Schematron Step-By-Step Guide

Octavian Nadolu, Syncro Soft
octavian_nadolu@oxygenxml.com
@OctavianNadolu

tcworld, 2018

syncro soft

<oXygen/> XML Editor

About the Author

- Software architect
- 15+ years XML technologies experience
- Contributor to a number of XML-related open source projects
- Editor of Schematron QuickFix specification developed by a W3C community group

Overview

- Schematron history
- Schematron step-by-step
- Schematron in the development process
- Schematron for technical and non-technical users
- Schematron Quick Fixes

What is Schematron?

- A natural language for making assertions in documents

Schematron is an Open Standard

- Schematron is an Open Standard adopted by hundreds of major projects in the past decade

Schematron History

- Schematron was invented by Rick Jelliffe
  - Schematron 1.0 (1999), 1.3 (2000), 1.5 (2001)
  - ISO Schematron (2006, 2010, 2016)

Why Schematron?

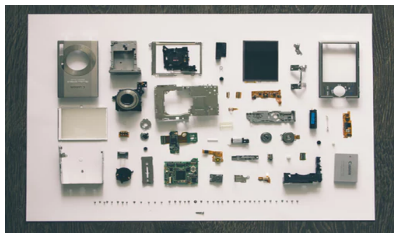
- You can express constraints in a way that you cannot perform with other schemas (like XSD, RNG, or DTD).
  - XSD, RNG, and DTD schemas define structural aspects and data types of the XML documents
  - Schematron allows you to create custom rules specific to your project

Schematron Usage

- Verify data inter-dependencies
- Check data cardinality
- Perform algorithmic checks

Used in Multiple Domains

- Financial
- Insurance
- Government
- Technical publishing



Schematron is Very Simple

- There are only 6 basic elements:

assert

report

rule

pattern

schema

ns

Schematron Step-by-Step

XML

A list of books from a bookstore:

```
...
<book price="75">
    <title>XSLT 2.0 Programmer's Reference</title>
    <author>Michael Kay</author>
    <isbn>0764569090</isbn>
</book>
...
```

<assert>

An **assert** generates a message when a test statement evaluates to **false**

```
<assert test="@price > 10">The book price is too small</assert>
```

<report>

A **report** generates a message when a test statement evaluate to **true**.

```
<report test="@price > 1000">The book price is too big</report>
```

<rule>

A **rule** defines a context on which a list of assertions will be tested, and is comprised of one or more **assert** or **report** elements.

```
<rule context="book">
   <assert test="@price > 10">The book price is too small</assert>
   <report test="@price > 1000">The book price is too big</report>
</rule>
```

<pattern>

A set of rules giving constraints that are in some way related

```
<pattern>
   <rule context="book">
    <assert test="@price > 10">The book price is too small</assert>
     <report test="@price > 1000">The book price is too big</report>
   </rule>
</pattern>
```

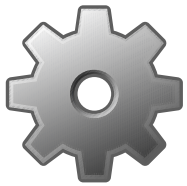
<schema>

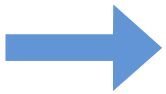
The top-level element of a Schematron schema.

```
<schema xmlns="http://purl.oclc.org/dsdl/schematron">
  <pattern>
    <rule context="book">
      <assert test="@price > 10">The book price is too small</assert>
       <report test="@price > 1000">The book price is too big</report>
    </rule>
  </pattern>
</schema>
```

Apply Schematron

```
<schema xmlns="http://purl.oclc.org/dsdl/schematron">
  <pattern>
    <rule context="book">
      <assert test="@price > 10">The book price is too small</assert>
       <report test="@price > 1000">The book price is too big</report>
    </rule>
  </pattern>
</schema>
```





⚠ The price is too small

```
...
<book price="7">
   <title>XSLT 2.0 Programmer's Reference</title>
   <author>Michael Kay</author>
   <isbn>0764569090</isbn>
</book>
...
```

<ns>

Defines a namespace prefix and URI

```
<schema xmlns="http://purl.oclc.org/dsdl/schematron">
   <ns uri="http://book.example.com" prefix="b"/>
   <pattern>
      <rule context="b:book">
       <assert test="@price > 10">The book price is too small</assert>
        <report test="@price > 1000">The book price is too big</report>
      </rule>
   </pattern>
</schema>
```

Schematron Uses XPath

- XPath it is very important in Schematron:
  - Rule context is expressed using an XPath expression

```
<rule context="book">
```

  - Assertions are expressed using XPath expressions that are evaluated as *true* or *false*

```
<assert test="@price > 10">
```

- *@queryBinding – determines the XPath version*

```
<schema queryBinding="xslt2">
```

  - *Possible values: xslt, xslt2, xslt3*

What is XPath?

- A language for addressing parts of an XML document (XML Path Language)
- A W3C Recommendation in 1999
  www.w3.org/TR/xpath
- XPath versions 1.0, 2.0, 3.0, 3.1 (2017)

- Tutorials:
  - www.data2type.de/en/xml-xslt-xslfo/xpath/
  - www.xfront.com/xpath/
  - www.zvon.org/comp/m/xpath.html

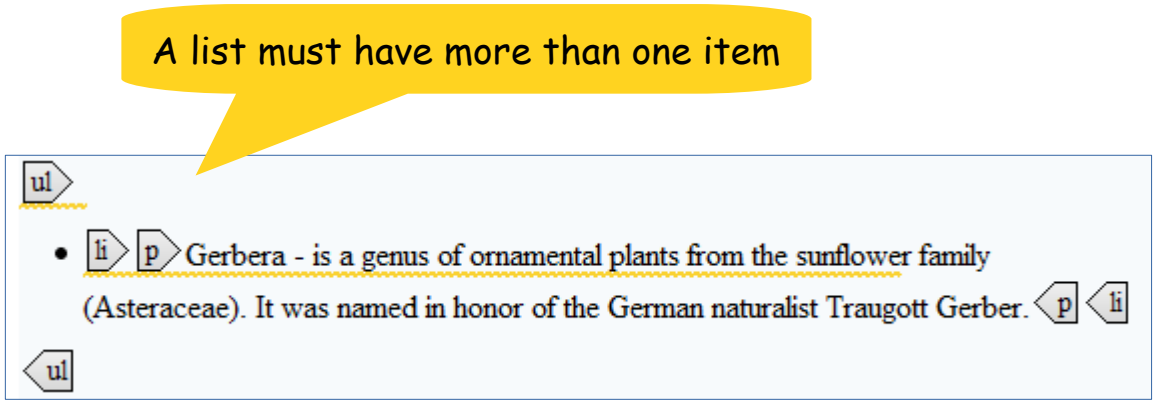
Conclusion

- Schematron is simple (6 basic elements)
- Used in multiple domains
- Schematron uses XPath

Examples of Schematron Rules

Check Number of List Items

- Example of a rule that checks the number of list items in a list



Check Number of List Items

- The number of list items from an unordered list should be greater than one

```
<sch:rule context="ul">
   <sch:assert test="count(li) > 1">
      A list must have more than one item</sch:assert>
</sch:rule>
```

