XML

Yannick Prié
UFR Informatique – Université Lyon 1

UE2.2 – Master SIB M1 – 2004-2005

Objectifs des trois cours

- Etre capable de comprendre des documents XML et des DTD
- Etre capable de construire des documents XML et des DTD
- Découverte de quelques DTD « importantes »

Un document XML

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE livre SYSTEM "E:\prie\Enseignement\2004-2005\Master
   SIB\SIB2.2-bloc2-XML\Fichier CM XML\exemple-intro.dtd">
<livre id="561" nbpages="190" titre="La compagnie des spectres">
   <auteur>
        <nom>Salvayre</nom>
        <prenom>Lydie</prenom>
   </auteur>
   <format type="poche">
        <mesure type="largeur" unite="cm">11</mesure>
        <mesure type="longeur" unite="cm">19</mesure>
        <mesure type="hauteur" unite="mm">10</mesure>
   </format>
</livre>
```


La DTD correspondante

```
<!ELEMENT livre (auteur, format)>
<!ATTLIST livre
   id CDATA #REQUIRED
   nbpages CDATA #REQUIRED
   titre CDATA #REQUIRED >
<!ELEMENT auteur (nom, prenom)>
<!ELEMENT format (mesure+)>
<!ATTLIST format
   type CDATA #REQUIRED >
<!ELEMENT mesure (#PCDATA)>
<!ATTLIST mesure
   type (hauteur | largeur | longeur) #REQUIRED
   unite (cm | mm | in) #REQUIRED >
<!ELEMENT nom (#PCDATA)>
<!ELEMENT prenom (#PCDATA)>
```


Plan

- Documents XML
  - Syntaxe XML et documents bien formés
- Types de documents XML
  - DTD et documents valides
  - Introduction à XML-Schema
- Le monde XML
  - Quelques normes liés à XML
  - Quelques DTD importantes

Plan

- **Documents XML**
  - Syntaxe XML et documents bien formés
- Types de documents XML
  - DTD et documents valides
  - Introduction à XML-Schema
- Le monde XML
  - Quelques normes liés à XML
  - Quelques DTD importantes

Qu'y a-t-il dans un document XML ?

- Prologue
  - Déclaration XML
  - Déclarations de DTD
    - Instructions pour les processeurs XML
  - Instructions de traitement
    - Instructions pour applications externes
- Arbre des éléments
  - Eléments
    - Balises XML pour le marquage
    - Contenu
      - texte
      - autres éléments
  - Attributs des éléments
    - Information associées aux éléments
- Commentaires

Déclaration XML

- Syntaxe générale

<?xml version="1.0" [encoding = "encodage"] [standalone="yes|no »] ?>

- Est une des informations de traitement
- Indique
  - Conformité du document à une version de la norme XML
    - version="1.0"
  - Jeu de caractères utilisé dans le document
    - encoding = "UTF-8"
  - Présence ou non de références externes
    - standalone="yes"

Instructions de traitement

- Informations nécessaires à une application externe
- Format :
  - **<?NomApplication paramètres ?>**
- Exemples
  - Déclaration de feuille de style à utiliser
    - **<?xml-stylesheet href="fichier.xsl" type="text/xsl"?>**
  - Déclaration XML de début de fichier
    - **<?xml version='1.0' ?>**

Eléments : règles de base

- Un nom d'élément
  - commence par une lettre ou souligné
  - contient des lettres, chiffres, et "-", ".", ":", "_"
  - peut posséder un nom de domaine
    - `domaine:nom_element`
    - Ex. : `xsl:template`
- Les noms d'éléments dépendent de la casse
  - `<nom_element>` ≠ `<nom_Element>`
- Balises
  - de début : `<nom_element>`
  - de fin : `</nom_element>`
- Les éléments peuvent être vides
  - pas de contenu
  - `<element_vide />`
  - Ex :
    - `<img src= "toto.jpg" />`

Arbre des éléments

- Un seul élément racine qui contient tous les autres
- Pas d'intersections entre éléments
  - Mauvais : `<nom1><nom2>…</nom1></nom2>`
  - Bon : `<nom1><nom2>…</nom2></nom1>`
- Blancs ou retours chariot en général non significatifs
  - `<section><p> … </p></section>`
  - ```
    <section>
       … 
    </section>
    ```
- Les éléments sont ordonnés

Caractères spéciaux

- Ces caractères ont une signification spéciale pour les outils XML
- Il faut les écrire différemment
  - < `<`
  - > `>`
  - & `&`
  - ' `'`
  - " `"`
    ```

Attributs associés aux élément : règles de base

- Dans les balises ouvrantes
  - `<el att1="valeur1" att2="valeur2">`
- Les noms d'attributs dépendent de la casse
  - `<el att1="valeur1" Att1="valeur2">`
- Valeurs d'attributs entourées
  - par des guillemets (") ou des apostrophes (')
- Les attributs sont non-ordonnés

Attributs

- Les valeurs peuvent être
  - des données textuelles
    - `value="N'importe quoi"`
  - des *tokens* (noms XML) simples
    - `value = "blue"`
  - des ensemble *de tokens*
    - `value = "red green blue"`
- Possibilité d'énumérer les valeurs possibles et de mettre des valeurs par défaut (voir DTD)

Attributs de type ID et IDREF(S)

- Permettent des relations non hiérarchiques entre éléments
  - ID : identificateur unique dans le document XML
  - IDREF : référence à un élément ayant un attribut de type ID
  - IDREFS : références à *des* éléments ayant un attribut de type ID
- Exemple

```
<société codes_services="A001 A003">
<service code="A001">
<employé code="E205" code_service="A001"> Jean Dupont </employé>
<employé code="E206" code_service="A001"> Frédéric Marc </employé>
<employé code="E207" code_service="A001"> Fabrice Detterne
   </employé>
<employé code="H107" code_service="A003"> Angélique Millet
   </employé>
</service>
<service code="A003">
<employé code="A115" code_service="A003"> Isabelle Mascot
   </employé>
</service>
</société>
```


Commentaires

- Les commentaires ne sont pas considérés comme faisant partie du document XML.
  - **`<!-- Un commentaire -->`**
- Pas de '--' dans un commentaire !
- Un commentaire ne peut pas se trouver dans une autre déclaration

