

# An Incremental XSLT Transformation Processor for XML Document Manipulation



Lionel Villard – Nabil Layaida

Presented by Emmanuel Pietriga

Projet Opéra

INRIA Rhône-Alpes

<http://www.inrialpes.fr/opera/>



# Outline

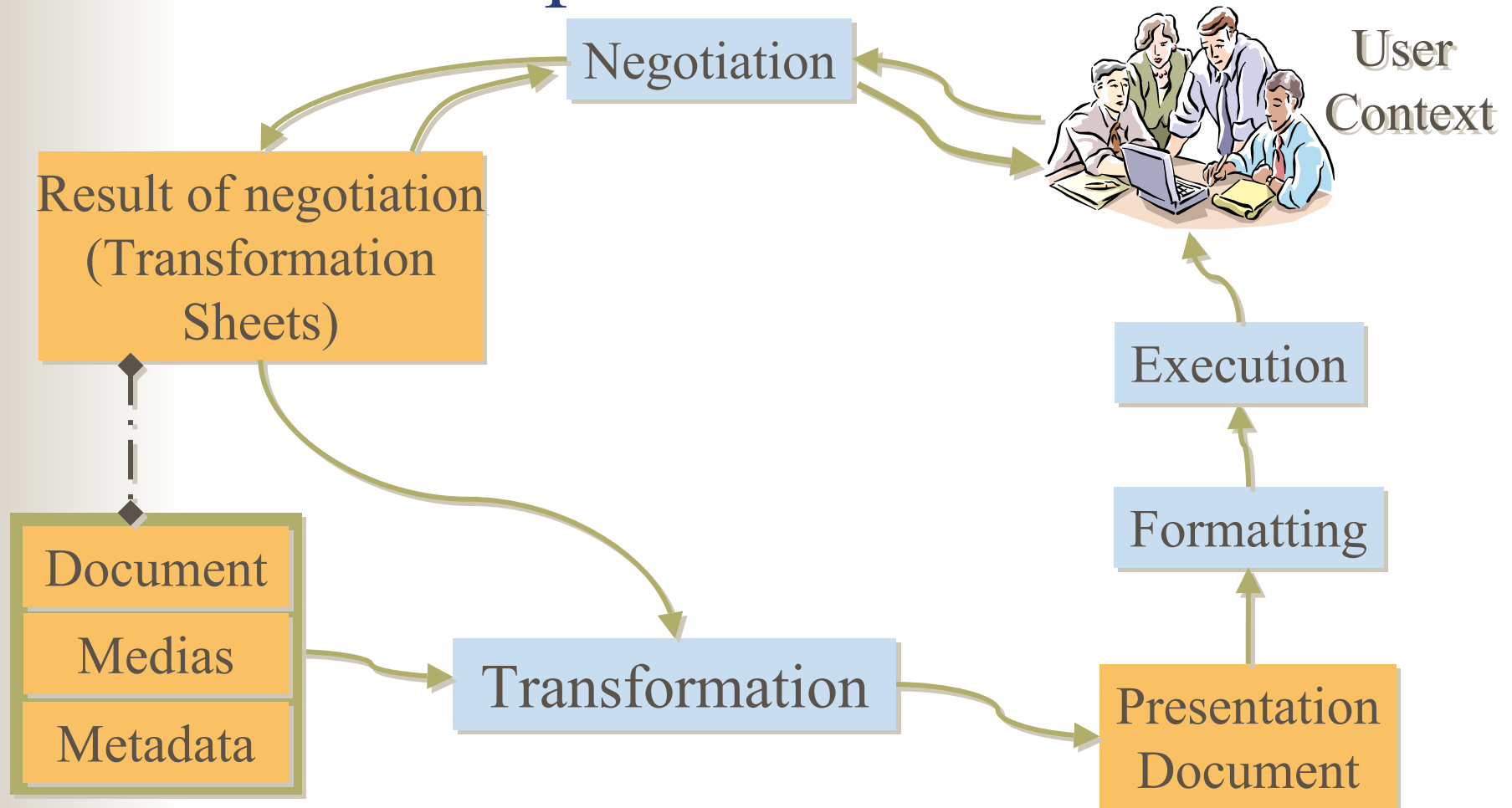
- Motivation
- Incremental transformations
  - Principles
  - Static Analysis
  - Incremental execution
- Conclusion and perspectives