Styling in Titles

- Example rule that checks for styling in titles

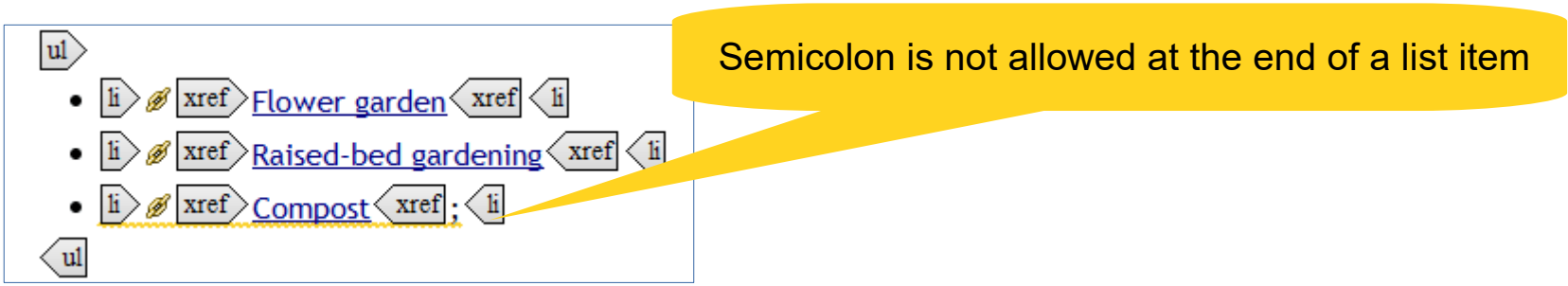
Styling in Titles

- The b element should not be used in a title

```
<sch:rule context="title">
    <sch:report test="b">
      Bold is not allowed in <sch:name/> element</sch:report>
</sch:rule>
```

Semicolon at End of List Items

- Example of rule that checks if a semicolon is at the end of a list item



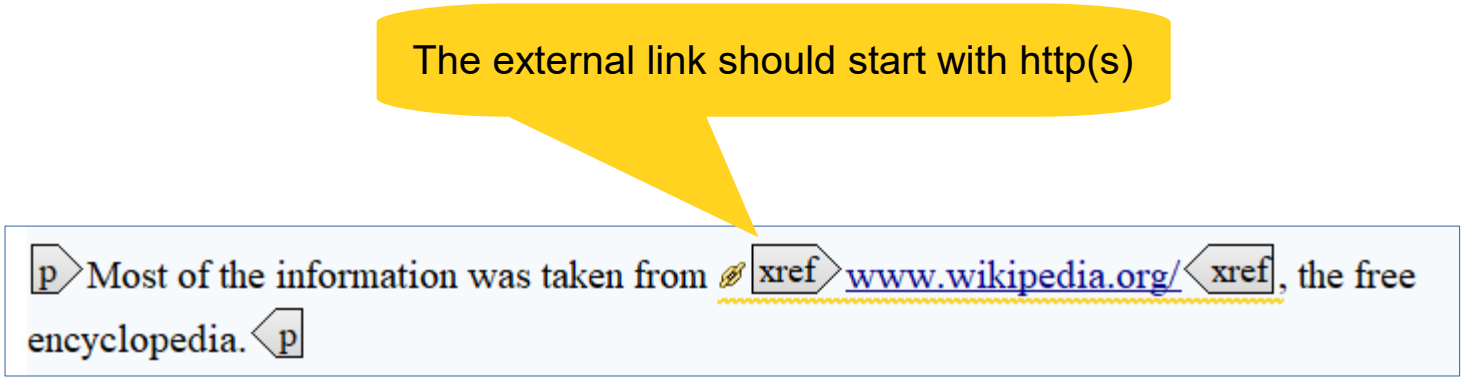
Semicolon at End of List Items

- The last text from a list item should not end with semicolon

```
<sch:rule context="li">
   <sch:report test="ends-with(text()[last()], ';')">
      Semicolon is not allowed after list item</sch:report>
</sch:rule>
```

Check External Link Protocol

- Example of rule that check the external link protocol



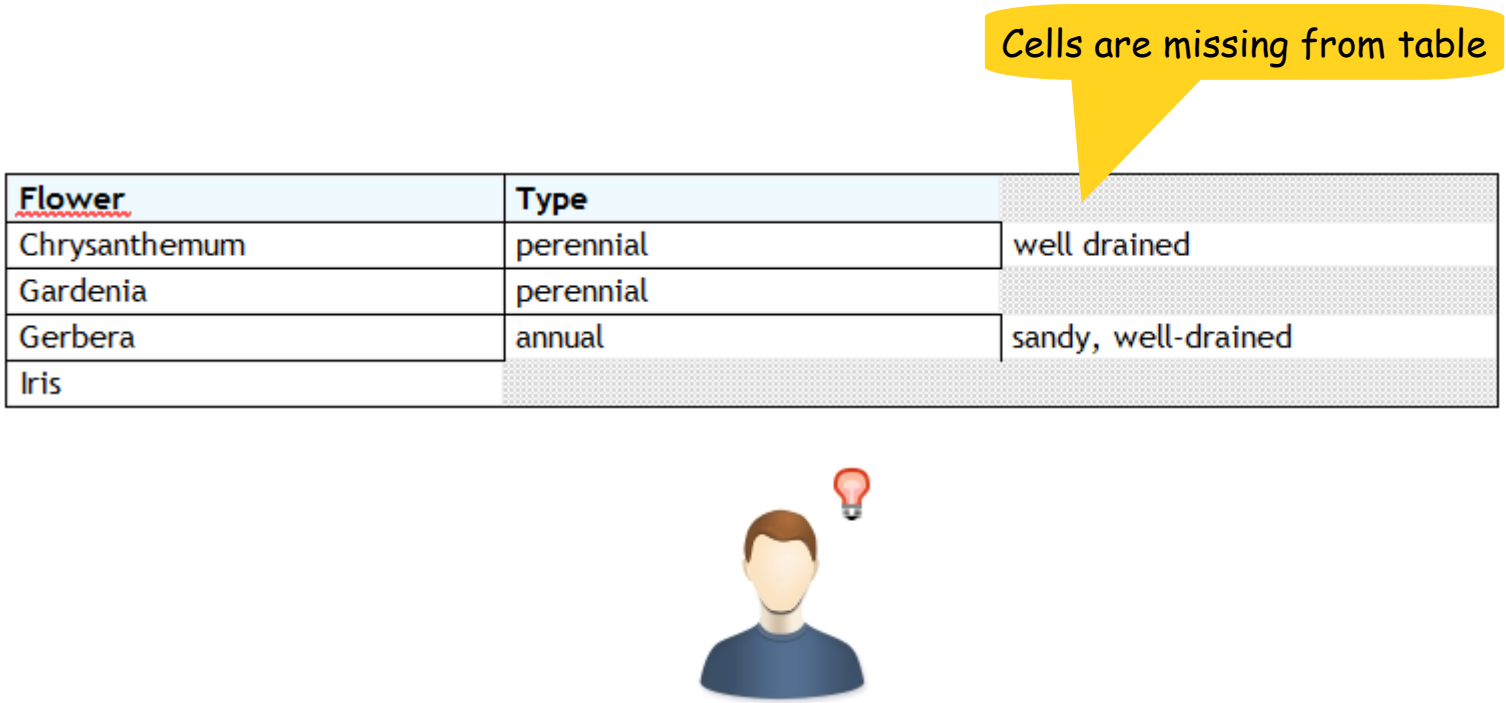
Check External Link Protocol

- The @href attribute value it is an external link and should start with http or https.

```
<rule context="xref">
   <sch:assert test="matches(@href, '^http(s?)://')">
     An link should start with http(s).</sch:assert>
</rule>
```

Missing Tables Cells

- Missing cells in a table



Missing Tables Cells

- Check that are the same number of cells on each row

```
<sch:rule context="table">
  <sch:let name="minColumsNo" value="min(//row/count(entry))"/>
  <sch:let name="reqColumsNo" value="max(//row/count(entry))"/>

  <sch:assert test="$minColumsNo >= $reqColumsNo">
    Cells are missing. (The number of cells for each row must be
    <sch:value-of select="$reqColumsNo"/>)
  </sch:assert>
</sch:rule>
```