Au bilan : dans un document XML

- Prologue
  - en-tête
  - déclaration de DTD *(pas encore vu)*
  - instructions de traitement
- Eléments
  - attributs
  - contenus
- Commentaires

Plan

- Documents XML
  - Syntaxe XML et documents bien formés
- **Types de documents XML**
  - DTD et documents valides
  - Introduction à XML-Schema
- Le monde XML
  - Quelques normes liés à XML
  - Quelques DTD importantes

Traiter automatiquement un document XML

- Parser
  - Outil qui lit un document XML et construit l'arbre des éléments en mémoire
- Vérifier qu'un document répond bien à la syntaxe XML
  - Document *bien formé*
  - Possibilité de l'utiliser en tant que tel
    - ex. : le présenter à l'utilisateur
- Vérifier en plus qu'un document suit bien la grammaire définie dans une DTD
  - Document *valide*

Document Type Definition

- Définir le type de document XML voulu
  - décrire comment construire un document XML qui lui corresponde (grammaire)
- Permet de
  - valider un document XML (parser validant)
    - vérifier que tous les éléments sont présents et corrects
    - vérifier que les noms d'attributs et leurs valeurs sont corrects
  - transmettre cette connaissance à d'autres
    - ils pourront définir leurs propres documents XML dans le même cadre
    - d'où possibilité de standardisation et d'échanges

DTD

- Une déclaration
  - contenant la définition formelle de la structure autorisée,
  - qui décrit donc
    - quels noms sont utilisés pour les types d'éléments
    - comment ces types d'éléments s'organisent
      - ordre
      - hiérarchie
    - les attributs des éléments
    - des entités analysables ou non
    - des notations pour les types de données binaires
- Liaison DTD / document XML
  - Soit la DTD est dans le document XML (inline)
  - Soit le document XML réfère à la DTD avec une URI (la DTD est dans un fichier externe)

Déclarations

- Instructions pour le processeur XML
- Format : `<! … >` ou `<! … [<! … >]>`
  - Document type - `<!DOCTYPE … >`
  - Character data - `<![CDATA[ … ]]>`
  - Entities - `<!ENTITY … >`
  - Notation - `<!NOTATION … >`
  - Element - `<!ELEMENT … >`
  - Attributes - `<!ATTLIST … >`
  - `<![INCLUDE[…]]>` et `<![IGNORE[…]]>`

Déclaration *Document Type*

- Identifie le nom de l'élément racine du document
  - **`<!DOCTYPE My_XML_Doc>`**
- Permet aussi de rajouter des définitions d'entités et des DTD
  - **`<!DOCTYPE My_XML_Doc [ … ] >`**

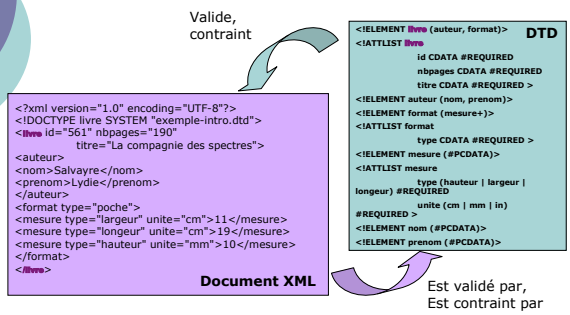
```
<My_XML_Doc>
  ...
</My_XML_Doc>
```

déclaration de la DTD

Déclaration *Character Data*

- Dans les occasions pour lesquelles le texte doit contenir des caractères qui ne doivent pas être interprétés
- Deux textes équivalents
  - **`Press <<<ENTER>>>`**
  - **`<![CDATA[Press <<<ENTER>>>]]>`**

DTD et document XML



Mettre en place une DTD

- Utiliser les composants de XML…
  - Entités, éléments, déclarations, instructions de traitements, listes d'attributs, *etc*.
- … dans des DTD pour spécifier les règles permettant de valider des documents XML
  - Définir un modèle (type) de document de façon formelle
- Une DTD décrit
  - Quels noms peuvent être utilisés pour les types d'éléments
  - L'ordre dans lesquels ceux-ci peuvent apparaître
  - La hiérarchie documentaire
  - Les noms et les types des attributs d'éléments

Déclaration de DTD

- La DTD est stockée
  - soit dans le fichier XML
  - soit dans un fichier extérieur
  - soit dans les deux
- Une DTD interne peut écraser ou ajouter des ENTITY ou des ATTLIST à des définitions de DTD externes
- Une DTD est composée de déclarations
  - ELEMENT – définitions d'éléments
  - ATTLIST – définitions d'attributs
  - ENTITY – définitions d'entités
  - NOTATION – définitions de notations

Déclarations d'éléments

- Définir un élément et son contenu
  - `<!ELEMENT name (#PCDATA)>`
    `⇒ <name> ...du texte... </name>`
- Un élément vide n'a pas de contenu
  - `<!ELEMENT name EMPTY>`
    `⇒ <name/>`
- Si on autorise les fils
  - Quelconques :
    `<!ELEMENT name ANY>`
  - Spécifiés :
    `<!ELEMENT person (name, e-mail*)>`

Spécification des fils (grammaire)

- Définir le contenu des éléments
  `<!ELEMENT person (name, e-mail*)>`
- …et définir une hiérarchie d'éléments
  `<!ELEMENT name (fname, surname)>`
  `<!ELEMENT fname (#PCDATA)>`
  `<!ELEMENT surname (#PCDATA)>`
  `<!ELEMENT e-mail (#PCDATA)>`

- Organisation des sous-éléments
  - Connecteur de séquence ',' : `(A, B, C)` [puis]
  - Connecteur de choix '|' : `(A | B | C)` [ou]

Indicateurs de quantité

- Contraintes sur les éléments des DTD

`A?`	Possible	[0..1]
`A+`	1 fois et plus	[1..*]
`A*`	0 ou plus	[0..*]

- Exemples
  `(A, B)+`
  `((A,B?) | C+)*`

Déclaration d'attributs

- Les attributs sont associés aux types d'éléments
- Déclarés dans une déclaration ATTLIST
  - **<!ELEMENT element … >**
  - **<!ATTLIST element … >**
  - Il faut ensuite définir
    - le nom de l'attribut
    - le type de l'attribut
    - sa valeur par défaut