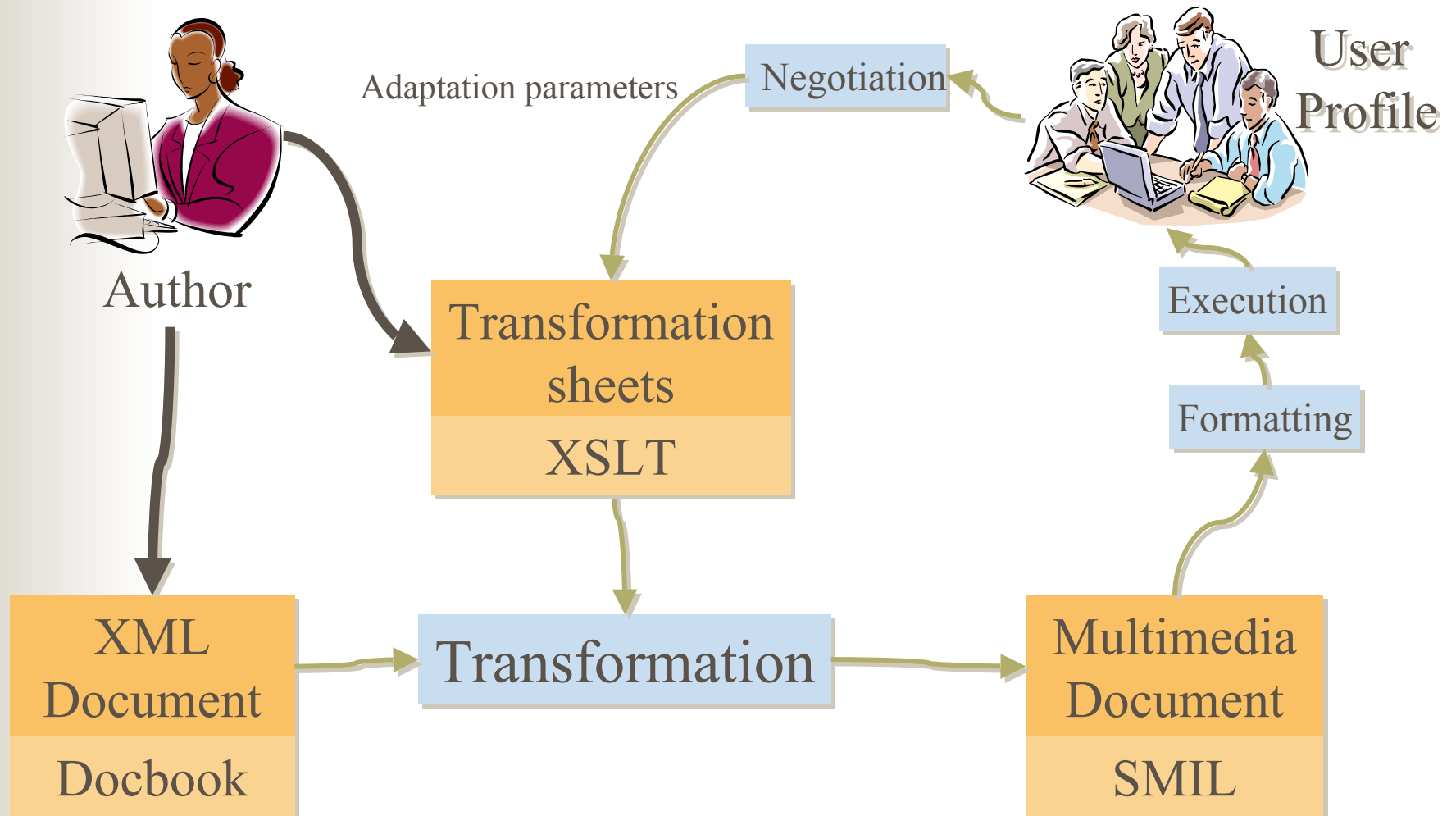
# Motivation

- XML Vocabularies
  - Content (logical structure)
  - Presentation (formatting)
- Authoring multimedia presentations for classes of document
- Authoring adaptable presentations

# Multimedia presentation architecture



# Author involvement





# Author skills

- Classes of document
  - Generally for professionals
  - Need heavy infrastructure (database, schema)
- Adaptable multimedia presentation
  - For any author (novice, professional, etc.)

