Transformation XSLT

Claude Moulin

claude.moulin@utc.fr

Exemple - 1

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE uv SYSTEM "intervenants.dtd">
<uv sigle="NF11">
   <titre>Théorie des langages</titre>
   <responsable ref="moulin"/>
   <site>http://www4.utc.fr/~nf11/</site>
   <enseignement>
         … … … … …
   </enseignement>
   <enseignants>
         … … … … …
   </enseignants>
</uv>
```

Exemple 2

```
<?xml version="1.0"
<!DOCTYPE uv SYSTEM
<uv sigle="NF11">
   <titre>Théorie de
   <responsable ref=
   <site>http://www4
   <enseignement>
         … … … … …
   </enseignement>
   <enseignants>
         … … … … …
   </enseignants>
</uv>
```

```
<enseignement>
  <cours>
    <intervenant ref="moulin"/>
  </cours>
  <td>
    <intervenant ref="moulin"/>
    <intervenant ref="boulanger"/>
  </td>
  <tp>
    <intervenant ref="moulin"/>
    <intervenant ref="boulanger"/>
  </tp>
</enseignement>
```

Exemple - 3

```
<?xml versio
<!DOCTYPE uv
<uv sigle="NF
   <titre>Thé
   <responsab
   <site>http
   <enseignem
            … … .
   </enseigne
   <enseignants>
            … … … … …
   </enseignants>
</uv>
```

```
<enseignants>
 <enseignant id="moulin">
  <nom>Claude Moulin</nom>
  <email>claude.moulin@utc.fr</email>
 </enseignant>
 <enseignant id="boulanger">
  <nom>Jean-Louis Boulanger</nom>
  <email>jean-louis.boulanger@utc.fr</email>
 </enseignant>
</enseignants>
```

XSLT

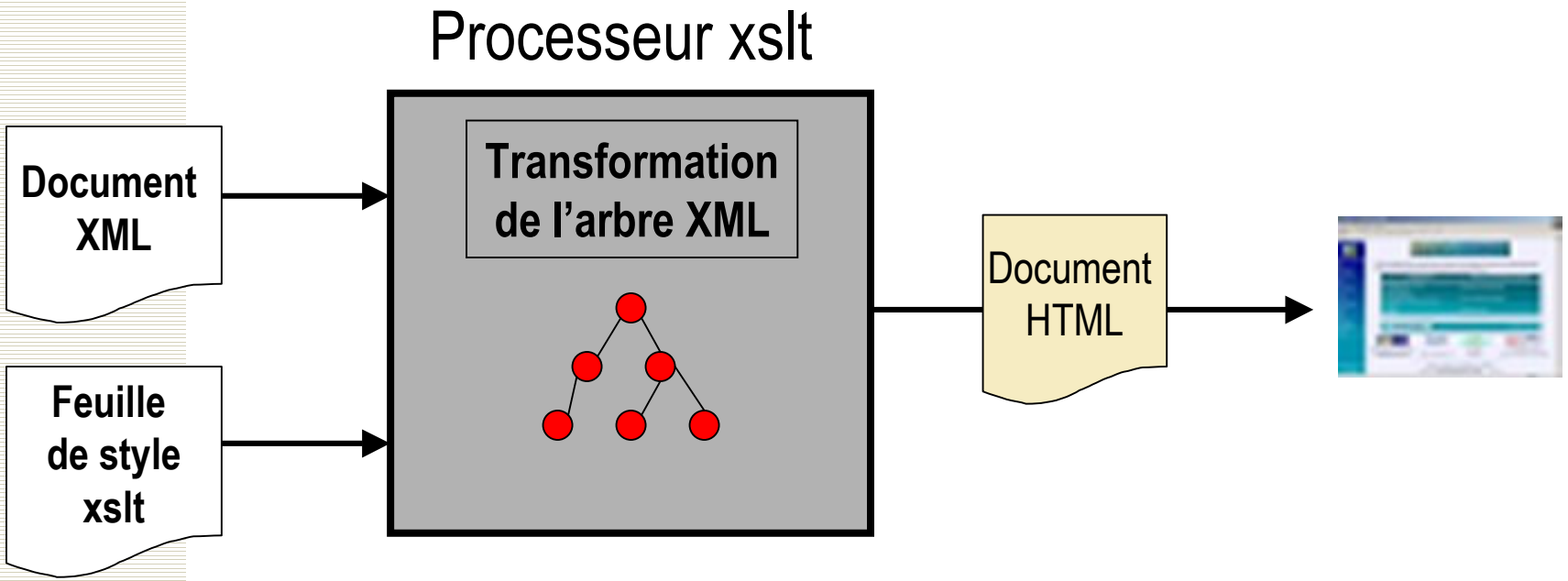
- XSLT (extensible stylesheet language transformation) est un langage de feuilles de style permettant la transformation de documents XML.
- Il permet en particulier de donner une représentation à un contenu XML ; un tel contenu n'a pas d'information sur la façon d'être représenté (il peut y avoir plusieurs représentations).
- Une feuille de style XSLT est un document XML bien formé ayant comme namespace:

  `http://www.w3.org/1999/XSL/Transform`.

Pourquoi utiliser XSLT ?

- Séparer la structure d'un document, de son contenu et de sa présentation, permet de créer une série de présentations, toutes différentes, pour le même document et pour tous les documents similaires.
- Une feuille XSLT est associée en fait à un type de document XML.

Comment l'utiliser ?



Modèle de transformation XSLT

Processeur XSLT

- Il prend en entrée un document XML, une feuille de style XSLT et produit une présentation du contenu XML selon ce qui est indiqué dans l'XSLT.
- L'arbre XML est transformé en un autre arbre (transformation d'arbre).
  - Peut être aussi très différent de l'arbre origine.
  - Le contenu initial peut être filtré et réordonné.
  - De nouvelles informations peuvent être ajoutés.

Mise en oeuvre PHP

- Liens
  - http://www.zend.com/zend/tut/tutorial-wong3.php
  - http://bob.developpez.com/phpxslt/
- Versions
  - PHP 4.3.x, Apache 1.3.x
  - Utiliser easyPHP
- Configuration Windows
  - Copier les macros sablot.dll, expat.dll, iconv.dll dans le répertoire windows/system32
- Charger les extensions XML et XSL du moteur PHP

Exemple de Page PHP - 1