Noms et types d'attributs

- Noms d'attributs
  - **<!ATTLIST elem** name type default**>**
  - **<!ATTLIST elem** first_attr …
    secon_attr …
    third_attr … **>**
- Types d'attributs

CDATA	ID
NMTOKEN	IDREF
NMTOKENS	IDREFS
ENTITY	NOTATION
ENTITIES	name group

Types d'attributs (1)

- CDATA
  - Chaîne de caractères
  - `<!ATTLIST person name CDATA … >`
  - `<person name = "Tom Jones">`
- NMTOKEN
  - Token unique
  - `<!ATTLIST mug color NMTOKEN … >`
  - `<mug color="red">`
- NMTOKENS
  - Multiples tokens
  - `<!ATTLIST temp values NMTOKENS … >`
  - `<temp values="12 15 34">`

Joue sur la manière dont le parser interprète l'attribut

Types d'attributs (2)

- ENTITY
  - L'attribut est une référence d'entité
  - `<!ATTLIST person photo ENTITY … >`
  - `<person photo="MyPic">`
- ENTITIES
  - Plusieurs références d'entités
  - `<!ATTLIST album photos ENTITIES …>`
  - `<album photos="pic1 pic2">`
- ID
  - Identificateur unique
  - `<!ATTLIST person id ID … >`
  - `<person id = "P09567">`
- IDREF
  - Référence à un ID
  - `<!ATTLIST person father IDREF … >`
  - `<person father="P09567">`

Types d'attributs (3)

- IDREFS
  - Référence à plusieurs ID
  - `<!ATTLIST person children IDREFS … >`
  - `<person children="P09567 P09677">`
- Name group
  - Liste restreinte de valeurs possibles
  - `<!ATTLIST point coord (X|Y|Z) … >`
  - `<point coord="X">`
- NOTATION
  - Décrit des données non XML
  - `<!NOTATION jpg SYSTEM "JPEG Image" >`
  - `<!NOTATION gif PUBLIC "+//ISBN 0-7923-9432-1::Graphic Notation//NOTATION CompuServer Graphic Interchange Format//EN">`
  - `<!ATTLIST image format NOTATION (jpg|gif) …>`
  - `<image format="gif">`

Valeurs d'attributs par défaut

- Quatre types
  - **#REQUIRED** doit être spécifié
  - **#IMPLIED** peut être spécifié
  - **"*default*"** valeur par défaut si non spécifié
  - **#FIXED** une seule valeur autorisée

```
<ATTLIST  tag      name      type       default>
<!ATTLIST citoyen  parents   IDREFS     #REQUIRED
                   id        ID         #IMPLIED
                   sex       (m|f)      "f"
                   adress    CDATA      #IMPLIED
                   nat       CDATA      #FIXED "Fr">
```

Définition interne de DTD

- Dans la déclaration DOCTYPE

```
<?xml version="1.0" standalone="yes" ?>
<!DOCTYPE racine [
  <!-- ici la DTD -->
  <! … >
  <! … >
]>
<!-- ici le reste du fichier XML -->
<racine>
...
</racine>
```


Définition externe privée de DTD

- Référence à la DTD externe par un chemin dans la déclaration DOCTYPE
  - DTD externe privée

```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE racine
  SYSTEM "dossiers/MyDoc.dtd" [
  <!-- déclarations supplémentaires -->
  <! … >
  <! … >
]>
<!-- ici le reste du fichier XML -->
```

  - Les déclarations spécifiques au document restent définies de façon interne

Définition externe privée de DTD

- Référence à la DTD externe par une URL dans la déclaration DOCTYPE
  - DTD externe privée

```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE racine
  SYSTEM "http://…/MyDoc.dtd" [
  <!-- déclarations supplémentaires -->
  <! … >
  <! … >
]>
<!-- ici le reste du fichier XML -->
```

  - Les déclarations spécifiques au document restent définies de façon interne

Définition externe publique de DTD

- Utilisation du mot-clé PUBLIC

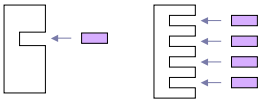
```
<!DOCTYPE racine
  PUBLIC "identifiant public" "url" ?>
```

- Exemple

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
  ......
</html>
```


Les entités

- Sont des alias associant un nom à des « unités d'information »
- Les entités spécifiques au document sont décrites dans sa DTD interne
- Les entités plus générales sont décrites dans des DTD externes
- Chaque entité
  - est identifiée par un nom
  - est définie par une déclaration d'entité
  - est utilisée en appelant une référence d'entité



Utilisation des entités

- Quand l'information
  - Est utilisée dans plusieurs endroits
    - Ex. déclaration légale, caractère spécial
  - Est une partie d'un document qui doit être tronçonné pour rester gérable
    - Ex. livre : 1 fichier + n chapitres : n fichiers
  - Est conforme à un format de donnée différent de XML
    - Ex. : image JPEG

Types d'entités

- Entités internes
  - générales
    - utilisées dans les documents XML
  - paramètre
    - utilisées dans les déclarations dans les DTD
- Entités externes
  - générales
  - paramètres
- Entités analysables
- Entités non analysables
- Entités caractères
  - déjà vues

Entités générales internes

```
<!ENTITY nom "chaîne de remplacement">
```

- Entités analysables utilisées uniquement dans le document
- Référence : `&nom_entité;`
- Exemple
  - Déclaration dans la DTD

```
<!ENTITY PCI "Permis de conduire informatique"
  >
```

  - Utilisation

```
<p>Le cours du PCI (&PCI;) se compose de...</p>
```


Entités générales externes

```
<!ENTITY nom SYSTEM "URI"  >
```

- Permet de constuire un document XML à partir de plusieurs autres documents
- Référence : `&nom_entité;`
- Exemple
  - Déclaration dans la DTD

```
<!ENTITY doc SYSTEM "http://toto.org/doc.xml"  >
```

  - Utilisation

```
<aide> &doc; </aide>
```


Entités paramètres internes

```
<!ENTITY % nom "caractères de remplacement"  >
```

- Entités analysables uniquement utilisées dans les DTD
- Référence dans la DTD : `(%nom_entité;)` (parenthèses conseillées)
- Exemples
  - Déclarations DTD

```
<!ENTITY % tout "ANY"  >
<!ENTITY % common "(para|list|table)">
```

  - Utilisations dans la DTD

