Schematron QuickFix concept session during XMLPrague 2016

Heino Schmull, pagina GmbH
<heino.schmull@pagina-tuebingen.de>

Nico Kutscherauer, data2type GmbH
<kutscherauer@schematron-quickfix.com>

Octavian Nadolu, oXygen XML Editor
<octavian_nadolu@sync.ro>

Patrik Stellmann, GDV Dienstleistungs-GmbH & Co. KG <Patrik.Stellmann@gdv-dl.de>

Abstract

During the XMLPrague, we discussed in a small group current issues of the Schematron QuickFix language to prepare the second draft version of the specification.

In the following we want to update you by a summary of the discussed issues and our conclusions.

This article is based on some short notes which were made during the session.

Table of Contents

Setting

Participant List (Alphabetical Order)

- Heino Schmull
- Nico Kutscherauer
- Octavian Nadolu
- Patrik Stellmann

Issue Overview

- Dynamic fixes
- Call fix concepts

- Localisation concepts
- Nested processing of nodes by Activity Elements
- Additional substitution for common XSLT elements

Details

Dynamic fixes

Issues Summary

https://github.com/schematron-quickfix/sqf/issues/3

Conclusion

Everyone was satisfied by using Patrik's example.

Short summary:

- Dynamic fixes will be created by `sqf:fix` elements with an `use-for-each` attribute.
- The `use-for-each` attribute has an XPath as value. For each return value one fix should be generated.
- Context is still the same, with `$sqf:current` (build-in variable) you can have access to the current returned value.

Call fix elements

Issues Summary

The current problem was already described in this issue: https://github.com/schematron-quickfix/sqf/issues/2

Conclusion

- `sqf:description` is optional and unbounded, but the language (`@xml:lang`) needs to be unique in one `sqf:fix` element

  (if possible, auto completion should add it automatically when creating `sqf:fix` elements)
- The `sqf:fix` must have a description, if there is no or more than one `sqf:call-fix` with a description inside.
- If you have exactly one `sqf:call-fix` and no `sqf:description` element, the called fix must have a `sqf:description` or inherit one.
- The languages of the descriptions of the called fix and the calling fix could be different:

  Use case: Use `sqf:call-fix` to add a different language to an existing fix.

Open Issues

1. This concept ignores the localisation proposal below.
2. XSD has limitations, so the schematron-quickfix.xsd [https://github.com/schematron-quickfix/sqf/blob/master/schemas/schematron-quickFix.xsd] would be very lax. The order of the `sqf:fix` content would be checked by Schematron (sqf.sch [https://github.com/

schematron-quickfix/sqf/blob/master/schemas/sqf.sch]). One suggestion was, to switch the basic schema (schematron-quickfix.xsd [https://github.com/schematron-quickfix/sqf/blob/master/schemas/schematron-quickFix.xsd]) to RELAX NG

Localisation concepts

Issues Summary

- The concept above includes a localisation concept:
  - Multiple descriptions which have to be in different languages.
  - Concept for adding a description to an existing fix.
- There is no concept for referring to descriptions, which are stored in an external file (library).

Conclusion

There was no final concept for this issue, just a couple of proposals:

- Use Schematron localisation concept.
  - Add `@diagnostic` to `sqf:title` and `sqf:p`.
  - Remove the `@xml:lang` from the `sqf:description`.
  - Add `@diagnostic` to `sqf:description` with maybe multiple ids for localisation.
  - Multiple `sqf:title` with `@xml:lang`.
  - Only one `sqf:description` per `sqf:fix` element.

  Main goal would be to have all `sqf:title`/`sqf:p` in one external document.
- Design an own referencing concept for descriptions.
  - Something like `sqf:diagnostic`, but there was no further discussion about that.
- Allow to specify global descriptions and refer them.
  - Allow `sqf:description` in `sqf:fixes`.
  - Add `ref` and `id` attribute to `sqf:description`.
  - An empty `sqf:description` element with a `ref` attribute refers to a global `sch:description` element with an `id` attribute.

Open Issues

- The different proposals should be discussed.
- Maybe a localisation specialist should be called in.
- The discussion should be continued in this issue https://github.com/schematron-quickfix/sqf/issues/1

Nested processing of nodes by Activity Elements

Issue Summary

The first question was after a use case for the `sqf:keep` element:

The following example shows an Activity Element, which transforms all child elements of an specific namespace in the `NULL` namespace:

```
<sqf:replace match=".//sqf:*" target="{local-name()}" node-type="element">
    <sqf:keep/>
</sqf:replace>
```

Conclusion

- The group agreed, that this is a reasonable use case of `sqf:keep` and it is impossible to do this with other SQF structures.
- But because this is a very special case, we proposed to remove the `sqf:keep` element from the specification.
- The use cases are also realizable by embedded XSLT.
- Opened a new issue to discuss this https://github.com/schematron-quickfix/sqf/issues/14

Additional substitution for common XSLT elements

Issue Summary

- The issue resulted by removing the `sqf:keep` element.
- It was noticed, that there is a huge usage of the `xsl:copy-of` element (instead of `sqf:keep`).

Conclusion

- Introduction of a `sqf:copy-of` element, because it (`xsl:copy-of`) will be used really often to copy existing nodes.
- For non-XSLT user, it is more understandable, if he don't have to change the namespace to copy nodes (like `xsl:value-of` -> `sch:value-of`).
- No default for the `select` attribute, because there are two reasonable options (`.` and `node()`).

Open Issues

This issue popped up, after the session: https://github.com/schematron-quickfix/sqf/issues/9