Link in Text Content

- Links are added directly in text content

Link in Text Content

- Check if a link is added in text content but is not tagged as a link

```
<sch:rule context="text()">
  <sch:report test="matches(., '(http|www)\S+')
                    and local-name(parent::node()) != 'xref'">
    The link should be a xref element</sch:report>
</sch:rule>
```

Conclusion

- Simple to complex Schematron rules
- Additional Schematron elements used:
  - <let> - A declaration of a named variable
  - <value-of> - Finds or calculates values from the instance document
  - <name> - Provides the names of nodes from the instance document

Integrating Schematron in the Development Process

Validate XML with Schematron

- Associate Schematron in the XML file

```
<?xml-model href="books.sch" type="application/xml"
                schematypens="http://purl.oclc.org/dsdl/schematron"?>
```

- Use tool-specific association options
  - Associate Schematron file with a set of XML files (all files with a specific namespace, or from a directory)
  - Associate Schematron with all XML files from a project

Embed Schematron in XSD or RNG

- Schematron can be added in the XSD **appinfo** element

```
<xsd:appinfo>
  <sch:pattern>
    <sch:rule context="...">
        <sch:assert test="...">Message.</sch:assert>
    </sch:rule>
  </sch:pattern>
</xsd:appinfo>
```

- Add Schematron checks in any element on any level of an RNG schema

```
<grammar
    xmlns="http://relaxng.org/ns/structure/1.0"
    xmlns:sch="http://purl.oclc.org/dsdl/schematron"
    <sch:pattern>
      <sch:rule context="...">
        <sch:assert test="...">Message.</sch:assert>
      </sch:rule>
    </sch:pattern>
    <start>
     ..............
    </start>
</grammar>
```

Run Schematron Validation

- From an XML editing framework
  - Check the XML files as you type
  - Run the Schematron validation on multiple files
- Using W3C's XProc pipeline language through its "validate-with-schematron" step

- Using Apache Ant, from bat/shell

  https://github.com/Schematron/ant-schematron

Schematron Validation Result

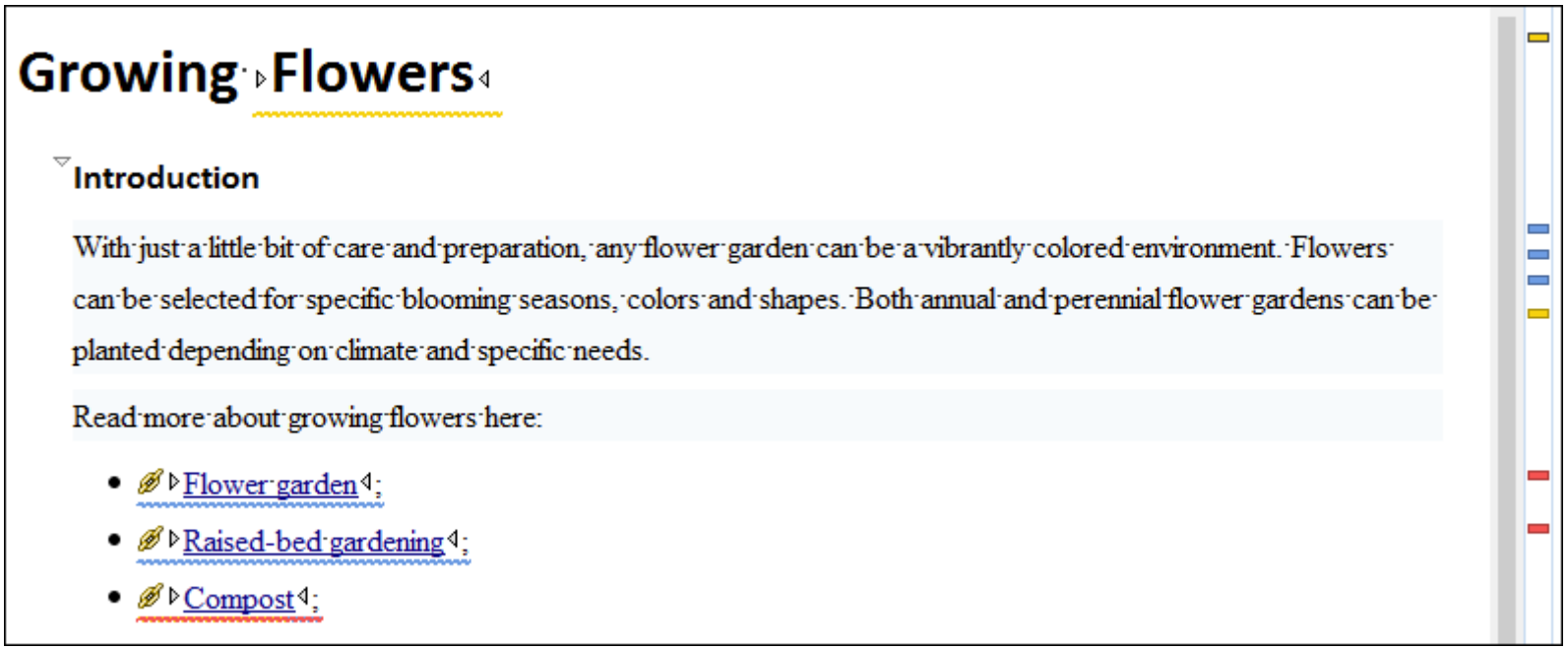
- Messages presented in the editing framework

- As an XML file describing the validation errors, normally in Schematron Validation Reporting Language (SVRL)

```
<svrl:failed-assert test="matches(@href, '^http(s?)://')" role="warn"
                    location="/topic[1]/body[1]/section[1]/p[3]/xref[1]">
   <svrl:text>An external link should start with http(s).</svrl:text>
</svrl:failed-assert>
```

Schematron Messages Severity

- Severity level can be set in the value of the @role attribute from an assert or report element (fatal, error, warn, or info)
- Depending on the severity, the message can be rendered differently in an editing framework



Multilingual Support in Schematron

- Based on the Schematron diagnostic element
- A diagnostic element is used for each language
- Multilingual support defined in the Schematron specification

```
<sch:assert test="bone" diagnostics="d_en d_de">
  A dog should have a bone.
 </sch:assert>
....
<sch:diagnostics  xml:lang="en">
   <sch:diagnostic id="d_en">A dog should have a bone.<sch:diagnostic>
</sch:diagnostics>
<sch:diagnostics  xml:lang="de">
   <sch:diagnostic id="d_de">Das Hund muss ein Bein haben.</sch:diagnostic>
</sch:diagnostics>
```

Conclusion

- Multiple ways to associate and apply the validation
- Validation results as you type or in a report
- Messages with different severity
- Multilingual support

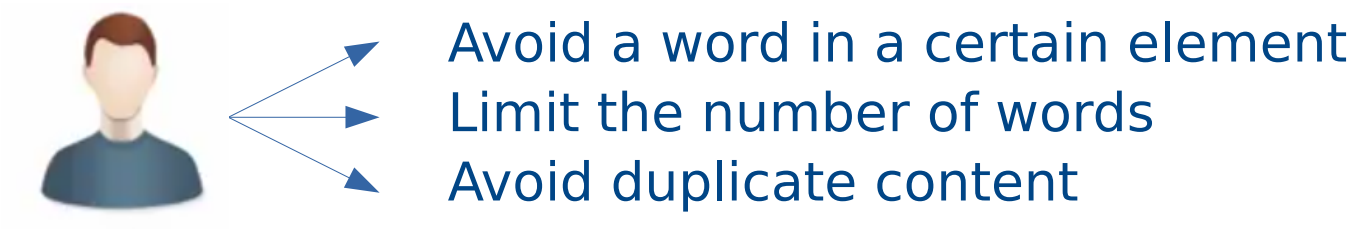
Schematron for Non-Technical and Technical Users

