XML & DTD

Aurélien Tabard, Université Lyon 1

Basé sur les cours de Yannick Prié

Objectifs du cours

- Être capable de comprendre des documents XML et des DTD
- Être 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 "/Users/aurelien/Cours/CCI/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="longueur" 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 fichier XML

- Prologue
  - En-tête 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"] ?>
```

C’est une des informations de traitement

Cela 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"**

Éléments : 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: ****

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> ... </section>**
  - **<section>**
    ** ... **
    </section>
- Les éléments sont ordonnés

Caractères spéciaux

Ces caractères ont une signification spéciale pour les parsers XML

Il faut les écrire différemment :

- < <
- > >
- & &
- ‘ '
- “ "

Attributs : 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 ensembles 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="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

Instructions de traitement

Informations nécessaire à une application externe

Format :

```
<?NomApplication paramètres ?>
```

Exemples :

- Déclaration XML obligatoire en début de fichier

```
<?xml version='1.0' ?>
```

- Déclaration de feuille de style à utiliser

```
<?xml-stylesheet href="fichier.xsl"
  type="text/xsl"?>
```

Déclaration

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 *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>>>]]>`

Au bilan : dans un document XML

Prologue

- en-tête
- déclaration de DTD
- 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

Un fichier

- 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

- La DTD est dans le document XML (inline)
- Le document XML réfère à la DTD avec une URI

DTD et document XML

Valide,
contraint

DTD

```
<!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)>
```

```
<?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>
```

Document XML

Est validé par,
Est contraint par

Déclarations d'éléments

Définir un élément et son contenu

- `<!ELEMENT name (#PCDATA)>`
  ⇒ `<name> ... </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
  - Chaine de caractères
  - `<!ATTLIST person name CDATA ... >`
  - `name = "Tom Jones"`
- NMTOKEN
  - Token unique
  - `<!ATTLIST mug color NMTOKEN ... >`
  - `color="red"`
- NMTOKENS
  - Multiples tokens
  - `<!ATTLIST temp values NMTOKENS ... >`
  - `values="12 15 34"`

Joue sur la manière dont le parser interprète l’attribut

Le nom doit être un nom XML valide

Types d’attributs (2)

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

Types d’attributs (3)

- IDREFS
  - Référence à plusieurs ID
  - `<!ATTLIST person children IDREFS ... >`
  - `IDREFS="A01 A02"`
- NOTATION
  - Décrit des données non XML
  - `<!ATTLIST image format NOTATION (TeX|TIFF)...>`
  - `FORMAT="TeX"`
- Name group
  - Liste restreinte
  - `<!ATTLIST point coord (X|Y|Z)...>`
  - `coord="X"`

Types d’attributs

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 seqlist sepchar NMTOKEN        #REQUIRED
                  type    (alpha|num)    "num"
```

Mise en place de 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

Définition interne de DTD

Dans la déclaration DOCTYPE

```
<?xml version="1.0" standalone="yes" ?>
<!DOCTYPE racine [
    <!-- Ici la DTD -->
    <! ... >
    <! ... >
]>
<!-- Rest of XML file -->
<racine>
...
</racine>
```

Définition externe privée de DTD

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

```
<?xml version="1.0" standalone="no" ?>
   <!DOCTYPE racine
      SYSTEM "./MyDoc.dtd" [
      <!-- Extra declarations -->
      <! ... >
      <! ... >
   ]>
   <!-- Rest of XML file -->
```

DTD externe privée

- Les déclarations spécifiques au document restent définies de façon interne

Définition externe privée de DTD

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

```
<?xml version="1.0" standalone="no" ?>
   <!DOCTYPE racine
      SYSTEM "http://.../MyDoc.dtd" [
      <!-- Extra declarations -->
      <! ... >
      <! ... >
   ]>
   <!-- Rest of XML file -->
```

DTD externe privée

- 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">`

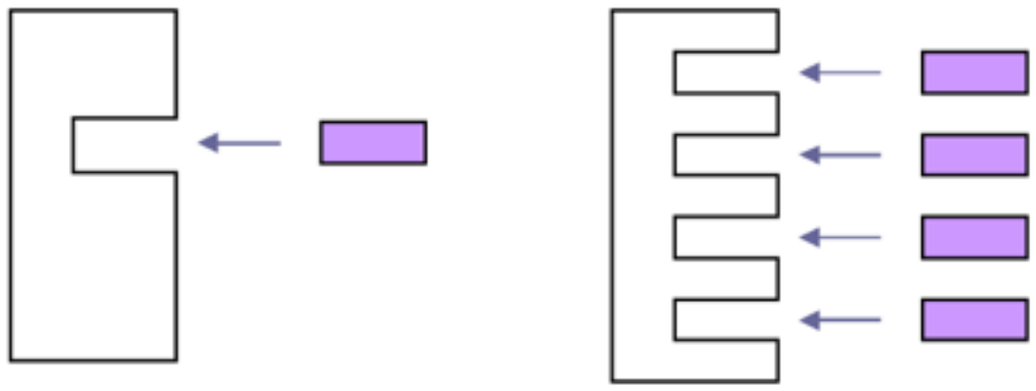
Définition externe privée de DTD

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

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



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 "chaine 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; </doc>
```

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

