Incremental Transformation for XML Document Manipulation

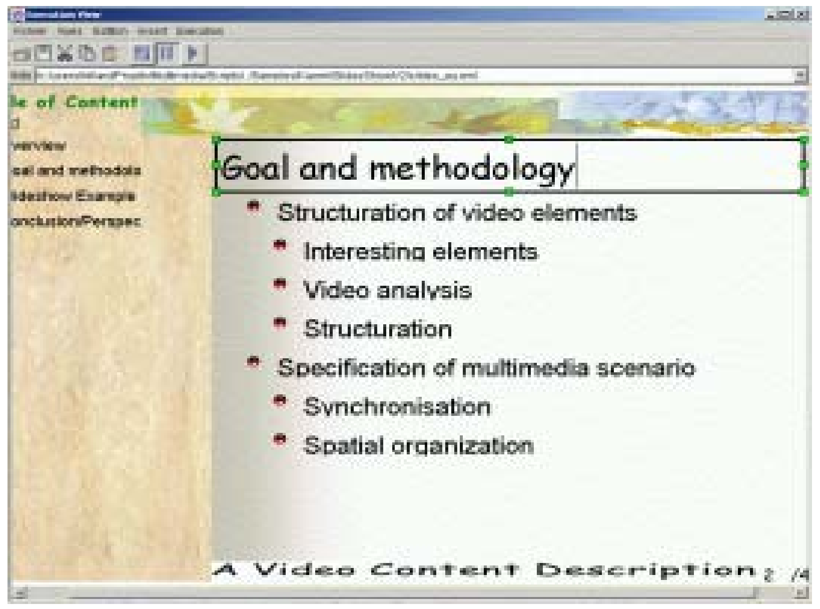
Lionel Villard
lionel.villard@inrialpes.fr

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

Goal: Interactive authoring of

- **XML source documents**
- **XML presentations**

through one or many presentations

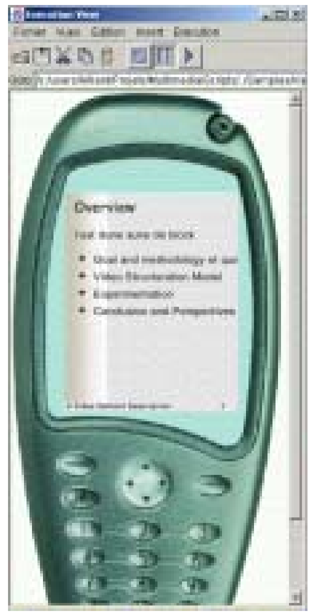


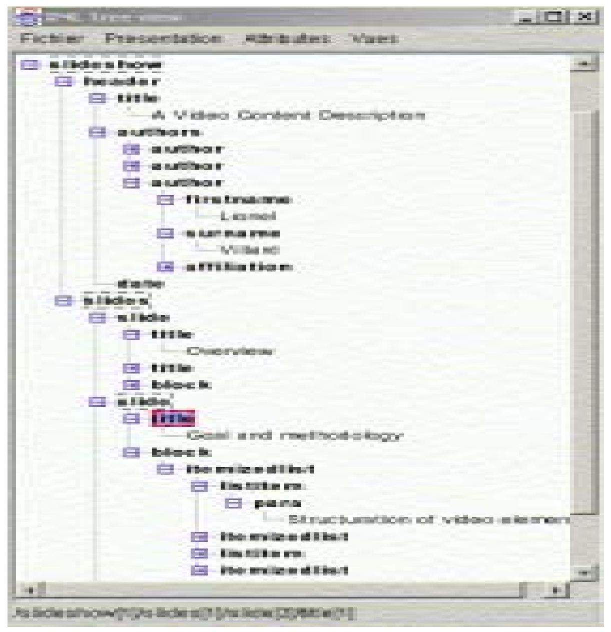
le of Content
d
verview
oal and methodolo
ideshow Example
onclusion/Perspec