Schematron for Non-Technical Users

- Schematron is used in multiple domains
- There are more and more non-technical users that want to develop their own rules
- Providing a user interface to create the Schematron rules will be more appropriate for them

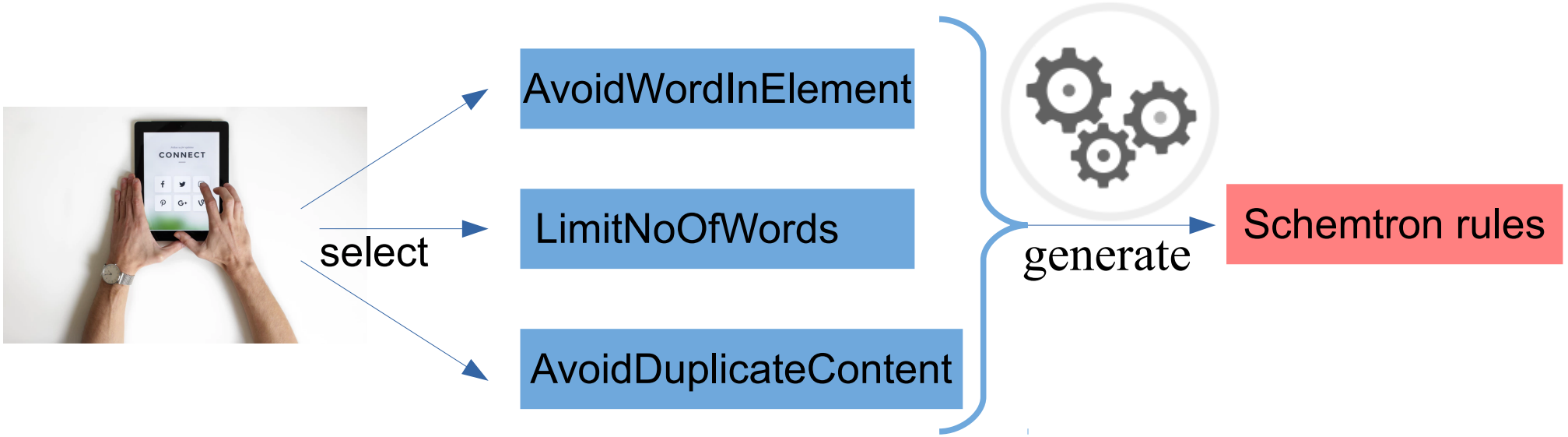
Library of Rules

- A solution can be to provide a library of rules
- The user can choose the rule that he wants to add



Abstract Patterns

- Library of rules implemented using abstract patterns



Abstract Pattern

- Allows to reuse patterns by making them generic

```
<pattern id="LimitNoOfWords" abstract="true">
   <rule context="$parentElement">
    <let name="words" value="count(tokenize(normalize-space(.), ' '))"/>
    <assert test="$words le $maxWords">
     $message You have <value-of select="$words"/> word(s).
    </assert>
    <assert test="$words ge $minWords">
     $message You have <value-of select="$words"/> word(s).
   </assert>
  </rule>
</pattern>
```

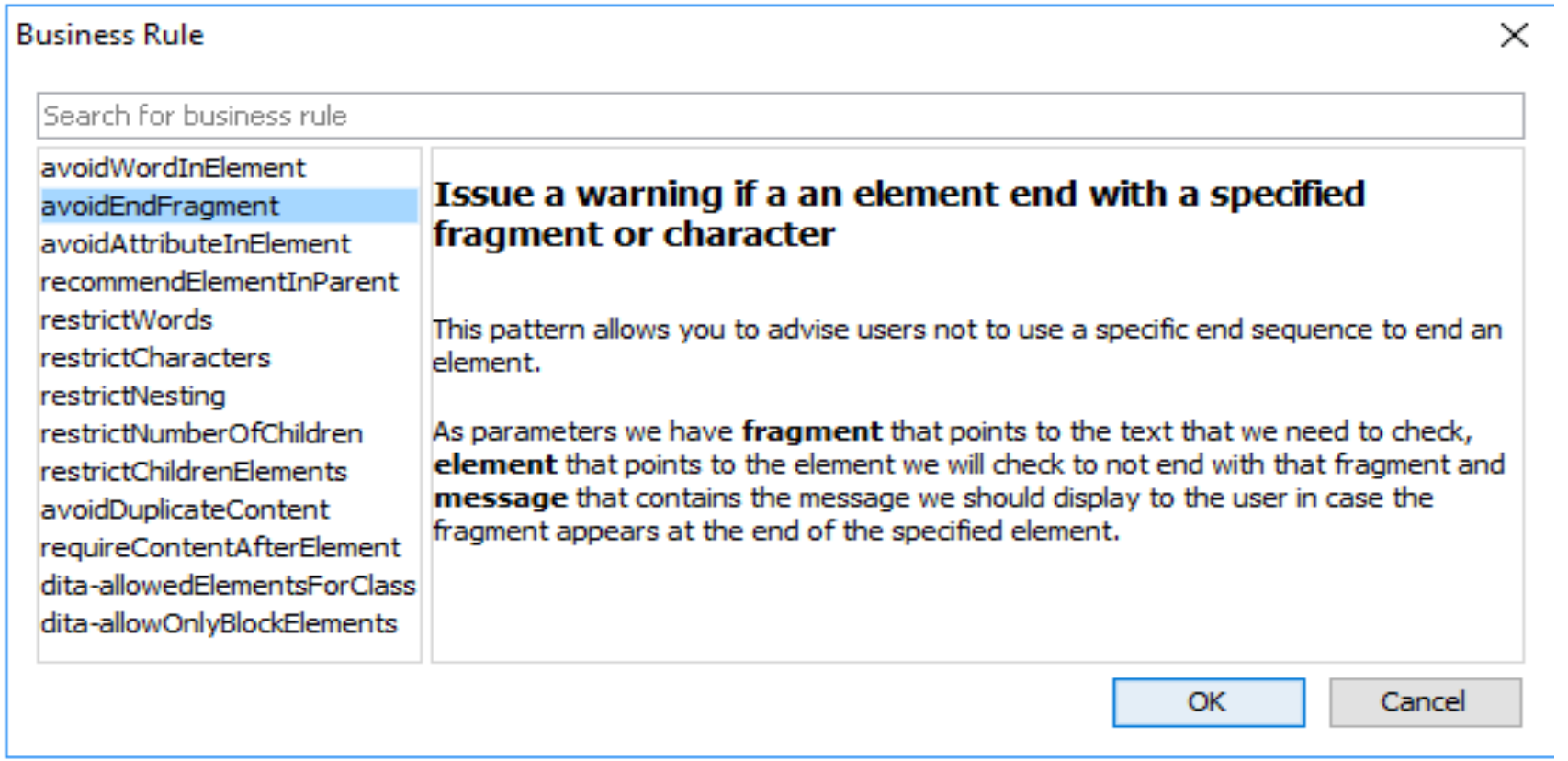
Pattern Instantiation

- Refer an abstract pattern and specifies values for parameters

```
<pattern is-a="LimitNoOfWords">
   <param name="parentElement" value="shortdesc"/>
    <param name="minWords" value="1"/>
    <param name="maxWords" value="50"/>
    <param name="message" value="Keep short descriptions between 1 and 50 words!"/>
</pattern>
```

```
<pattern is-a="LimitNoOfWords">
   <param name="parentElement" value="title"/>
     <param name="minWords" value="1"/>
     <param name="maxWords" value="8"/>
     <param name="message" value="Keep titles between 1 and 8 words."/>
</pattern>
```

