Incremental Transformation for XML Document Manipulation

Lionel Villard
lionel.villard@inrialpes.fr

Opéra Project – INRIA Grenoble - France
http://www.inrialpes.fr/opera

Goal: Interactive authoring of

• XML source documents

• XML presentations

trough one or many presentations
Incremental transformation
To update as fast as possible the document after modifications of:
- XML source document(s)
- Transformation sheet(s)

1: Batch Transformation
2: Authoring
3: Reverse Transformation
   Determine source/transformation modification
4: Incremental transformation
1. Incremental transformation: static analysis of transformation sheets

**Goal:** determine statically which XSLT instructions need to be re-evaluated when:

- **The source document changes**
  - Modification of attribute values, addition of elements, etc.

- **A transformation sheet change**
  - Input parameters, addition/removal of template, addition/removal of instruction

**Result of the analysis:** set of re-evaluation rules (pattern, XSLT instructions)
Main steps of analysis

A. Creation of template/apply-templates dependency graph

B. For each instruction with expression:
identify the re-evaluation conditions and express them as a pattern

1: Extract expressions

2: Remove dynamic context references

3: Determine patterns

4: Use instruction declaration context
2. Incremental transformation: execution step

Standard transformation execution with:

- Only a **subset** of instructions are executed (thanks to re-evaluation rules)
- Compute processor context **as needed**
- Use the **minimal** execution flow tree
- **Incremental** execution of XSLT instructions:
  - `value-of`: replace characters
  - `apply-templates`:
    - use execution flow tree for retrieving template instantiation
    - re-instantiation is done if needed
      (thanks to re-evaluation rules)

Example: insert a title in slide

only the red instruction is re-evaluated
Experimentation / Evaluation

- Implementation of a WYSIWYG docbook editor
- Incremental transformation processor implemented in Xalan 2.0 (AFS)
- First evaluation results

<table>
<thead>
<tr>
<th></th>
<th>Batch</th>
<th>Dummy</th>
<th>Change title</th>
<th>Insert section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of instruction to re-execute</td>
<td>N/A</td>
<td>0</td>
<td>795</td>
<td>819</td>
</tr>
<tr>
<td>Time to get instruction to re-execute</td>
<td>N/A</td>
<td>0</td>
<td>80ms</td>
<td>80ms</td>
</tr>
<tr>
<td>Variables value computed</td>
<td>6572</td>
<td>6572</td>
<td>6572</td>
<td>6572</td>
</tr>
<tr>
<td>Variable access count</td>
<td>10279</td>
<td>6899</td>
<td>6899</td>
<td>6983</td>
</tr>
<tr>
<td>Overall timing / ratio</td>
<td>4.5s</td>
<td>1</td>
<td>2.8s</td>
<td>0.6</td>
</tr>
</tbody>
</table>

Speed costs of the transformations applied to Norman Walsh’s docbook transformations sheets (2219 instructions and 1200 templates)

Interpretation

- Incremental transformation is half time as batch transformation
- But it is still too slow partly because all variables are evaluated (to be done)