Optimizing Time-Performance of Streaming Schematon

Rick Jelliffe, 2002-08-19

Heuristic for improving the performance of Schematron implementations which allow evaluation of individual elements

The intent of these rules is to give a validation result as soon as possible during the parse of a document. This would be of use to using Schematron to match or validate incoming documents.

1. Score each @context and @test according to the maximum number found, so that every @context or @test gets a score 0,1 or 2.

- ancestor=0
- ancestor-or-self=0
- parent=0
- self=0
- attribute=0
- namespace=0
- previous=0
- previous-sibling=0
- preceding=0
- namespace=0
- child=1
- descendent=1
- descendent-or-self=1
- following-sibling=2
- following-sibling=2 (document order)

2. If any path uses key() or id() it has a score of 2
3. Create an “up-looking schema”:

- For each pattern, remove all rules after the first rule with a score >0.
- For each rule, remove all assertions with a score >0
- Remove any empty rules or patterns

4. Create a “down-looking schema”
- For each pattern, remove all rules after the first-rule with a score of 2
- For each rule, remove all assertions with a score of 2
- If a rule has a score = 0, remove any assertions =0
- Remove any empty rules or patterns

5. Create a “document-end schema”
- If all rules have a score < 2, remove the pattern
- If the rule has a score <2, remove all assertions with score <2
- Remove any empty rules or patterns

6. Parse the document using a SAX parser, building a DOM dynamically.
7. At the close of each start tag, evaluate the DOM so far against the “up-looking schema”. At the arrival of a close tag, evaluate against the “down-looking schema.” At the finish of the document, evaluate against the “document-end schema.”

An optimization possible if the document has been validated against a grammar first:

- For every element, identify the maximum possible number of occurrences for that element in a document.
- For any rule that has required self of that element, after the first match, disable that rule.
- For any elements where there is a sequence in the grammar a,b and b can only appear after a, if there are any patterns where one rule has a required self of a and a latter rule has a required self of b, disable rule b until rule a has fired. More complex, disable a, and then let the firing of rule b re-enable rule a.