```
<?php
    // page test.php for XML, XSL transformation
    // $xml and $xsl contain the XML and XSL data...
    $xml = implode('',file('xml/document.xml'));
    $xsl = implode('',file('stylesheet/feuille.xsl'));
    $arguments = array(
        '/_xml' => $xml,
        '/_xsl' => $xsl );

    // Allocate a new XSLT processor
    $xh = xslt_create();
    // Process the document, returning the result into the $result variable
    $result = xslt_process($xh, 'arg:/_xml', 'arg:/_xsl', NULL, $arguments);
```

Exemple de Page PHP - 2

```
if ($result) {
    print "SUCCESS, document.xml was transformed by feuille.xsl into the \$result";
    print " variable, the \$result variable has the following contents\n<br>\n";
    print "<pre>\n";
    print $result;
    print "</pre>\n";
}
else {
    print "Sorry, document.xml could not be transformed by feuille.xsl into";
    print "  the \$result variable; the reason is that " .  xslt_error($xh) .
    print " and the error code is " . xslt_errno($xh);
}
xslt_free($xh);
?>
```

Organisation PHP

- Répertoire des pages PHP
  - transform.php
  - Dossier XML
  - Dossier stylesheet

Mise en oeuvre Java

- Installer JAVA 1.4.x
- Télécharger la distribution Xalan et sélectionner les trois librairies java :
  - xalan.jar
  - xercesImpl.jar
  - xml-apis.jar

Organisation Java

- Répertoire du fichier batch (C:/data/xalan-xsl)
  - xalanxsllo18.bat
  - Dossier librairies java (lib)
  - Dossier documents XML, XSL (examples/lo18)

Exemple de fichier batch