```
<!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

```
<!DOCTYPE mybook PUBLIC "-//EBI//DTD My book//EN" "url">
```

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

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

ou?

```
<book title="TheForty-nineSteps">
   ...
</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
- 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 chaine de caractères
- Impossible de spécifier des types simples
  - Entiers, dates, etc.
- Impossible de spécifier des cardinalités simples
  - Un ARTICLE aura entre 1 et 4 MOTS-CLE
- Impossible de spécifier des contraintes simples
  - 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 pratique XML -> DTD

```
<?xml version="1.0"?>
<!DOCTYPE PARTS SYSTEM "parts.dtd">
<?xml-stylesheet type="text/css"
href="xmlpartsstyle.css"?>
<PARTS>
   <TITLE>Computer Parts</TITLE>
   <PART type="computer">
      <ITEM>Motherboard</ITEM>
      <MANUFACTURER>ASUS</
MANUFACTURER>
      <MODEL>P3B-F</MODEL>
      <COST> 123.00</COST>
   </PART>
   <PART>
      <ITEM>Video Card</ITEM>
      <MANUFACTURER>ATI</
MANUFACTURER>
      <MODEL>All-in-Wonder Pro</
MODEL>
      <COST> 160.00</COST>
   </PART>
   <PART>
      <ITEM>Sound Card</ITEM>
      <MANUFACTURER>Creative Labs</
MANUFACTURER>
      <MODEL>Sound Blaster Live</
MODEL>
      <COST> 80.00</COST>
   </PART>
   <PART>
      <ITEM>9 inch Monitor</ITEM>
      <MANUFACTURER>LG Electronics</
MANUFACTURER>
      <MODEL> 995E</MODEL>
      <COST> 290.00</COST>
   </PART>
</PARTS>
```

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 ou
  - 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 standards XML

- The XML Bookmark Exchange Language (XBEL)
- Open eBook Publication Structure
- SportsML
- NewsML
- XML Book Industry Transaction Standards (XBITS)
- 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 totqui ont un attribut name
- `//tata/descendant::*`
  - Tous les descendants des éléments tata

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:label="fr"
  xlink:href=http://www.xml.fr/FAQ.xml
  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 titre de doc.xml
- http://site.fr/page.xml|id('ref12").child(l,session)