Goal and methodology

- Structuration of video elements
  - Interesting elements
  - Video analysis
  - Structuration
- Specification of multimedia scenario
  - Synchronisation
  - Spatial organization

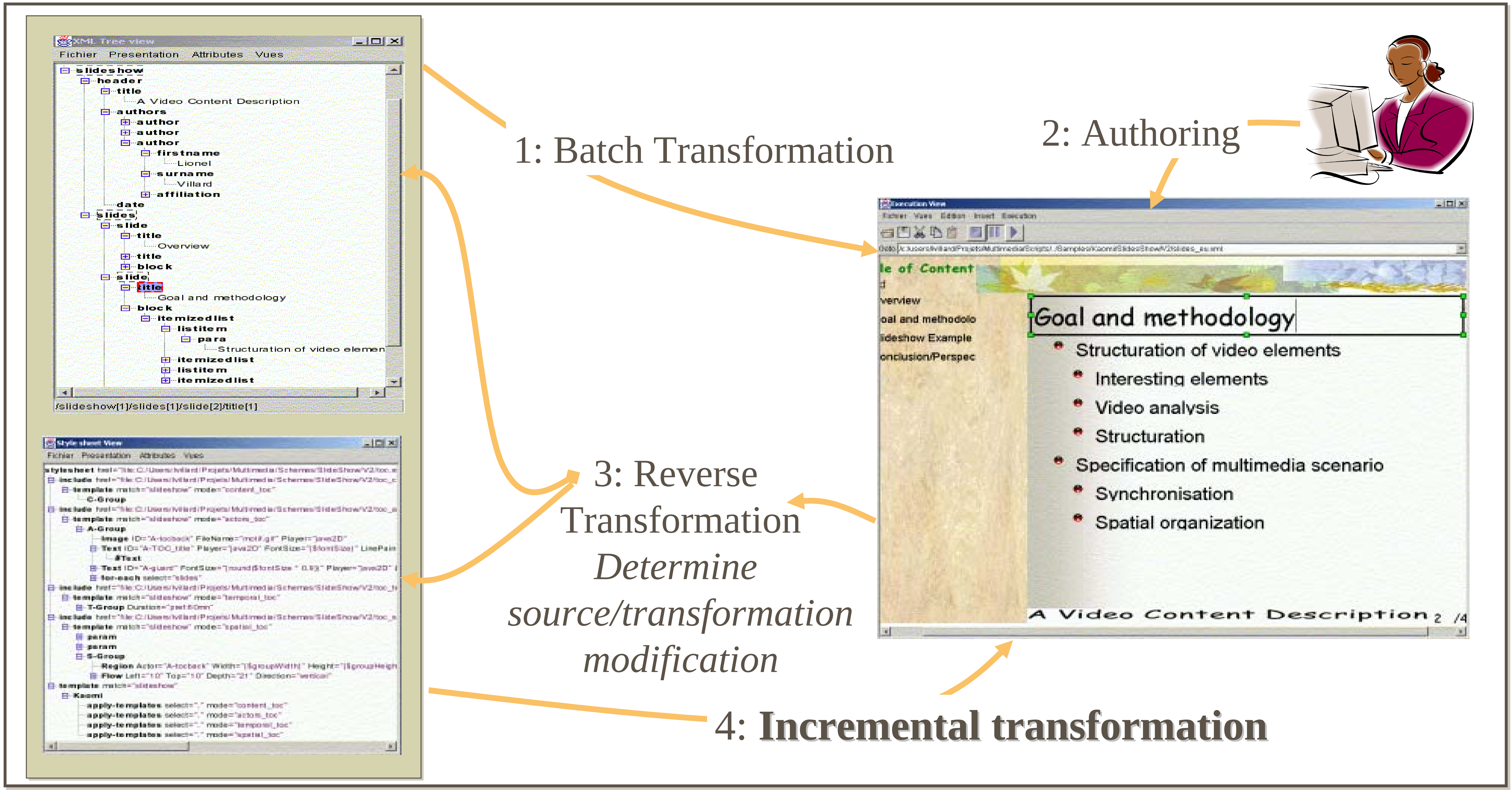
A Video Content Description 2 /4





```
slideshow
  header
    title
      A Video Content Description
    authors
      author
      author
      author
        firstname
          Lionel
        surname
          Villard
        affiliation
    date
  slides
    slide
      title
        Overview
      title
      block
    slide
      title
        Goal and methodology
      block
        itemizedlist
          listitem
            para
              Structuration of video elemen
          itemizedlist
          listitem
          itemizedlist
```