```
<!ELEMENT paragraphe %tout; >
<!ELEMENT chapter ((%common;)*, section*)>
<!ELEMENT section (%common;)*>
```


Entités paramètres externes

```
<!ENTITY % nom SYSTEM "URI"  >
```

- Pour construire une DTD complexe à partir d'autres DTD complémentaires
- Référence dans la DTD : `%nom_entité;`
- Exemple
  - Déclaration dans la DTD

```
<!ENTITY % règles SYSTEM "http://toto.org/regles.dtd" >
```

  - Utilisation dans la DTD

```
%règles;
```


Entités analysables

- Le texte de remplacement fait partie intégrale du document
  - les données sont analysées correctement par le parser XML
- Déclaration dans la DTD comme ENTITY
- Utilisation avec `&nom;` ou `%nom;`

Entités non analysables

```
<!ENTITY % nom SYSTEM "URI" NDATA notation >
```

- Pour déclarer un contenu non XML dans un document XML
  - fichier image, audio, *etc.*
- Référence : `&nom_entité;` uniquement comme attribut de type ENTITY
- Exemple
  - Déclaration DTD

```
<!NOTATION TIFF SYSTEM "format TIFF">
<!ENTITY photo SYSTEM "photo.tif" NDATA TIFF>
<!ELEMENT pic EMPTY)>
<!ATTLIST pic name ENTITY #REQUIRED>
```

  - Utilisation dans le document XML

```
<pic name="photo" />
```


Déclarations de notations

```
<!NOTATION nom SYSTEM "URI" >
```

- Pour
  - Identifier par un nom le format des entités non XML externes
  - Définir les formats des données et les applications qui permettent de les traiter
- Exemple

```
<!NOTATION GIF SYSTEM "GIF" >
<!NOTATION GIF89a PUBLIC "-//Compuserve//NOTATION
  Graphics Interchange format 89a//EN" >
```


Identificateurs publics

```
PUBLIC "-//EBI//DTD My book//EN" "url"
```

- PUBLIC keyword
- Identifier type - Registered + / - / ISO
- Owner identifier
- Public text class
  - DTD, NOTATIONS, ENTITIES, TEXT
- Public text description
- Public text language
- url non obligatoire mais conseillée

Exemples d'utilisation

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<!NOTATION GIF89a PUBLIC "-//Compuserve//NOTATION
Graphics Interchange format 89a//EN" >
```


Construire une DTD

- Non trivial : il faut éviter de se tromper
  - Changer une DTD XML a des conséquences sur les documents qui la suivent
- Ressemble à la création d'un schéma de base de données
- Il faut considérer
  - Le problème de la granularité
  - La questions des attributs et des éléments
  - Les limitations inhérentes aux DTD

Identifier les données qui nécessitent d'être balisées

- Pour chaque unité d'information, déterminer
  - Peut-on lui donner un nom ?
  - Apparaît-elle tout le temps ?
  - Peut-il y en avoir plusieurs ?
  - Peut-on la décomposer en des unités plus petites ?
  - Y-a-t'il du contenu textuel qui ne change pas ?
  - Comment est-elle associée aux autres unités ?

Granularité

-
```
<PERSON>
    <NAME>Jon Smith</NAME>
</PERSON>
```

-
```
<PERSON>
    <FORENAME>Jon</FORENAME>
    <SURNAME>Smith</SURNAME>
</PERSON>
```

Eléments ou attributs ?

- Comment les données doivent-elles être encapsulées ?
  - <book>
    <title>The Forty-nine Steps</title>
    …
    </book>
  - <book title="The Forty-nine Steps">
    …
    </book>
- Tout dépend de ce que l'on veut faire…
- Il existe des avis tranchés…

Eléments ou attributs ? (2)

- Séparer le contenu des métadonnées
  - Données qui doivent être imprimées comme du texte comme contenu
  - Métadonnées comme attributs
- Règles générales
  - Si on enlève toutes les balises, le document doit encore être lisible et utilisable
  - S'il y a doute, utiliser un attribut

Limites des DTD / XML

- XML est seulement une syntaxe
- XML ne porte pas de sémantique
- Uniquement description de structure
- Pas de types

- Un des moyens de pallier certains problèmes
  - XML-schema

DTD

- Une syntaxe de description non-XML, héritée de SGML
  - Oblige à apprendre un langage supplémentaire
  - Ne permet pas de manipuler les DTD avec des outils XML
- Pas assez de contraintes sur les données manipulées
  - Toute données est une chaîne de caractères
  - Impossible de
    - spécifier des types simples
      - ex. entiers, dates, etc.
    - spécifier des cardinalités simples
      - ex. « un ARTICLE aura entre 1 et 4 MOTS-CLE »
    - spécifier des contraintes simples
      - ex. entier positif

XML-Schema

- Autre manière de spécifier des types de documents XML
- Le schéma est exprimé en XML
- Possibilité de spécifier plus de contraintes sur les données
- Possibilités avancées d'extension des schémas
- On élargit l'approche de gestion documentaire à celle plus générale de gestion de données