Create Business Rules Using DIM



Rule	avoidEndFragment
element	li
fragment	;
message	Avoid ; at the end of a list item

Dynamic Information Model (DIM)

- An implementation of an intelligent style guide
- Describes and enforces rules
- Open-source project available on GitHub
  https://github.com/oxygenxml/dim

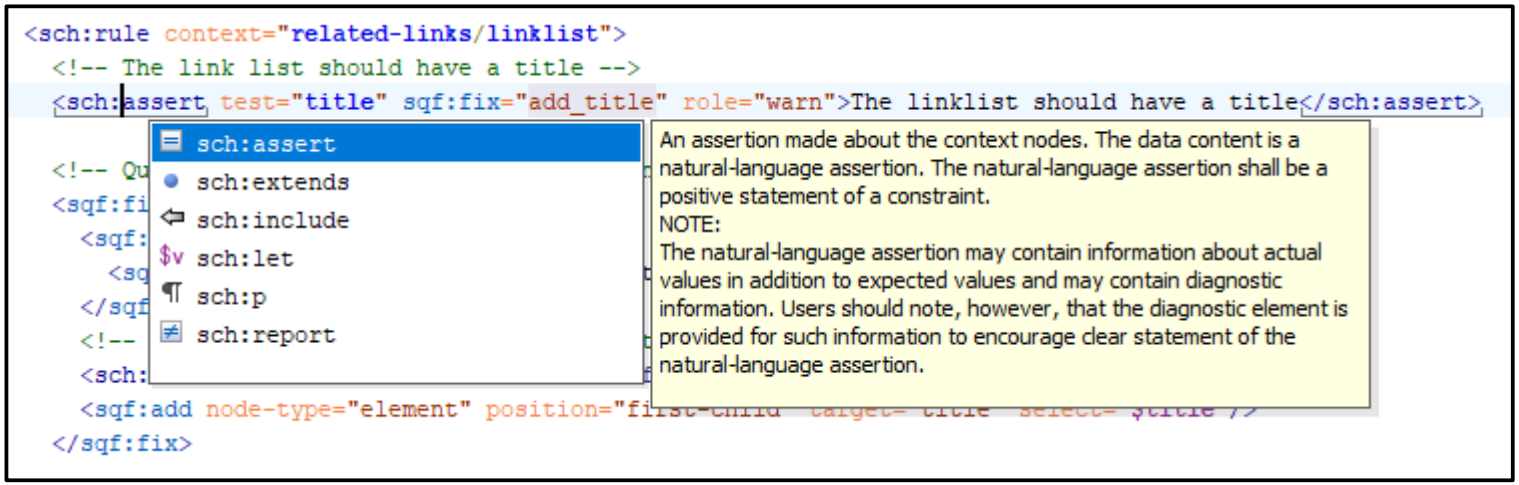
Schematron for Technical Users

Environment to develop Schematron schemas

- Editor
- Validation
- Search and refactoring
- Working with modules
- Unit testing

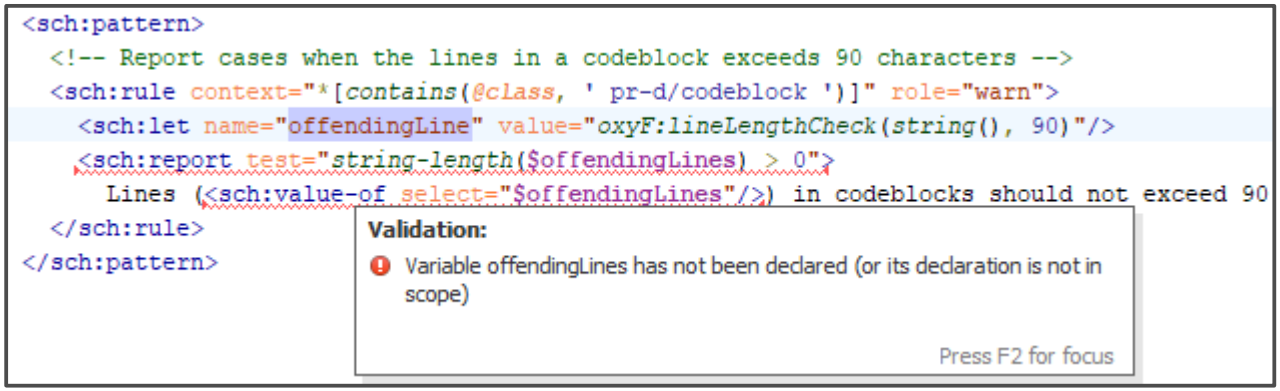
Editor

- Syntax highlighting - to improve the readability of the content
- Content completion assistant - offering proposals that are valid at the cursor position
- Documentation - with information about the particular proposal



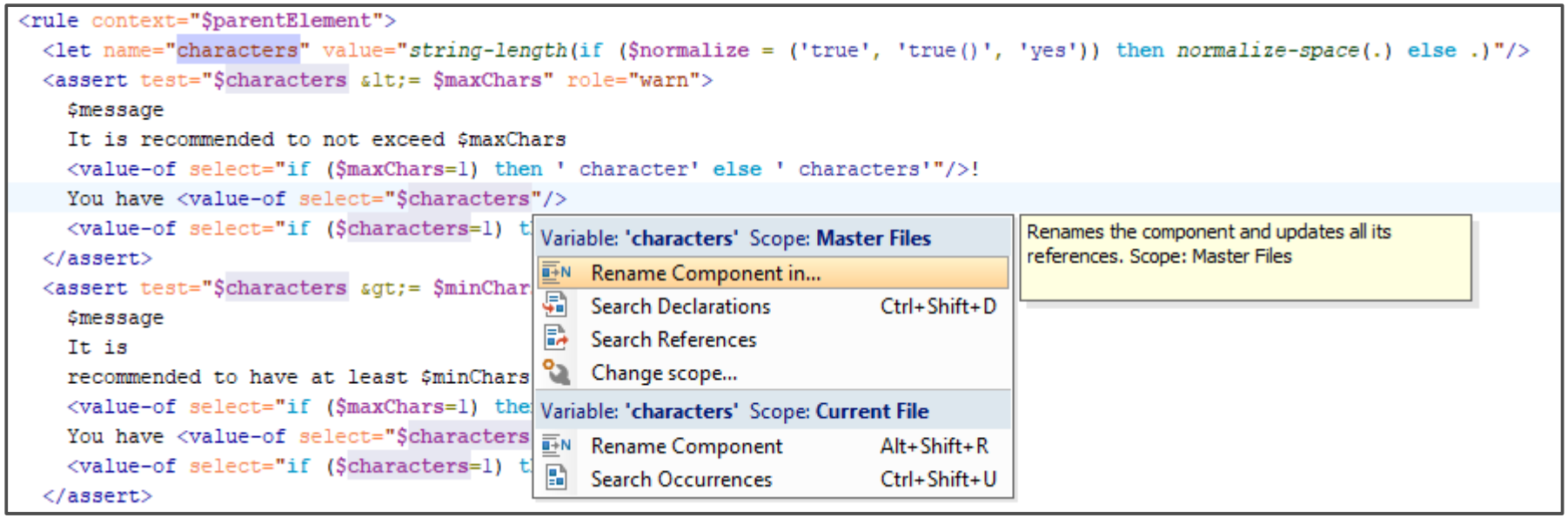
Validate

- Validate Schematron schemas, and highlight problems directly in the editor
- Validate Schematron modules in context
- Support for validation as you type