```
@echo off
set MBACKUP=%CLASSPATH%
set javapath=C:/j2sdk1.4.2_05/bin
set CLASSPATH=.;C:/data/xalan-xsl/lib/xalan.jar
set CLASSPATH=%CLASSPATH%;C:/data/xalan-xsl/lib/xercesImpl.jar
set CLASSPATH=%CLASSPATH%;C:/data/xalan-xsl/lib/xml-apis.jar
Rem ----- adapter
set XML=exemples/xml/lo18example.xml
set XSL=exemples/xsl/lo18example.xsl
set OUT=result.html
%javapath%/java org.apache.xalan.xslt.Process -IN %XML% -XSL %XSL% -OUT %OUT%
pause
set CLASSPATH=%MBACKUP%
```

Edition

- Utiliser un éditeur XML pour construire les documents XML et XSL
  - XML SPY
  - OXYGEN
- On peut utiliser les moteurs XSL intégrés à ces éditeurs pour réaliser la transformation
  - Pb : le lien entre le document XML et la feuille est inséré dans le document XML

Exemple - 1

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE uv SYSTEM "intervenants.dtd">
<uv sigle="NF11">
   <titre>Théorie des langages</titre>
   <responsable ref="moulin"/>
   <site>http://www4.utc.fr/~nf11/</site>
   <enseignement>
         … … … … …
   </enseignement>
   <enseignants>
         … … … … …
   </enseignants>
</uv>
```

Exemple 2

```
<?xml version="1.0"
<!DOCTYPE uv SYSTEM
<uv sigle="NF11">
   <titre>Théorie de
   <responsable ref=
   <site>http://www4
   <enseignement>
         … … … … …
   </enseignement>
   <enseignants>
         … … … … …
   </enseignants>
</uv>
```

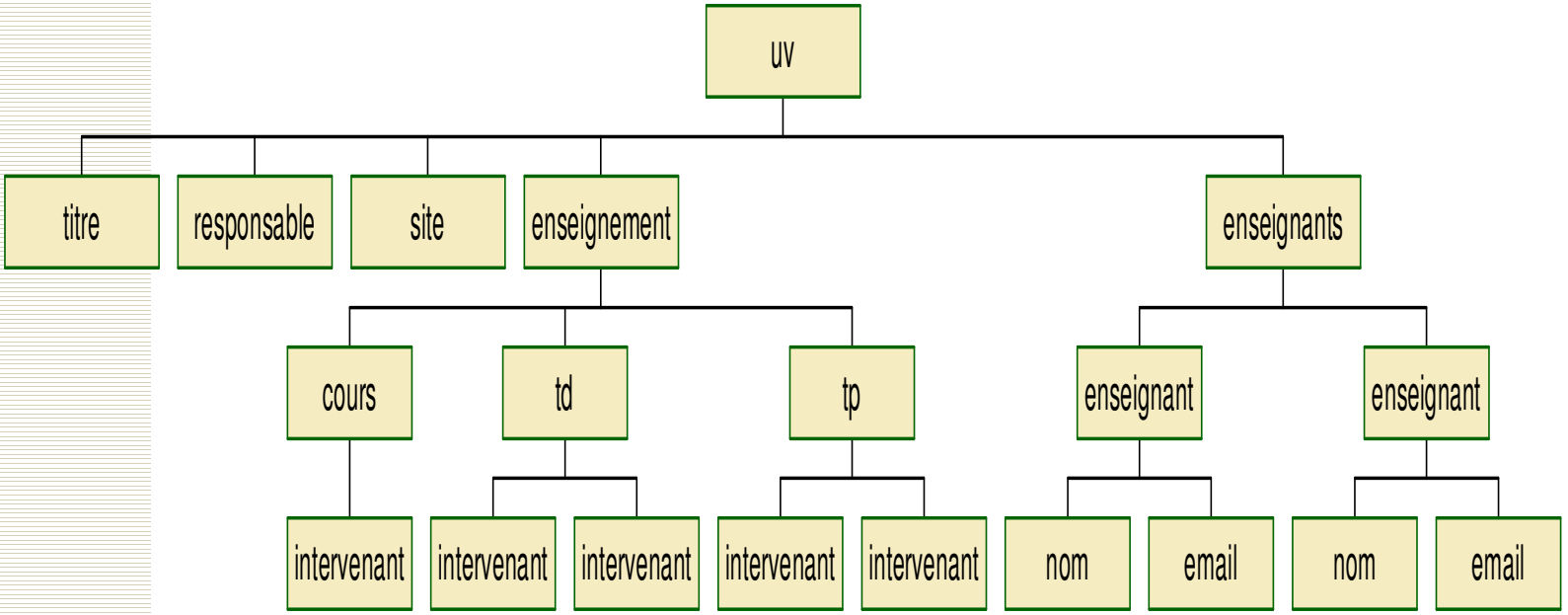
```
<enseignement>
  <cours>
    <intervenant ref="moulin"/>
  </cours>
  <td>
    <intervenant ref="moulin"/>
    <intervenant ref="boulanger"/>
  </td>
  <tp>
    <intervenant ref="moulin"/>
    <intervenant ref="boulanger"/>
  </tp>
</enseignement>
```