No transformation coding



# Goals

- Author source (content) document and transformation sheet by direct manipulation of target (presentation) document





# Goals

- Author source (content) document and transformation sheet by direct manipulation of target (presentation) document
- Update as fast as possible the target document after modifications of:
  - XML source document(s)
  - Transformation sheet(s)



# Incremental transformation processor

- Change only target document fragments that need to be updated w.r.t changes in the source document or in the transformation sheet
- Focus on target document updates due to changes in the source document



# XSLT transformations

## Source document:

```
<Picture loc="result.jpg" l="100" h="150">
```

## Transformation rule:

```
<xsl:template match="Picture">  
  <Image src="{@loc}" width="{@l}" height="{@h}">  
</xsl:template>
```

## Result :

```
<Image src="result.jpg" width="100" height="150">
```



# Source modification

## Source document:

```
<Picture loc="result.jpg" l="300" h="150">
```

## Transformation rule:

```
<xsl:template match="Picture">  
  <Image src="{@loc}" width="{@l}" height="{@h}">  
</xsl:template>
```

## Result :

```
<Image src="result.jpg" width="300" height="150">
```



# incXSLT: incremental transformation

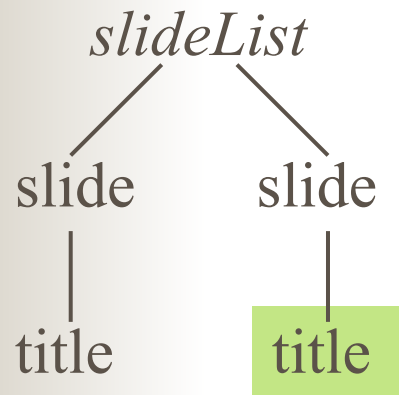
Two step process:

- Preprocessing: static transformation sheet analysis
  - Build template and variable dependency graphs
  - Build re-evaluation rules  
(editing operations, instructions to re-evaluate)
- Incremental processing
  - Run instructions identified thanks to the rules computed during the static analysis

