it makes sense

```
<sch:rule context="head/title">
   <sch:assert test="string-length(normalize-space(.)) le 20 "
      sqf:fix="title">
      A title shouldn't have more than 20 characters.</sch:assert>
   <sqf:fix id="title" use-when="//h1[1][string-length(.) le 20]">
      <sqf:description>
         <sqf:title>
            Set the title to "<sch:value-of select="//h1[1]"/>".
         </sqf:title>
      </sqf:description>
      <sqf:replace target="title" node-type="element">
         <sch:value-of select="//h1[1]"/>
      </sqf:replace>
   </sqf:fix>
</sch:rule>
```

4. QuickFix Conditions

- Conditions – provide a QuickFix only if it makes sense

```
<sch:rule context="head/title">
   <sch:assert test="string-length(normalize-space(.)) le 20 "
      sqf:fix="title">
      A title shouldn't have more than 20 characters.</sch:assert>
   <sqf:fix id="title" use-when="//h1[1][string-length(.) le 20]">
      <sqf:description>
         <sqf:title>
            Set the title to "<sch:value-of select="//h1[1]"/>".
         </sqf:title>
      </sqf:description>
      <sqf:replace target="title" node-type="element">
         <sch:value-of select="//h1[1]"/>
      </sqf:replace>
   </sqf:fix>
</sch:rule>
```

5. Dynamic QuickFixes

- Dynamic QuickFixes – in the future

```
<sch:rule context="head/title">
   <sch:assert test="string-length(normalize-space(.)) le 20 "
      sqf:fix="title">
      A title shouldn't have more than 20 characters.</sch:assert>
   <sqf:fix id="title" use-for-each="//h1[string-length(.) le 20]">
      <sqf:description>
         <sqf:title>
            Set the title to "<sch:value-of select="$sqf:current"/>".
         </sqf:title>
      </sqf:description>
      <sqf:replace target="title" node-type="element">
         <sch:value-of select="$sqf:current"/>
      </sqf:replace>
   </sqf:fix>
</sch:rule>
```

6. Call QuickFixes

```
<sqf:fix id="title">
   <sqf:call-fix ref="createElementRowAsFirstChild">
      <sqf:with-param name="match" select="."/>
      <sqf:with-param name="el" select=" 'col' "/>
      <sqf:with-param name="count" select="count(max(.//tr/count(td|th)))"/>
   </sqf:call-fix>
</sqf:fix>
<sqf:fix id="createElementRowAsFirstChild">
   <sqf:param name="match" type="node()*"/>
   <sqf:param name="el" type="xs:QName"/>
   <sqf:param name="count" type="xs:integer"/>
   <sqf:description>
      <sqf:title>Create a row of <sch:value-of select="$count"/>
         <sch:name path="$el"/> elements as a first child of the
         <sch:name path="$match"/> element(s).</sqf:title>
   </sqf:description>
   <sqf:add match="$match" position="first-child">
      <xsl:for-each select="1 to $count">
         <xsl:element name="{$el}"/>
      </xsl:for-each>
   </sqf:add>
</sqf:fix>
```

Call QuickFixes

- In the first draft:
  - Just a prototype
  - Not really functional
- Improvements of the second draft:
  - Multiple sqf:call-fix in one sqf:fix
  - Use the description of the called fix

  → Open discussion

Review

- Structure / Reference
  - sqf:fix, @id, @use-when, @sqf:fix, sqf:group, sqf:fixes
- Description
  - sqf:description, sqf:title, sqf:p
- Activity
  - sqf:delete, sqf:replace, sqf:add, sqf:keep, sqf:stringReplace
- Generic
  - sqf:call-fix, sqf:with-param, sqf:param, sqf:user-entry, @use-for-each

  → Only 15 Elements (XSLT 2: 49, Schematron: 21)

SQF Implementations

- <oXygen/> XML Editor validation engine

  http://www.oxygenxml.com
- Escali Schematron engine

  http://schematron-quickfix.com/escali_xsm.html
  - Escali Schematron command line tool
  - Oxygen plugin for invoking Escali Schematron

Projects using SQF

- Thieme - publishing company uses a custom framework to create and edit XML documents
- parsX - a product developed by pagina GmbH used to facilitate EPUB production
- ART-DECOR - an open source tool suite that supports SDOs active in the healthcare industry
  Sample SQF embedded in XSD
- ATX custom framework – used by a major automotive producer

Projects using SQF

- Dynamic Information Model (DIM) - an implementation of an intelligent style guide
- Schematron for TEI - collection of Schematron and SQF resources for TEI
- <oXygen/> DITA framework - built-in framework in <oXygen/> XML Editor for DITA documents
- <oXygen/> XML userguide - the public version of the <oXygen/> User Manual

Conclusions and Future Plans

- SQF is a simple and useful language
- Helps users to solve the problems in less time and with fewer (no) errors

- Update the SQF specification
- Publish the second draft of the Schematron QuickFix specification

Thank you!

Questions?

contact@schematron-quickfix.com
@nkutsche

octavian_nadolu@oxygenxml.com
@OctavianNadolu