Exemple - 3

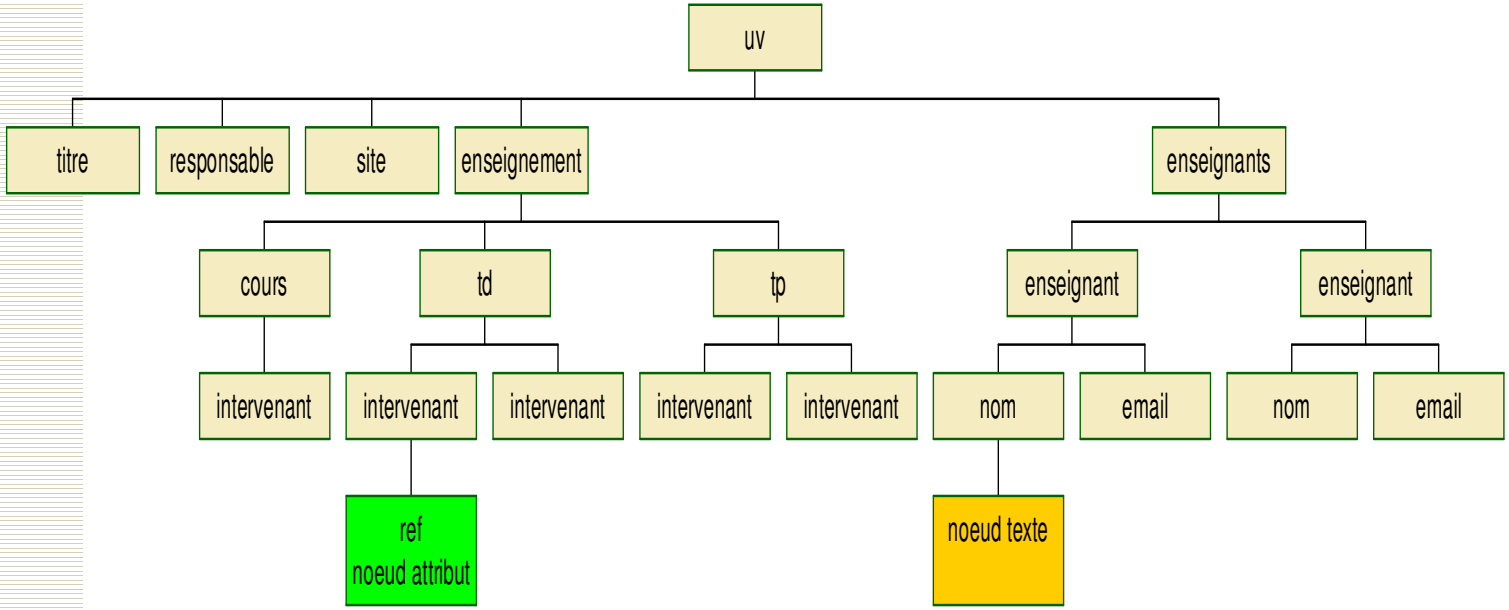
```
<?xml versio
<!DOCTYPE uv
<uv sigle="NF
   <titre>Thé
   <responsab
   <site>http
   <enseigneme
            … … …
   </enseigner
   <enseignant
            … … …
   </enseignants>
</uv>
```

```
<enseignants>
   <enseignant id="moulin">
      <nom>Claude Moulin</nom>
      <email>claude.moulin@utc.fr</email>
   </enseignant>
   <enseignant id="boulanger">
      <nom>Jean-Louis Boulanger</nom>
      <email>jean-
   louis.boulanger@utc.fr</email>
   </enseignant>
</enseignants>
```