Exercice 1 : Proposez plusieurs documents XML valides suivants la DTD suivante

```
<!ENTITY % opt_fields "year?, volume?, pages?,
month?, url?, abstract?, note?">
<!ENTITY % req_fields "author, title">
<!ENTITY % key_atts "key ID #REQUIRED">

<!ELEMENT bibtex-file
(article|inproceedings|book|techreport|phdthesis|
unpublished|misc)*>

<!ELEMENT key        (#PCDATA)>
<!ELEMENT author     (name+)>
<!ELEMENT name       (#PCDATA)>
<!ELEMENT title      (#PCDATA)>
<!ELEMENT booktitle  (short, long?)>
<!ELEMENT short      (#PCDATA)>
<!ELEMENT long       (#PCDATA)>
```

```
(suite)
<!ELEMENT year       (#PCDATA)>
<!ELEMENT volume     (#PCDATA)>
<!ELEMENT pages      EMPTY>
<!ATTLIST pages
        first NMTOKEN #REQUIRED
        last  NMTOKEN #REQUIRED>
<!ELEMENT month      (#PCDATA)>
<!ATTLIST month
        mtype (short | long) "short">
<!ELEMENT url        EMPTY>
<!ATTLIST url
        ftype (ps|pdf|html) #REQUIRED
        href  CDATA         #REQUIRED>
```

```
(suite)
<!ELEMENT abstract     (#PCDATA)>
<!ELEMENT number       (#PCDATA)>
<!ELEMENT note         (#PCDATA)>
<!ELEMENT publisher    (#PCDATA)>
<!ELEMENT institution  (#PCDATA)>
<!ELEMENT school       (#PCDATA)>
<!ELEMENT howpublished (#PCDATA)>
<!ELEMENT address      (#PCDATA)>
<!ELEMENT inproceedings (%req_fields;, booktitle, %opt_fields;)>
<!ATTLIST inproceedings %key_atts;>
<!ELEMENT article      (%req_fields;, journal, %opt_fields;)>
<!ATTLIST article      %key_atts;>
<!ELEMENT book         (%req_fields;, publisher, %opt_fields;)>
<!ATTLIST book         %key_atts;>
<!ELEMENT techreport   (%req_fields;, institution?, number?, %opt_fields;)>
<!ATTLIST techreport   %key_atts;>
<!ELEMENT phdthesis    (%req_fields;, school?, %opt_fields;)>
<!ATTLIST phdthesis    %key_atts;>
<!ELEMENT unpublished  (%req_fields;, %opt_fields;)>
<!ATTLIST unpublished  %key_atts;>
<!ELEMENT misc         (%req_fields;, howpublished, %opt_fields;)>
<!ATTLIST misc         %key_atts;>
```


Exercice 2

- Proposez deux DTD permettant de valider le document XML suivant

```
<examen code="coursXML">
<titre>Outils et documents XML</titre>
<date mois="jan" annee="2006"/>
<questions>
<question> <partie>Première partie<partie>Une sous-partie</partie>
/partie></question>
<question> <partie>Deuxième partie></partie> </question>
<question> <partie/> </question>
</questions>
</examen>
```


Exercice 3

- On veut représenter en XML des données concernant les étudiants en SIB et leurs enseignements
- Proposez une DTD

Plan

- Documents XML
  - Syntaxe XML et documents bien formés
- Types de documents XML
  - DTD et documents valides
  - Introduction à XML-Schema
- **Le monde XML**
  - Quelques normes liés à XML
  - Quelques DTD importantes

Standardisation

- XML permet de définir des DTD
  - modèles de documents
  - modèles de représentation de données
- Dès qu'on a un groupe, partage de données/documents
  - nécessité de partager les manières de décrire
  - accord
    - local
    - global → standardisation
- Des standards sous la forme de DTD (ou de schémas),
  - Stricts
  - Qui peuvent être raffinés
    - Les spécialiser avec des DTD internes
    - N'en utiliser que des parties

Avantages et applications XML

- Avantages
  - Réutilisabilité, partage
  - Pérennité
  - Intégrité
  - Portabilité
- Applications
  - Documents
  - Echange de données
  - Bureautique
  - Web
  - BDD semi-structurées
  - Commerce électronique
  - ...

Quelques standard XML

- The XML Bookmark Exchange Language (XBEL)
- Open eBook Publication Structure
- SportsML
- NewsML
- XML Book Industry Transaction Standards (XBITS)
- OpenDocument…
- DocBook
- ebXML (electronic Business)
- Universal Description, Discovery & Integration (UDDI)
- Text Encoding and Interchange (TEI)
- XTM (XML Topic Maps)
- ...

http://publishing.xml.org/standards/
http://www.oasis-open.org/specs/index.php

Quelques spécifications XML (W3C)

- XML Schema
- XLink et XPointer
- XPath
- XSL et XSLT
- XML Query
- Namespaces
- SAX
- DOM
- MathML
- OWL
- RDF
- SMIL
- SOAP
- SVG
- XHTML

Voir http://www.w3c.org/

XPath

- Standard permettant d'identifier et de spécifier toutes données dans un document XML
- Exemples

```
//toto[@name]
```

→ Tous les élément toto qui ont un attribut name

```
//tata/descendant::*
```

→ Tous les descendants des éléments tata

- Voir cours Xpath

XLink

- Objectif
  - Donner la possibilité de liens riches
- XLink
  - XML Linking Specification
  - Liens
    - simples (1:1) et étendus (n:n),
    - typés
    - internes ou externes
- Exemple

```
<site xlink:type="local"
      xlink:href="http://www.xml.fr/FAQ.xml"
      xlink:label="fr"
      xlink:title="Version française"/>
```


XPointer

- Objectif
  - Pointer précisément dans un document XML
- XPointer
  - XML Extended Pointer Specification
  - Une référence absolue (le document XML)
  - et une référence relative (à l'intérieur du document)
    - expression XPATH
- Exemples

http://www.toto.org/xml/doc.xml#xptr(/intro/title)
élément title dans élément intro de doc.xml
http://site.fr/page.xml|id('ref12").child(1,session)
premier élément session enfant de l'élément identifié par ref12, dans le document page.xml

XSL

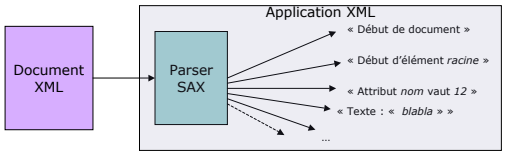
- Ensemble d'outils permettant de
  - Transformer les documents XML en d'autres documents XML (changement de format)
  - Visualiser les documents XML sous forme lisible, pour de multiples supports
- eXtensible Style Language
  - XSL-FO (« XSL Formatting Objects)
    - présenter des informations
  - XSLT (XSL transformation)
    - transformer un arbre XML en un autre arbre XML
  - Voir cours XSL-XSLT

Xquery

- Standard XML permettant d'exprimer des requêtes dans les documents XML
- Syntaxe XML ou non
- Utilisation de Xpath

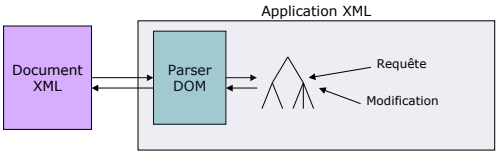
XML et les applications : SAX

- SAX : Simple API for XML
  - Principe : un parser SAX lit un document XML, et envoi un message à une application dès qu'il
    - rencontre un début ou une fin de document
    - rencontre un début ou une fin d'élément
    - trouve des caractères dans un élément, *etc.*
  - Le programmeur qui écrit l'application XML décide quoi faire de chaque message



XML et les applications : DOM

- DOM : Document Object Model
  - Principe : un parser lit un document XML et fabrique un arbre des éléments en mémoire.
  - Le programmeur qui écrit l'application XML peut alors accéder aux informations de cet arbre, les modifier, enregistrer celui-ci, etc.