/slideshow[1]/slides[1]/slide[2]/title[1]


1: Batch Transformation
2: Authoring
3: Reverse Transformation
Determine source/transformation modification
4: **Incremental transformation**
XML Tree view
Fichier Presentation Attributes Vues
Style sheet View
Execution View
Fichier Vues Edition Insert Execution

```
slideshow
  header
    title
      A Video Content Description
    authors
      author
      author
      author
        firstname
          Lionel
        surname
          Villard
        affiliation
    date
  slides
    slide
      title
        Overview
      title
      block
    slide
      title
        Goal and methodology
      block
        itemizedlist
          listitem
            para
              Structuration of video elemen
          itemizedlist
          listitem
          itemizedlist
/slideshow[1]/slides[1]/slide[2]/title[1]
```

```
stylesheet href="file:C:/Users/lvillard/Projets/Multimedia/Schemes/SlideShow/V2/toc.x
  include href="file:C:/Users/lvillard/Projets/Multimedia/Schemes/SlideShow/V2/toc_c
    template match="slideshow" mode="content_toc"
      C-Group
  include href="file:C:/Users/lvillard/Projets/Multimedia/Schemes/SlideShow/V2/toc_a
    template match="slideshow" mode="actors_toc"
      A-Group
        Image ID="A-tocback" FileName="motif.gif" Player="java2D"
        Text ID="A-TOC_title" Player="java2D" FontSize="{$fontSize}" LinePain
          #Text
        Text ID="A-guard" FontSize="{round($fontSize * 0.8)}" Player="java2D" I
        for-each select="slides"
  include href="file:C:/Users/lvillard/Projets/Multimedia/Schemes/SlideShow/V2/toc_t
    template match="slideshow" mode="temporal_toc"
      T-Group Duration="part:60mn"
  include href="file:C:/Users/lvillard/Projets/Multimedia/Schemes/SlideShow/V2/toc_s
    template match="slideshow" mode="spatial_toc"
      param
      param
      S-Group
        Region Actor="A-tocback" Width="{$groupWidth}" Height="{$groupHeigh
        Flow Left="10" Top="10" Depth="21" Direction="vertical"
  template match="slideshow"
    Kaomi
      apply-templates select="." mode="content_toc"
      apply-templates select="." mode="actors_toc"
      apply-templates select="." mode="temporal_toc"
      apply-templates select="." mode="spatial_toc"
```

le of Content
d
verview
oal and methodolo
lideshow Example
onclusion/Perspec

Goal and methodology
- Structuration of video elements
  - Interesting elements
  - Video analysis
  - Structuration
- Specification of multimedia scenario
  - Synchronisation
  - Spatial organization