Arbre des noeuds élément



Autres noeuds principaux



Sept types de noeuds

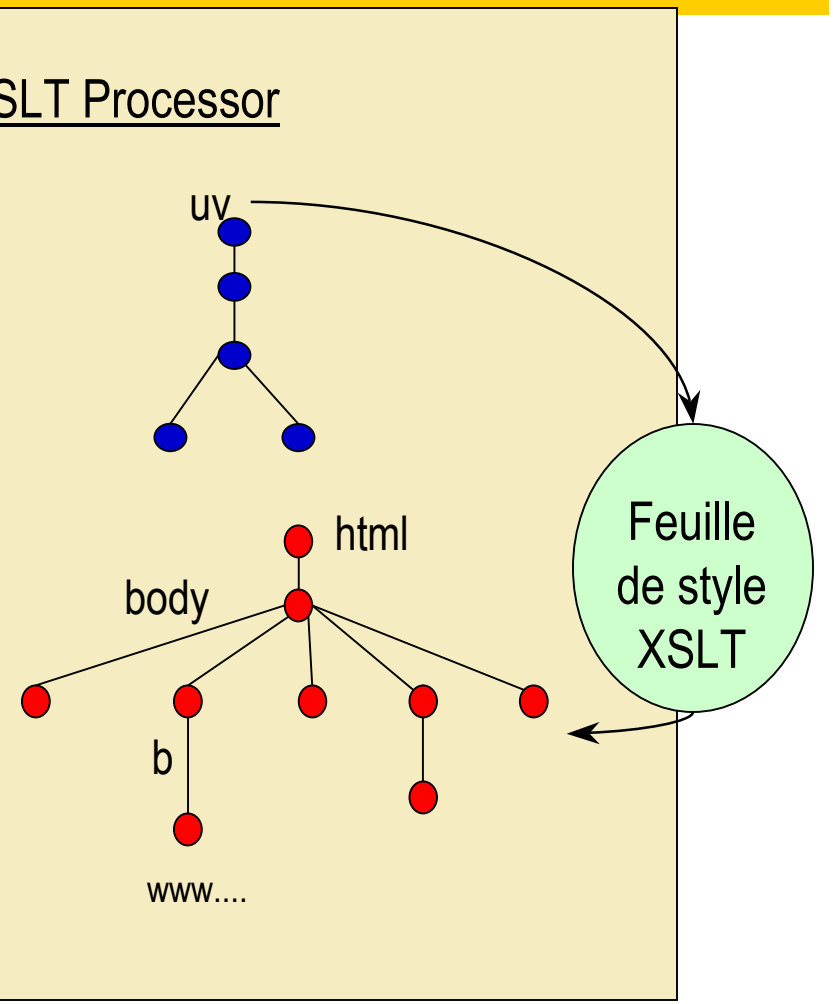
- Nœud racine
- Nœuds éléments
- Nœuds texte
- Nœuds attributs
- Nœud “namespace”
- Nœuds “processing instruction”
- Nœuds commentaires

DOM (Document Object Model)

- Le DOM définit une hiérarchie de types pour représenter un document sous la forme d'un objet composé.
  - Document
  - Node
  - NodeList
  - …