SVG

- Objectif
  - Description de schémas
  - Eléments
    - Formes (lignes, courbes, triangles, rectangles, *etc.*), images, textes, groupes d'éléments, ...
  - Affichages
    - opacités, redimensionnements, masques...
  - Hypermédia
    - liens, animations (changements de propriétés, déplacements)...

Un exemple qui bouge !

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG December 1999//EN"
      "http://www.w3.org/Graphics/SVG/SVG-19991203.dtd">
<svg width="8cm" height="3cm" viewBox="0 0 800 300">
<desc>Example anim01 - demonstrate animation elements</desc>
<rect id="RectElement" x="300" y="100" width="300" height="100" style="fill:rgb(255,255,0)" >
    <animate attributeName="x" attributeType="XML" begin="0s" dur="9s" fill="freeze" from="300"
    to="0" />
    <animate attributeName="y" attributeType="XML"  begin="0s" dur="9s" fill="freeze" from="100"
    to="0" />
    <animate attributeName="width" attributeType="XML" begin="0s" dur="9s" fill="freeze" from="300"
    to="800" />
    <animate attributeName="height" attributeType="XML" begin="0s" dur="9s" fill="freeze" from="100"
    to="300" />
</rect>
<g transform="translate(100,100)" >
  <text id="TextElement" x="0" y="0" style="font-family:Verdana; font-size:35.27; visibility:hidden" > It's
    alive!
  <set attributeName="visibility" attributeType="CSS" to="visible" begin="3s" dur="6s" fill="freeze" />
    <animateMotion path="M 0 0 L 100 100" begin="3s" dur="6s" fill="freeze" />
    <animateColor attributeName="fill" attributeType="CSS" from="rgb(0,0,255)" to="rgb(128,0,0)"
    begin="3s" dur="6s" fill="freeze" />
  <animateTransform attributeName="transform" attributeType="XML" type="rotate" from="-30" to="0"
    begin="3s" dur="6s" fill="freeze" />
    <animateTransform attributeName="transform" attributeType="XML" type="scale" from="1" to="3"
    begin="3s" dur="6s" fill="freeze" />
    </text > </g>  </svg>
```


(D'après V. Quint)

MathML

- Objectifs
  - Intégrer proprement des expressions mathématiques dans les pages Web
  - Permettre l'échange de formules entre logiciels mathématiques
  - Représenter la structure de présentation et la structure mathématique des formules
- Exemples
  - $(a + b)^2$

Mathématique

```
<apply>
   <power/>
   <apply>
      <plus/>
      <ci>a</ci>
      <ci>b</ci>
   </apply>
   <cn>2</cn>
</apply>
```

Présentation

```
<msup>
   <mrow>
      <mo>(</mo>
      <mi>a</mi>
      <mo>+</mo>
      <mi>b</mi>
      <mo>)</mo>
   </mrow>
   <mn>2</mn>
</msup>
```

Espaces de noms

- Problème
  - Deux schémas ou DTD peuvent définir des éléments qui ont le même nom
  - Exemple :
    - DTD biblio : <!ELEMENT name (nom,prénom) >
    - DTD vcard : <!ELEMENT name (titre,prénom,nom) >
  - Question
    - Comment utiliser plusieurs DTD dans un unique document en évitant les collisions de noms ?
- Solution
  - Utiliser des « espaces de nom », « espaces de nommage », « vocabulaires » (*namespaces*)

Namespaces

- Spécification W3C
- Principes
  - On considère qu'un schéma (DTD) définit son propre espace de nom, dans lequel tous les noms d'éléments et d'attributs sont uniques
  - On dispose d'un mécanisme pour
    - identifier les espaces de nom utilisés dans le document
    - identifier pour chaque élément ou attribut à quel espace de nom il appartient
  - Ainsi,
    - toute référence à un nom d'élément est non ambiguë
    - un document unique peut contenir des informations définies dans plusieurs espaces de nom

Identification des namespace

- Beaucoup de standards ont une URI officielle
  - une URI est unique
- On peut utiliser l'URI pour identifier l'espace de nom
  - Pas forcément besoin d'un accès à Internet
    - L'URI devient une simple chaîne de caractères identifiant un schéma
- On « marque » les noms d'éléments et d'attributs en les préfixant avec l'URI ou un raccourci
  - `prefixe:nom`
  - aussi appelé Qname (nom qualifié)

Exemple d'utilisation

- On définit les espaces de nom avec des attributs
- Le nom de l'attribut est **`xmlns`**
  - On peut le spécifier n'importe où (auquel cas il est valable pour tous les sous-éléments)

```
<X:html xmlns:X="http://www.w3c.org/TR/REC-html40"
        xmlns:alan="file:/DTD/myDTD.dtd">
  <X:p>Un paragraphe HTML</X:p>
  <alan:p>Un paragraphe spécial</alan:p>
</X:html>
```

  - L'espace de nom par défaut peut être spécifié sans identificateur