A Video Content Description 2 /4

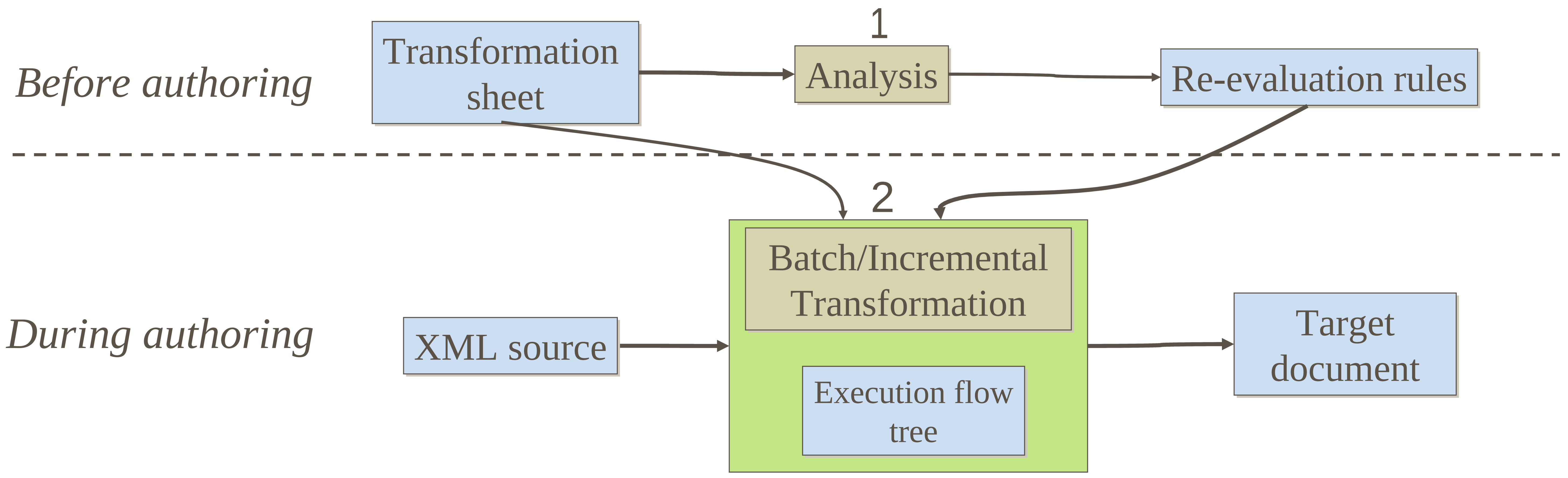
Incremental transformation

To update as fast as possible the document after modifications of:

-XML source document(s)

-Transformation sheet(s)

Incremental transformation: principles



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.

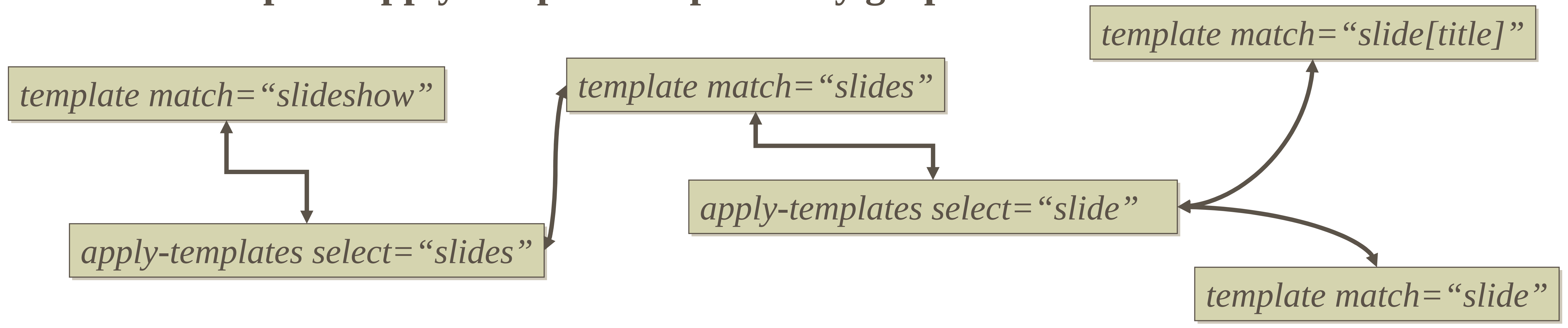
A transformation sheet change

Input parameters, addition/removal of template, addition/removal of instruction

Result of the analysis: set of re-evaluation rules (pattern, XLST instructions)