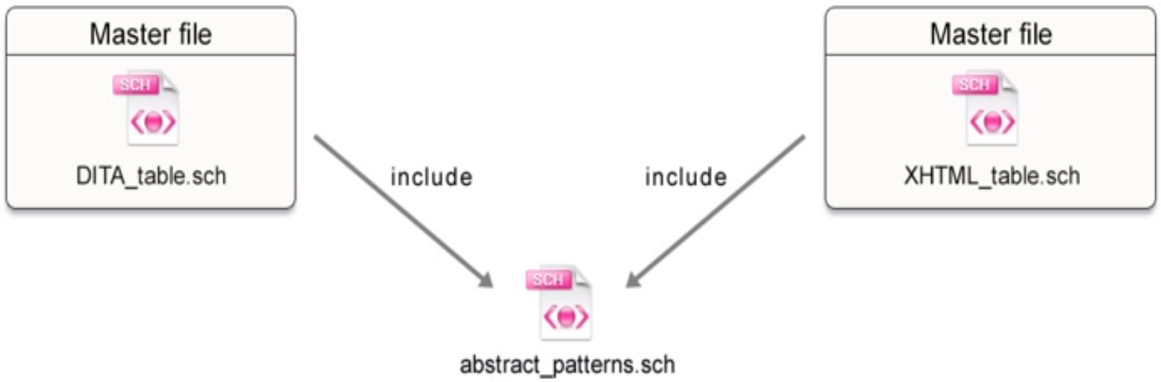
Search and Refactoring

- Find references to variables, pattern, phases, or diagnostics
- Support for renaming all references from the current document, or in the entire hierarchy
- Preview the changes



Working with Modules

- Visualize the hierarchy of modules
- Support for editing a Schematron module in the context
- Moving and renaming modules



Test Schematron Rules

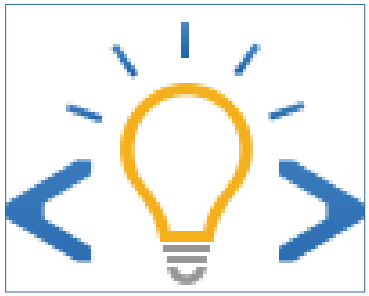
- Developing Schematron schema also involves testing
- Make sure Schematron rules work as expected
- Apply multiple XML examples over the Schematron rules and check the results

XSpec

- Solution to create Schematron unit tests
- Unit test and BDD (Behaviour Driven Development) framework
    - https://github.com/xspec/xspec/
- Oxygen XSpec Helper View plugin
    - github.com/xspec/oXygen-XML-editor-xspec-support
- Tutorials
    - github.com/xspec/xspec/wiki/Getting-Started-with-XSpec-and-Schematron
    - oxygenxml.com/events/2018/webinar_xspec_unit_testing_for_xslt_and_schematron.html

Conclusion

- Library of rules using abstract patterns
- User interface for non-technical users
- Editing, validation, and refactoring support for technical users
- Test cases for Schematron



Schematron Quick Fixes

Schematron Fix Proposals

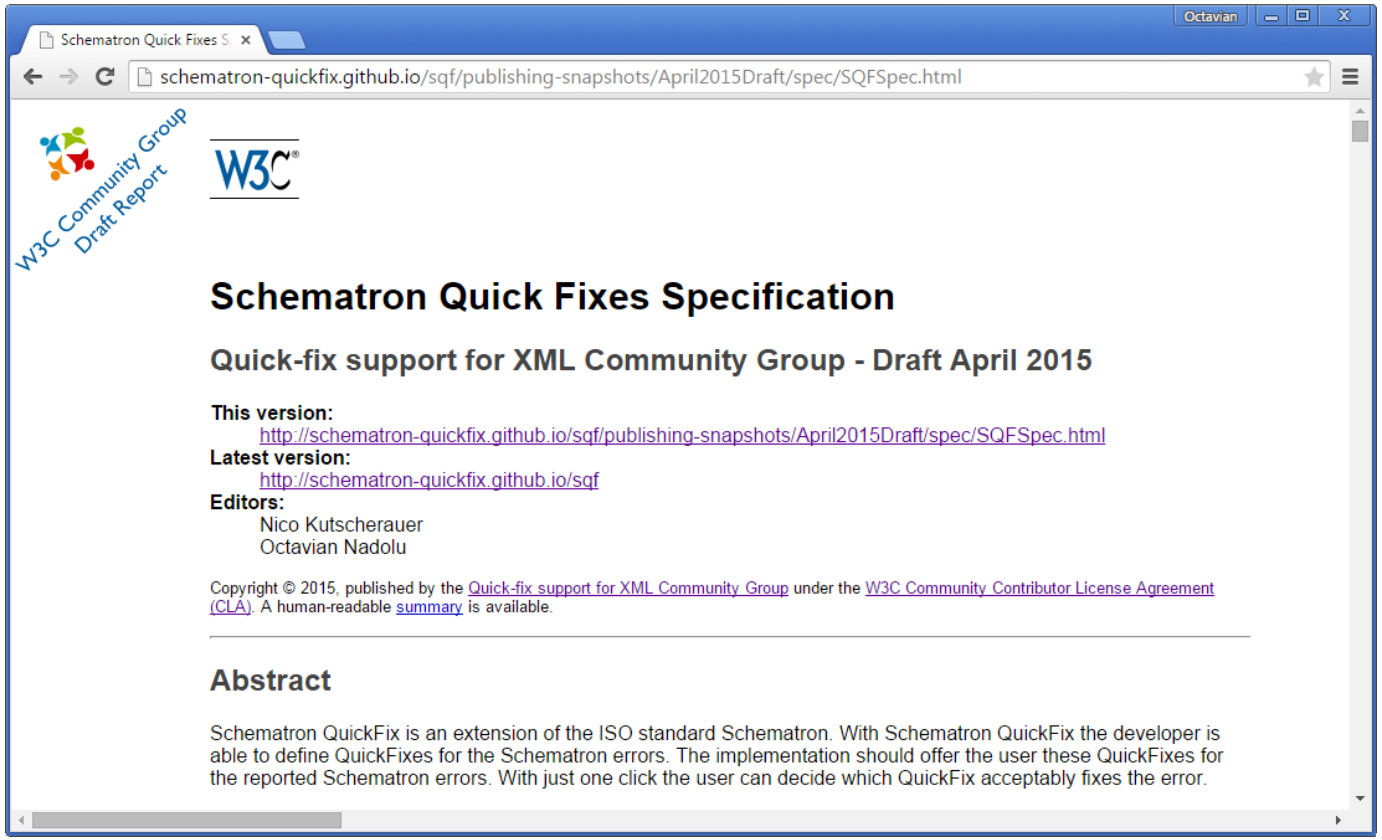
- Schematron assertion messages are not always enough for the user to find a solution
- It is better to have some proposals to correct the Schematron reported problem
- Similar to spell check proposals

Schematron Quick Fix Proposals

- User-defined fixes for Schematron errors
- Schematron Quick Fix (SQF) language
  - Extension of 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