```
<book xmlns="file:/DTD/myDTD.dtd"
      xmlns:X="http://www.w3c.org/TR/REC-html40" >
  <X:p>Un paragraphe HTML </X:p>
  <p>Un paragraphe spécial</p>
</book>
```


Autre exemple avec MathML

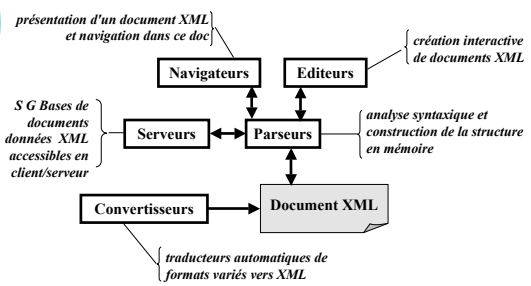
```
<?xml version="1.0"?>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>...</head>
<body>
<h1>Exemple</h1> ....
<math
  xmlns="http://www.w3.org/1998/Math/MathML">
  <mi>x</mi><mo>+</mo><mn>3</mn>
</math>
</body>
</html>
```


Espaces de noms et DTD

- On peut utiliser des préfixes dans les DTD
  - `<!ELEMENT document (feature, gene, sequence, collection:sequence, collection:list)*>`
- On peut inclure les définitions d'espace de nom dans les DTD
  - `<!ATTLIST document xmlns:collection #FIXED "file:/DTDs/collection.dtd">`
  - `// Implique un attribut fixé à l'élément "document",`
    `<document xmlns:collection = "file:/DTDs/collection.dtd">`

(cf. cours CNAM JM Pinon)

Différents types d'outils XML



Exemples d'outils

- Parseurs
  - SAX et DOM souvent intégrés directement dans les langages (Java, .NET, etc.)
- Editeurs
  - XML-Spy
  - Cooktop
  - XMetal...
- Navigateurs
  - Firefox, IE, *etc.*
- Convertisseurs
  - Nombreux outils avec format de sortie textuel
- SGB données/documents XML
  - Évolutions des SGBD classiques
  - SGBD dédiés

Autres outils

- XHTML / CSS
  - Dreamweaver, NVU...
- XSL
  - Style-vision...
- RDF
  - Outils du web sémantique...
- SMIL
  - Player : REAL ...
- SVG
  - Adobe...

DocBook

- Objectif
  - Codage de texte de documentation
  - Sorties multi-formats
- http://www.oasis-open.org/docbook/
- A voir en TP
  - sdocbook : sous-ensemble de balises

Open Document

- Objectif
  - codage des documents de suites bureautiques
    - textes, feuilles de calcul, présentations, dessins, formules, bases de données…
    - modèles pour ces documents
- Mis en place par OASIS
  - à partir des formats de OpenOffice (SUN)
- Version 1.0
  - mai 2005
- Utilisé dans Open Office 2.0
- Standard ouvert
  - intérêt pour le partage, la récupération, *etc.*
  - enjeu politique (Massachusetts, sept 2005)
    - réaction de Microsoft ?

Conclusion

- XML
  - Norme sortie en 1998
  - Unicode / généricité
  - Documents / données
  - Mondialement adoptée
- Standards et normes
  - Variés : dans tous les domaines nécessitant
    - Pérennité
    - Echange
  - Plus ou moins adaptés et adoptés
  - Questions récurrentes
    - Evolution
    - Interopérabilité

Annexes : EAD/EAC - TEI

- Pour info
  - quelques transparents sur ces normes utiles aux sciences de la documentation

EAD / EAC

- Objectif
  - Normalisation des instruments de recherche et des descriptions de contextes pour les archives
- EAD
  - Encoded Archival Description
- EAC
  - Encoded Archival Context
- Un point d'entrée
  - http://www.archivesdefrance.culture.gouv.fr/fr/archivistique/DAFlangage.html
- Il y a des spécialistes à l'ENSSIB

Les trois éléments principaux de l'instrument de recherche en XML/EAD

- Sous l'élément racine <ead> :

<eadheader>	en-tête EAD (description bibliographique de l'IR) (obligatoire)
<frontmatter>	préliminaires (page de titre, introduction, préface…)
<archdesc>	description archivistique (obligatoire)

Ce transparent et les suivants sur l'EAD sont extraits de « L'EAD, une DTD pour la rédaction, l'archivage et la diffusion des instruments de recherche archivistiques » F. Clavaud / 5 avril 2004

En-tête EAD <eadheader>

```
<eadheader>
    <eadid>Identifiant EAD (du fichier électronique)
    <filedesc>Description du fichier
                <titlestmt>Mention de titre et de responsabilité
                <titleproper>Titre propre de l'instrument de recherche
                <subtitle>Sous-titre de l'instrument de recherche
                <author> Auteur de l'instrument de recherche
                <editionstmt>Mention d'édition
      <publicationstmt>Mention de publication
      <seriesstmt>Mention de collection
      <notestmt> Mention de note
    <profiledesc> Description du profil
      <creation> Informations sur la création de l'inventaire
                <date> Date de l'inventaire
      <langusage> Langue utilisée
      <descrules> Règles de description (archivistique utilisée)
    <revisiondesc> Descriptions des révisions ; permet de gérer les versions
    successives de l'instrument de recherche encodé
</eadheader>
```


En-tête EAD <eadheader>

```
<eadheader audience="external" findaidstatus="edited-partial-draft" encodinganalog="DC">
    <eadid encodinganalog="identifier">FRDAFANCH00AP_000000001</eadid>
    <filedesc>
        <titlestmt>
            < titleproper encodinganalog="title">Etat sommaire des fonds d'archives privées du Centre historique des Archives
nationales</titleproper>
                <subtitle> Séries 317AP à 637AP et AB/XIX/368 à AB/XIX/5225</subtitle>
                <author encodinganalog="creator">Instrument de recherche rédigé par Claire Sibille, [...] encodé en XML conformément à la
DTD EAD puis converti en HTML par Martin Sévigny (société AJLSM) et Florence Clavaud (service des nouvelles technologies du
CHAN)</author>
        </titlestmt>
        <editionstmt> <edition>Quatrième édition </edition></editionstmt>
        <publicationstmt>
            <publisher encodinganalog="publisher">Centre historique des Archives nationales de France (CHAN)</publisher>
                <address>
                    <addressline>60 rue des Francs-Bourgeois</addressline>
                    <addressline>F-75141 PARIS CEDEX 03</addressline>
                </address>
                <date encodinganalog="date">août 2003</date>
        </publicationstmt>
    </filedesc>
    <profiledesc>
        <creation>Instrument de recherche produit sous Word et converti en XML [...] en <date> novembre 2001 </date>. Encodage relu et
complété par [...].</creation>
        <langusage>Instrument de recherche rédigé en <language langcode="fre">français</language>.</langusage>
    </profiledesc>
    <revisiondesc>
        <change>
            <date>8 septembre 2003</date>
            <item>fichier converti en EAD 2002 par programme XSL-T fourni par D. Pitti (SAA) adapté</item>        </change>
    </revisiondesc>
</eadheader>
```


Préliminaires <frontmatter> : exemple