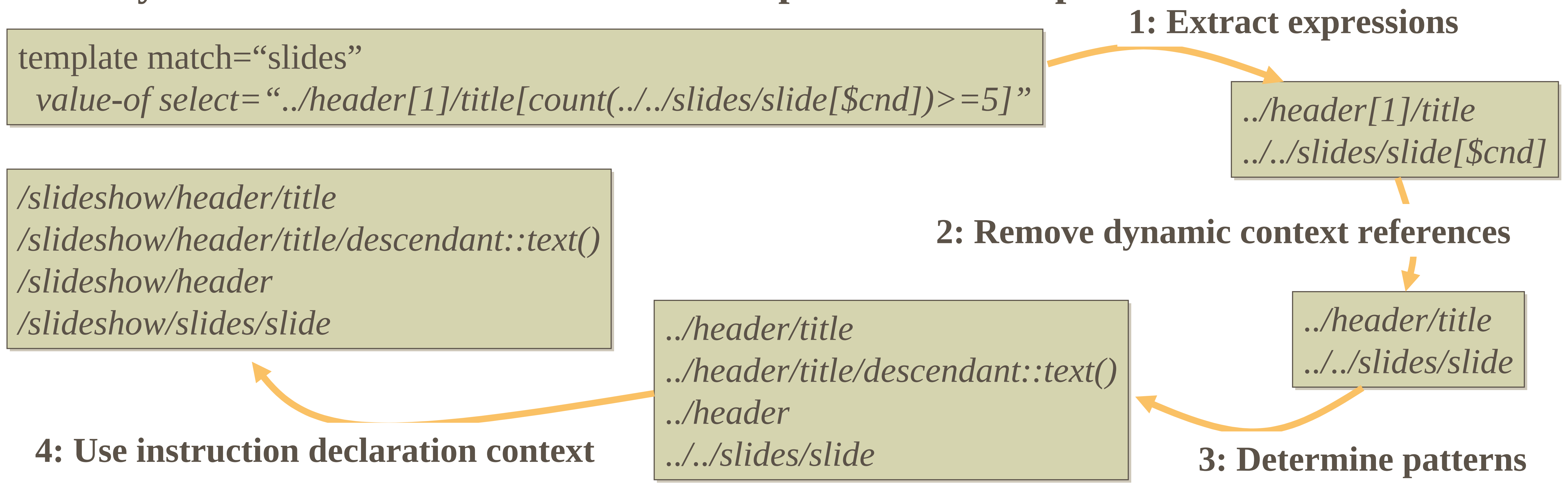
Main steps of analysis

A. Creation of template/apply-templates dependency graph



template match="slideshow"
apply-templates select="slides"
template match="slides"
apply-templates select="slide"
template match="slide[title]"
template match="slide"

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

```
template match="slides"
  value-of select="../header[1]/title[count(../../slides/slide[$cnd])>=5]"
```

```
../header[1]/title
../../slides/slide[$cnd]
```

```
../header/title
../../slides/slide
```

```
../header/title
../header/title/descendant::text()
../header
../../slides/slide
```

```
/slideshow/header/title
/slideshow/header/title/descendant::text()
/slideshow/header
/slideshow/slides/slide
```

2. Incremental transformation: execution step

Standard transformation execution with:

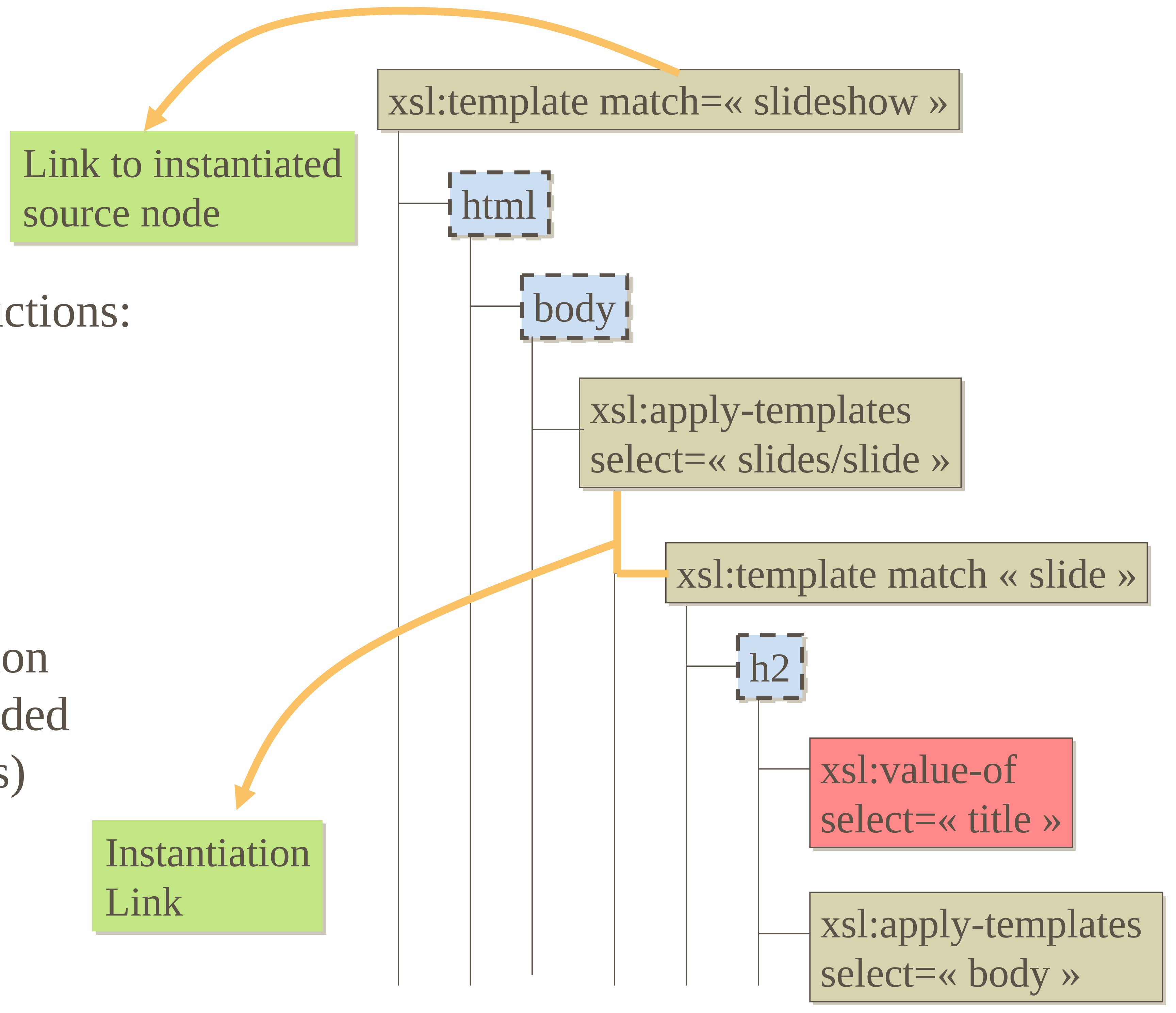
- Only a **subset** of instructions are executed (thanks to re-evaluation rules)
- Compute processor context **as needed**
- Use the **minimal** execution flow tree
- **Incremental** execution of XSLT instructions:

  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


xsl:template match=« slideshow »
Link to instantiated source node
html
body
xsl:apply-templates select=« slides/slide »
xsl:template match « slide »
h2
xsl:value-of select=« title »
Instantiation Link
xsl:apply-templates select=« body »

Experimentation / Evaluation

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

	Batch		Dummy		Change title		Insert section	
Number of instruction to re-execute	N/A		0		795		819	
Time to get instruction to re-execute	N/A		0		80ms		80ms	
Variables value computed	6572		6572		6572		6572	
Variable access count	10279		6899		6899		6983	
Overall timing / ratio	4,5s	1	2,8s	0.6	2,8s	0.62	2,9s	0.64

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)