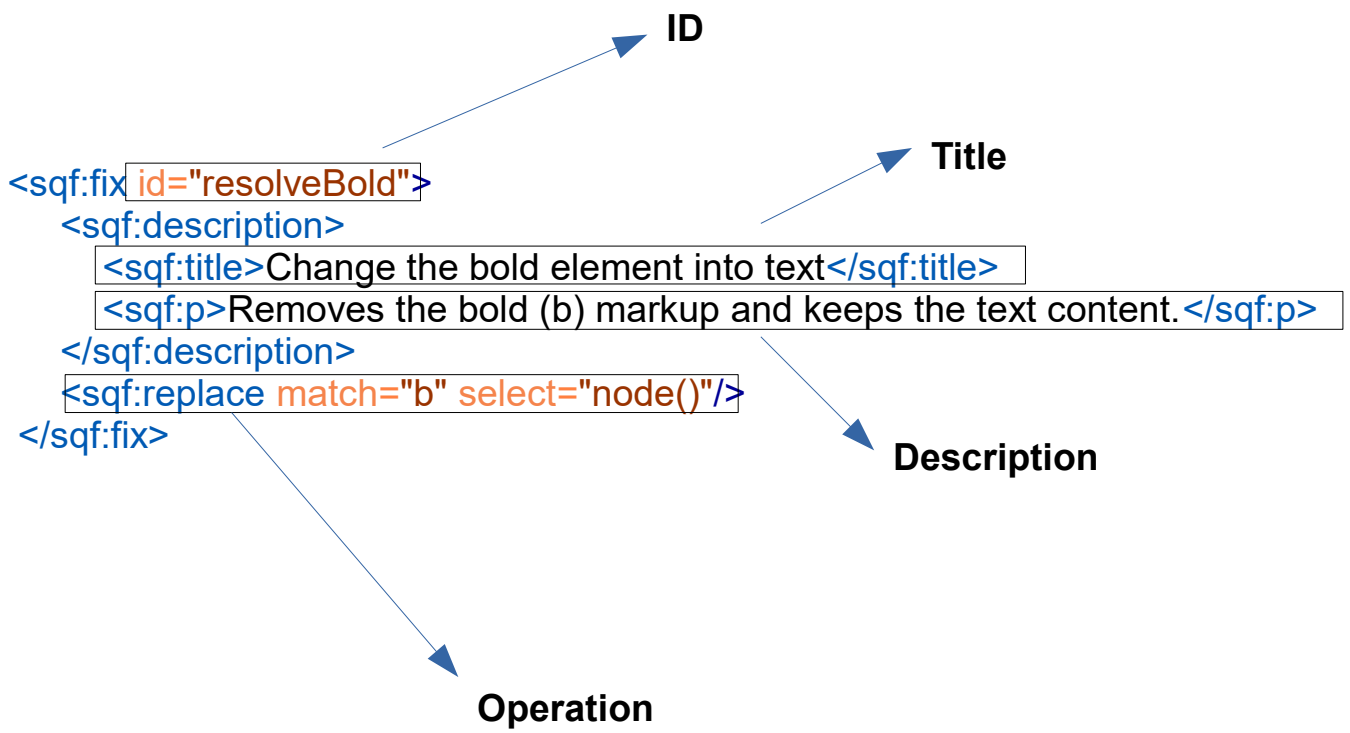
SQF Extension of Schematron

- Added as Schematron annotations
- Associate fixes with assert and report elements

```
<rule context="title">
    <report test="b" sqf:fix="resolveBold">
      Bold is not allowed in title element</report>

    <sqf:fix id="resolveBold">
      <sqf:description>
        <sqf:title>Change the bold element into text</sqf:title>
        <sqf:p>Removes the bold (b) markup and keeps the text content.</sqf:p>
      </sqf:description>
      <sqf:replace match="b" select="node()"/>
    </sqf:fix>
 </rule>
```

Schematron Quick Fix (SQF)



SQF Operations

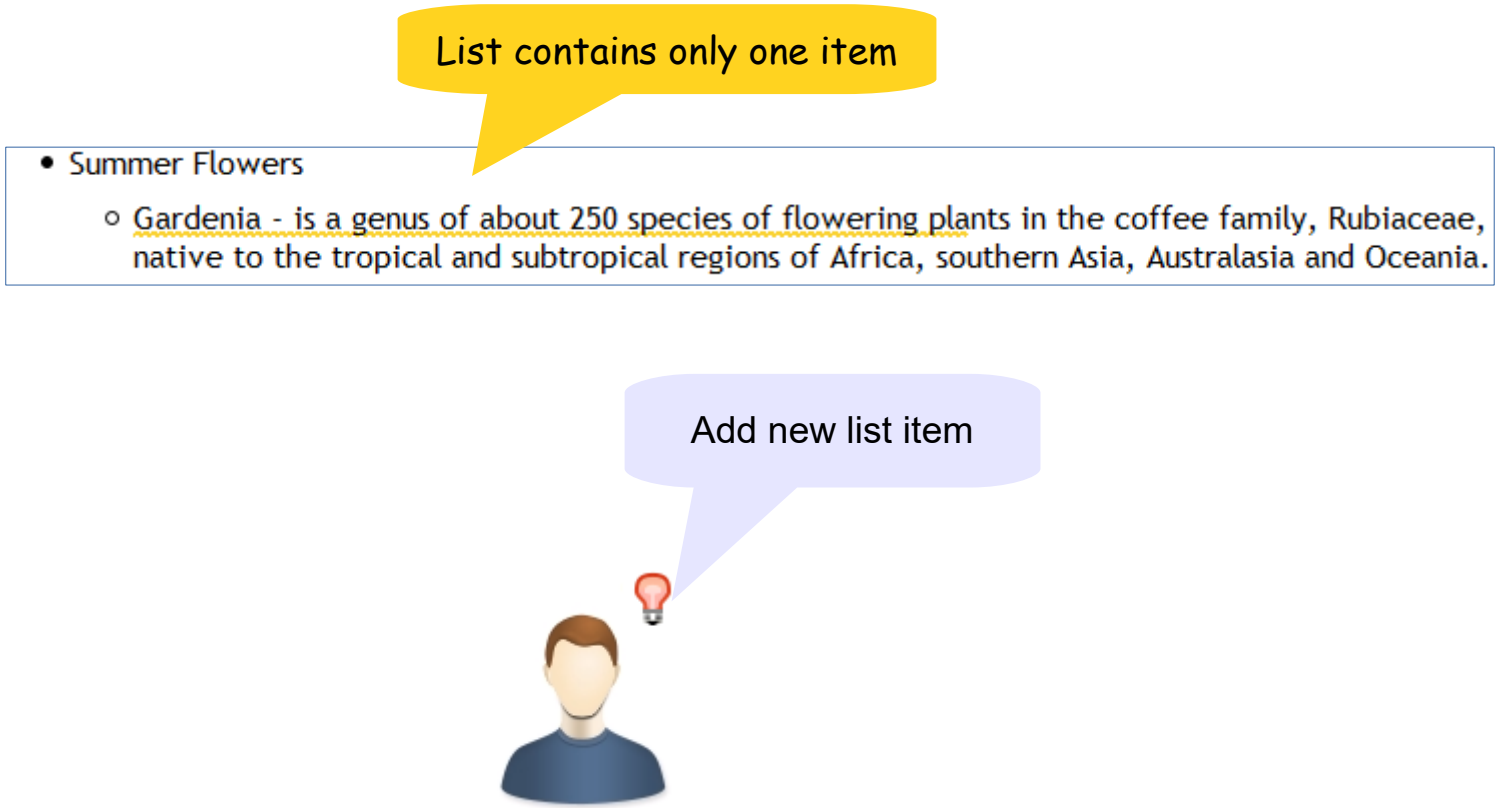
The following 4 types of operations are supported:

- <sqf:add> - To add a new node or fragment in the document
- <sqf:delete> - To remove a node from the document
- <sqf:replace> - To replace a node with another node or fragment
- <sqf:stringReplace> - To replace text content with other text or a fragment

Introduction to SQF Through Examples

1. SQF “add” Operation

- Example of using the “add” operation: add new list item in a list



1. SQF “add” Operation

- <sqf:add> element allows you to add one or more nodes to the XML instance

```
<rule context="ul">
   <assert test="count(li) > 1" sqf:fix="addListItem">A list must have more
  than one item.</assert>

   <sqf:fix id="addListItem">
      <sqf:description>
         <sqf:title>Add new list item</sqf:title>
      </sqf:description>
      <sqf:add node-type="element" target="li" position="last-child"/>
   </sqf:fix>
</rule>
```

2. SQF “delete” Operation

- Example of using the “delete” operation: remove redundant link text

2. SQF “delete” Operation

- <sqf:delete> element specifies the nodes for the deletion

```
<rule context="xref">
   <report test="@href = text()" sqf:fix="removeText">
      Link text is same as @href attribute value. Please remove.</report>

   <sqf:fix id="removeText">
      <sqf:description>
         <sqf:title>Remove redundant link text</sqf:title>
      </sqf:description>
      <sqf:delete match="text()"/>
   </sqf:fix>
</rule>
```

3. SQF “replace” Operation

- Example of using the “replace” operation: replace bold element with text

3. SQF “replace” Operation

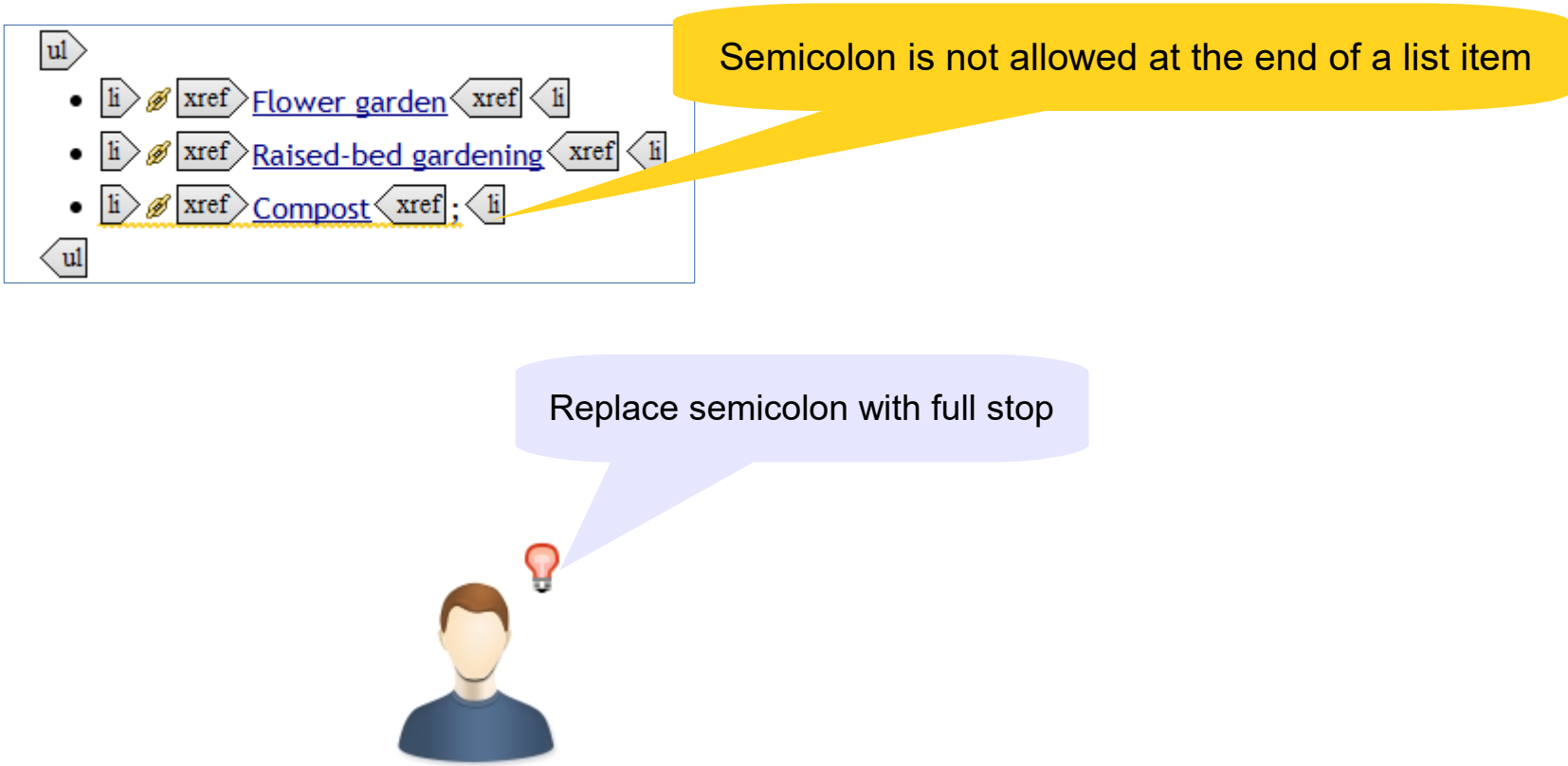
- <sqf:replace> element specifies the nodes to be replaced and the replacing content

```
<rule context="title">
   <report test="b" sqf:fix="resolveBold">
      Bold is not allowed in title element.</report>

   <sqf:fix id="resolveBold">
      <sqf:description>
         <sqf:title>Change the bold into text</sqf:title>
      </sqf:description>
      <sqf:replace match="b" select="node()"/>
   </sqf:fix>
</rule>
```

4. SQF “stringReplace” Operation

- Example of using the “stringReplace” operation: replace semicolon with full stop



4. SQF “stringReplace” Operation

- <sqf:stringReplace> element defines the nodes that will replace the substrings

```
<rule context="li">
  <report test="ends-with(text()[last()], ';')" sqf:fix="replaceSemicolon">
     Semicolon is not allowed after list item</report>

  <sqf:fix id="replaceSemicolon">
     <sqf:description>
        <sqf:title>Replace semicolon with full stop</sqf:title>
     </sqf:description>
     <sqf:stringReplace match="text()[last()]" regex=";$" select="'.'"/>
  </sqf:fix>
</rule>
```

Conclusions

- Schematron Quick Fix language is simple
- You can define custom fixes for your project
- Just 4 types of operations

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 manufacturer

Resources

- Schematron official site
- Schematron specification
- Schematron Quick Fix specification

- Sample files:
  github.com/octavianN/Schematron-step-by-step

THANK YOU!

Any questions?

<oXygen/> XML Editor
http://www.oxygenxml.com
octavian_nadolu@oxygenxml.com
@OctavianNadolu

tekom Jahrestagung 2018
STUTTGART, 13. – 15. NOVEMBER

tcworld conference 2018
STUTTGART, NOVEMBER 13 –15

Your opinion is important to us! Please tell us what you thought of the lecture. We look forward to your feedback under

http://ta14.honestly.de

or scan the QR code

The feedback tool will be available even after the conference!