```
<frontmatter>
        <titlepage>
            <publisher>Centre historique des Archives nationales</publisher>
            <titleproper>Etat sommaire</lb>des fonds d'archives privées</titleproper>
            <subtitle>Séries AP (317AP à 637AP) et AB XIX (AB/XIX/368 à AB/XIX/5225)</subtitle>
            <author>par Claire SIBILLE<lb>conservateur du Patrimoine</author>
            <author>avec la collaboration de Fanny AUDOUS et de Violaine LE NENAON </author>
            <publisher>Direction des Archives de France</publisher >
            <date>août 2003</date>
        </titlepage>
        <div>
            <head>Introduction</head>
            <p>Paru en 1973, l'état sommaire des fonds de la série AP (archives de personnes et de familles)
rédigé par Suzanne d'Huart et Chantal Bonazzi traitait des fonds privés 1AP à 315AP. Depuis, plus de 300
nouveaux fonds d'archives privées sont entrés aux Archives nationales. [...] Ces notices sont également
consultables dans la base <extref href="http://sdx.archivesdefrance.culture.gouv.fr/ap/">BORA</extref></p>
            <p>[...]</p>
        </div>
        <div>
            <head>Mode d'emploi</head>
            <p>[...]</p>
        </div>
[...]
</frontmatter>
```

Description archivistique <archdesc>

<archdesc level="niveau de description"> (nota : L'attribut level est obligatoire)

<did> *Identification et Description* ➔

<accessrestrict> et <userestrict> *Conditions d'accès et Conditions d'utilisation*
<accruals> *Accroissements*

<acqinfo> *Informations sur les modalités d'entrée*
<altformavail> *Documents de substitution*

<appraisal> *Informations sur l'évaluation*
<arrangement>*Classement*

<bibliography> *Bibliographie*

<bioghist> *Biographie ou Histoire (du producteur)*
<controlaccess> *Vedettes et accès contrôlés* ➔
<custodhist> *Historique de la conservation*
<dao> et <daogrp> *Objet archivistique numérique*
<note>*Note*
<odd>*Autres données descriptives*
<originalsloc> *Existence et lieu de conservation des documents originaux*

<otherfindaid> *Autre instrument de recherche*
<phystech> *Caractéristiques matérielles et contraintes techniques*
<prefercite> *Mention conseillée*

<processinfo> *Informations sur le traitement*

<relatedmaterial> et <separatedmaterial> *Documents en relation et documents séparés*

<scopecontent> *Présentation du contenu*
<dsc>*Description des sous-composants* ➔

</archdesc>

Les sous-éléments de <did> Identification et description

- <head> *Titre*
- <repository> *Organisme responsable de l'accès intellectuel*
- **<origination>** *Origine*
- **<unittitle>** *Intitulé de l'unité documentaire*
- **<unitdate>** *Date de l'unité documentaire*
- **<unitid>** *Identifiant de l'unité documentaire*
- **<physdesc>** *Description physique*
- <physloc> *Localisation physique*
- <container> *Unité de conditionnement*
- <note> *Note*
- <dao> *Objet archivistique numérique*
- <daogrp> *Groupe d'objets archivistiques numériques*
- <abstract> *Résumé*
- <langmaterial> *Langue des unités documentaires*
- <materialspec> *Particularités matérielles*

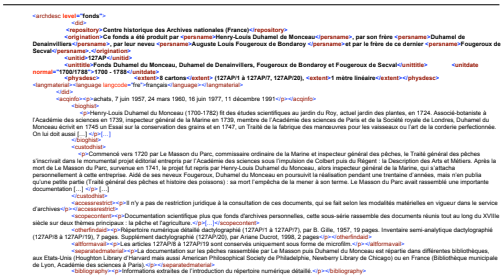
Exemple d'utilisation de <did>

```
<archdesc level="fonds">

        <did>
                <unitid>320 AP</unitid>
                <unittitle>Fonds Bazaine</unittitle>
                <unitdate normal="1801/1900"> XIX<emph
render="super">e</emph> siècle</unitdate>
                <origination>Bazaine, Achille</origination>
                <physdesc> 7  cartons (320AP/1 à 320AP/7),
0,75 mètres linéaires</physdesc>
        </did>

<!-- Etc. : les autres sous-éléments de archdesc -->
```


<archdesc> : exemple de description globale d'un fonds



TEI

- Text Encoding Initiative
- Lancé en 1987
  - incompatibilité totale des formats
  - chercheurs du domaine de l'archivagen structuration et analyse des textes electroniques
  - pour permettre la préparation et l'échange de textes électroniques
- Principes
  - – être aussi complet que possible,
  - – être simple, clair et concret,
  - – être facile à utiliser sans logiciel particulier,
  - – être rigoureusement défini,
  - – permettre un traitement efficace,
  - – être ouvert à des extensions définies par les utilisateurs,
  - être compatible avec les standards existants ou en développement.

TEI : suite de l'histoire

- Financé sur des programmes « Humanities »
- Sortie de la norme (Recommandation*)* en mai 1994
- Ensemble de conventions de codage utilisables dans une grande variété d'applications
  - publication électronique, analyse littéraire et historique, lexicographie, traitement automatique des langues, recherche documentaire, hypertexte, etc.
  - Textes écrits ou parlés, sans restriction de langue, de période, de genre ou de contenu
- Répondent aux besoins fondamentaux de nombreux utilisateurs
  - lexicographes, linguistes, philologues, bibliothécaires
    → tous ceux qui sont concernés par l'archivage et l'accès à des documents électroniques.

TEI : codage

- Construction de la DTD de façon modulaire :
  - jeu de balises «noyau» *(core tag set)* composé d'éléments communs à tous les types de textes (divisions, paragraphes, *etc.*)
  - des ensembles de balises de base *(base tag sets)* pour chaque type particulier de texte (prose, poésie en vers, *etc.*)
  - des jeux de balises additionnelles *(additional tag sets)* pour des mécanismes particuliers qui peuvent se superposer à n'importe quel type de texte (liens hypertextuels, *etc.*).
- Noyau obligatoire, autre éléments facultatifs
- Importance de l'en-tête
  - codage systématique des méta-données de n'importe quel document électronique

TEI : où en est on ?

- Au départ : SGML, puis XML
- Utilisation dans de très nombreux projets
- Reste la question du codage
  - Pour qui code-t'on ?
  - Un chercheur peut-il réutiliser le codage d'un autre ? Ajouter au codage d'un autre ?
  - Possibilité de mixer avec d'autres formats ? (particulièrement docbook)
  - *etc.*