Transformation de l'arbre

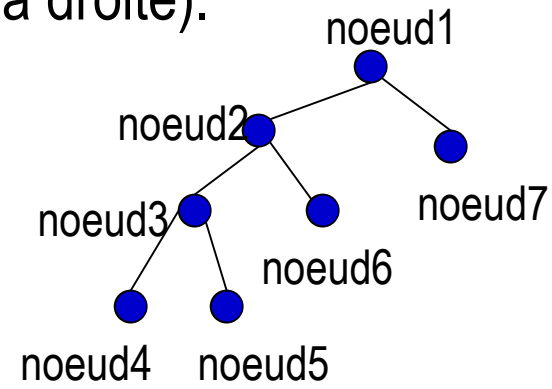
```
<uv sigle="NF11">
 <titre>Théorie des
langages</titre>
 <responsable ref="moulin"/>
<site>http://www4.utc.fr/~nf11/<
/site>
  <enseignement>
   … … … … …
  </enseignement>
  <enseignants>
   … … … … …
  </enseignants>
</uv>
```




```
<html>
 <body>
  Site web:
   <b>www4.....</b>
   .....
</body>
</html>
```

Principe de la transformation

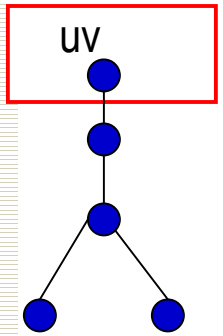
- Les nœuds de l'arbre XML sont analysés en profondeur d'abord (de la racine vers les feuilles et de gauche à droite).



- On applique les règles de transformation sur les nœuds :
  - Pour chaque noeud analysé du document XML, on cherche dans la feuille de style une règle de transformation à appliquer
    - Si une règle est trouvée, elle est appliquée. Le corps de la règle indique sur quel nœud poursuivre l'analyse
    - Sinon passer au noeud suivant

Définition des règles

- Une règle est définie par un modèle (pattern) et un patron (template).
  - Quand le modèle correspond à une position dans l'arbre XML, on applique le template associé, créant ainsi une portion de l'arbre résultant.




```
<xsl:template match="uv">
 <html>
  <body>
        … …
  </body>
 </html>
</xsl:template>
```

xsl:template

- Modèle

```
<xsl:template match="pattern">
   …
</xsl:template>
```

- Définit les règles appliquées durant le processus de la transformation de l'arbre
  - Le modèle est défini par l'attribut « match ».
  - Le template est défini dans le contenu.
- S'insère après le niveau de l'élément racine.

Comment rappeler les templates ?

- Modèle :

```
<xsl:apply-templates select="node-set"/>
```

- Il indique comment continuer l'analyse des nœuds de l'arbre initial, en passant (généralement) aux noeuds descendants, sélectionnés par l'intermédiaire de `select`.
  - Si l'attribut `select` n'est pas présent, le template est appliqué aux noeuds descendants.
- Exemple :

```
<xsl:template match="uv">
  … … …
  <xsl:apply-templates select="*"/>
</xsl:template>
```

Structure de la feuille de style

- Forme :

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
        xmlns:xsl="http://www.w3c.org/1999/XSL/Transform">

        <xsl:template match= "…">
        … …
        </xsl:template>
        …
</xsl:stylesheet>
```

- `xsl:stylesheet`
  - est le nom de l'élément root,
  - contient les autres règles,
  - a comme attributs nécessaires :
    - version : `version = "1.0"`
    - namespace: `xmlns:xsl="http://www.w3c.org/1999/XSL/Transform"`

Générer du texte

```
<xsl:value-of select="node"/>
```

- Extrait le contenu du nœud sélectionné dans l'arbre initial.
- Exemples :

```
<xsl:value-of select= "site"/>
```

  - Extrait le contenu du nœud de l'élément `site`, fils du nœud courant.

```
<xsl:value-of select="@ref"/>
```

  - Extrait la valeur de l'attribut du nœud courant dont le nom est `ref`.