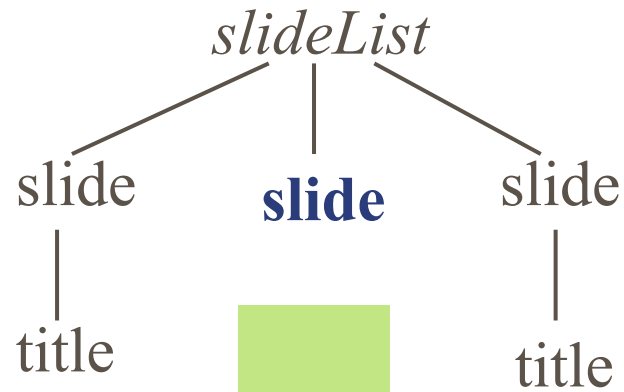
# Static analysis example

**Expression:** `slide[position() = 2]/title`

Initial  
evaluation

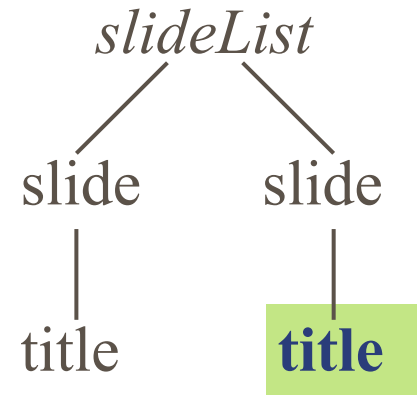


Addition/Removal  
of slide before  
the second position



**Pattern:**  
`slide[position() <= 2]`

Addition/Removal  
of second slide title



**Pattern:**  
`slide[position()=2]/title`

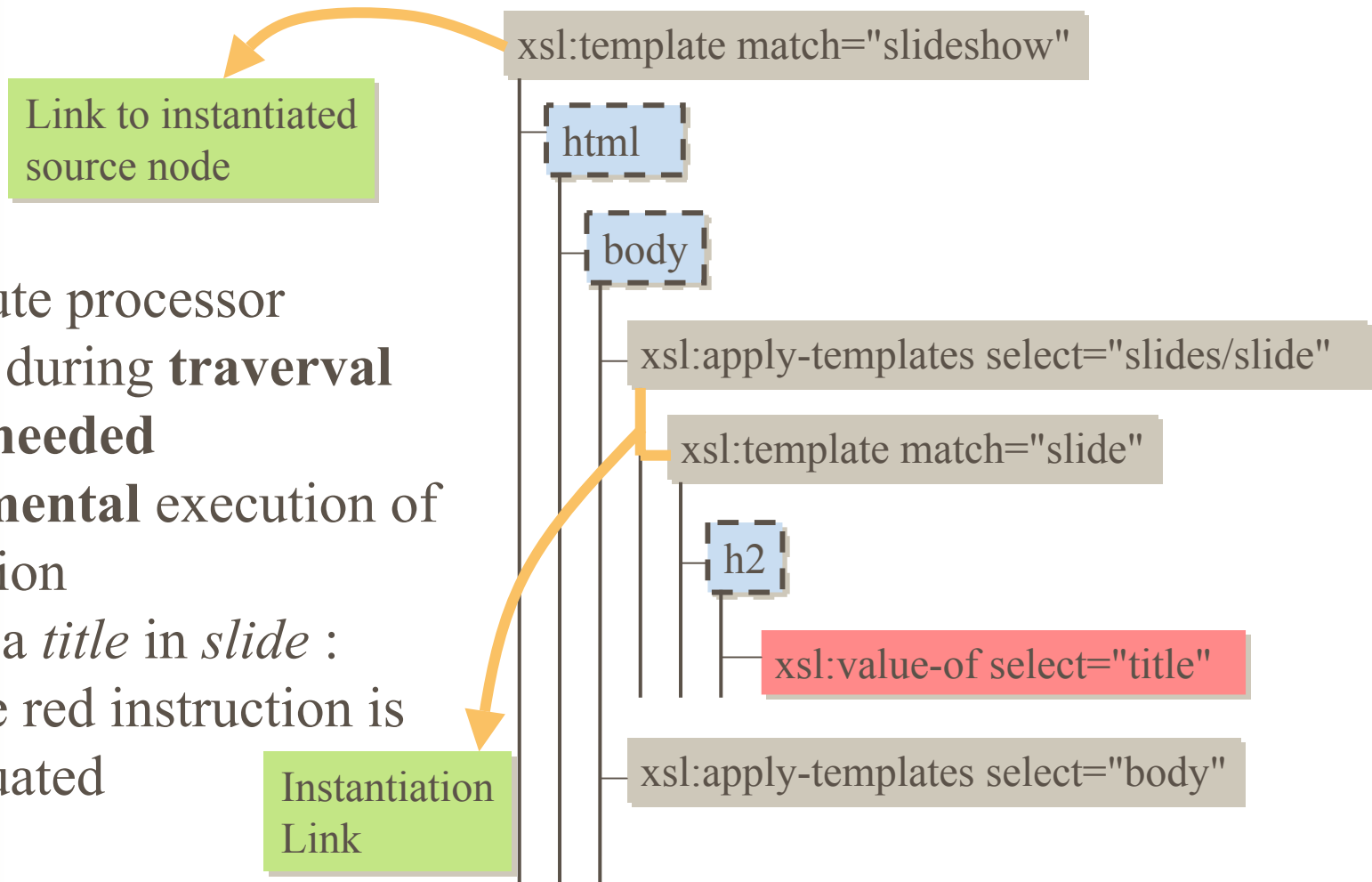


# Incremental execution

- Set of instructions to re-evaluate, determined during the static analysis
- Restore execution state: execution flow tree
  - Cache data structure
- Two execution models
  - Depth-first execution tree traversal
  - Direct execution of instructions

# Execution tree traversal

- Compute processor context during **traversal** and **as needed**
- **Incremental** execution of instruction
- Insert a *title* in *slide* : only the red instruction is re-evaluated





# Conclusion

- Experimentation in Xalan (XSLT engine from Apache)
  - Important execution gain
  - Limited memory overhead
- Other applications
  - Batch transformation optimization
  - Web site construction
  - Transformation parameter modifications



# Perspectives

- Go further about static analysis
  - Consider schema of source document
- Apply to XPath/XSLT 2.0
  - Propose a formal semantics
  - Enhance static analysis
  - Implications on incremental execution