  - premier élément session enfant de l'élément identifié par ref12, dans le document page.xml

XSL

Ensemble d'outils permettant de

- Visualiser les documents XML sous forme lisible, pour de multiples supports
- Transformer les documents XML en d'autres documents XML (changement de format)

eXtensible Style Language

- XSL-FO(«XSLFormattingObjects)
  - présenter des informations
- XSLT(XSLtransformation)
  - transformer un arbre XML en un autre arbre XML

Xquery

Standard XML permettant d'exprimer des requêtes dans les documents XML

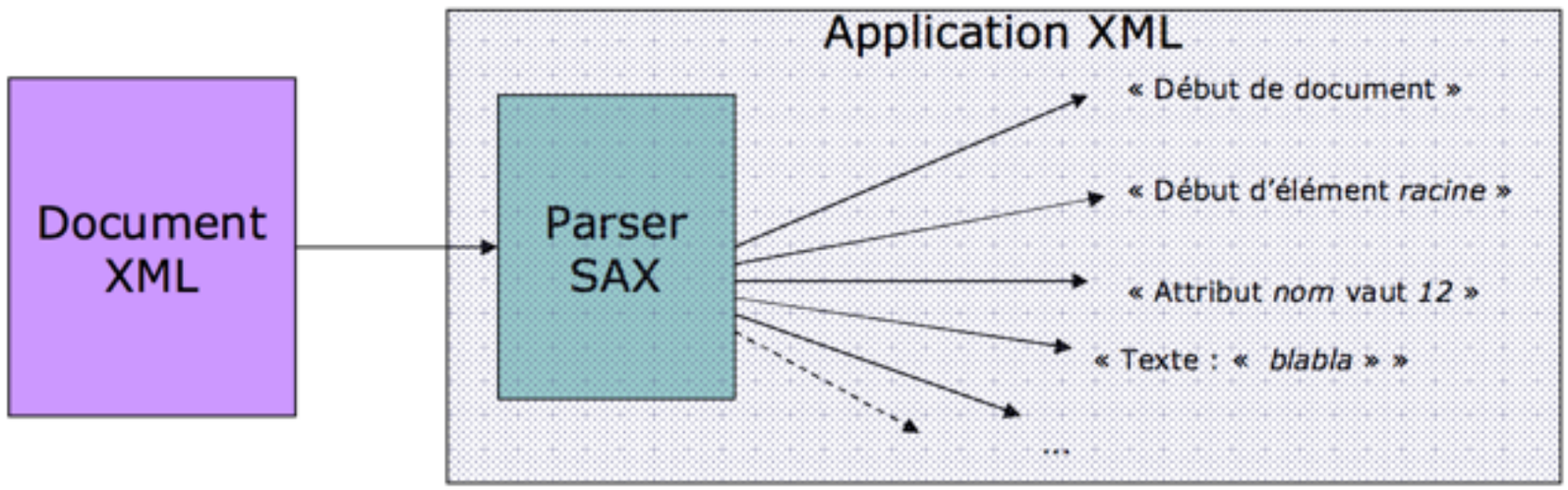
Syntaxe XML ou non

Utilisation de Xpath

XML et les applications : SAX

SAX : Simple API for XML

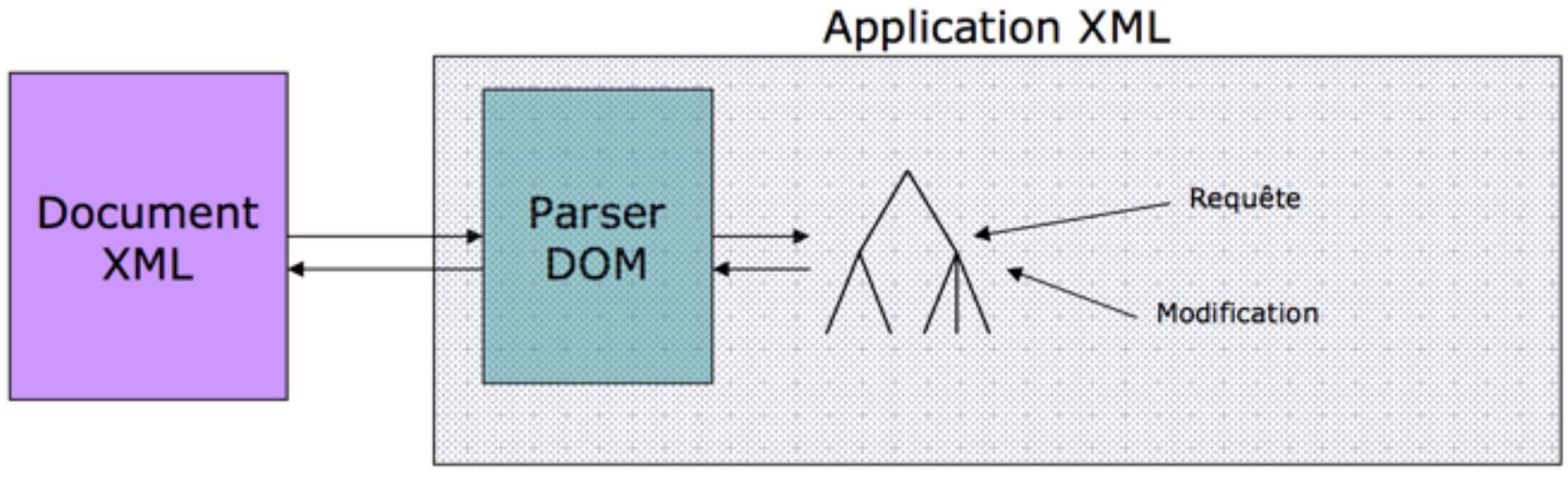
- Principe : un parser SAX lit un document XML, et envoi un message à une application dès qu'il :
  - rencontre un début ou une fin de document
  - rencontre un début ou une fin d'élément
  - trouve des caractères dans un événement, etc.
- Le programmeur qui écrit l'application XML décide quoi faire 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, rectanges, etc.), images, textes, groupes d'éléments, ...
- Affichage
  - opacités, redimensionnements, masques, ...
- Hypermédia
  - liens, animations (changements de propriétés, déplacements), ...

SVG exemple

https://commons.wikimedia.org/wiki/SVG_examples

MathML

https://en.wikipedia.org/wiki/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

- $(a + b)^2$

Content MathML

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

Presentation MathML

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

Espace 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 ambigue
  - Un document unique peut contenir des informations définies dans plusieurs espaces de nom.

Identification des *namespaces*

- 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 ce 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>An HTML paragraph</X:p>
   <alan:p>My own special p-value markup</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>An HTML paragraph</X:p>
   <p>My own special p-value markup</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>
```

Démo : http://localhost/~aurelien/mathml.xhtml

Espaces de noms et DTD

On peut utiliser des préfixes dans les DTD

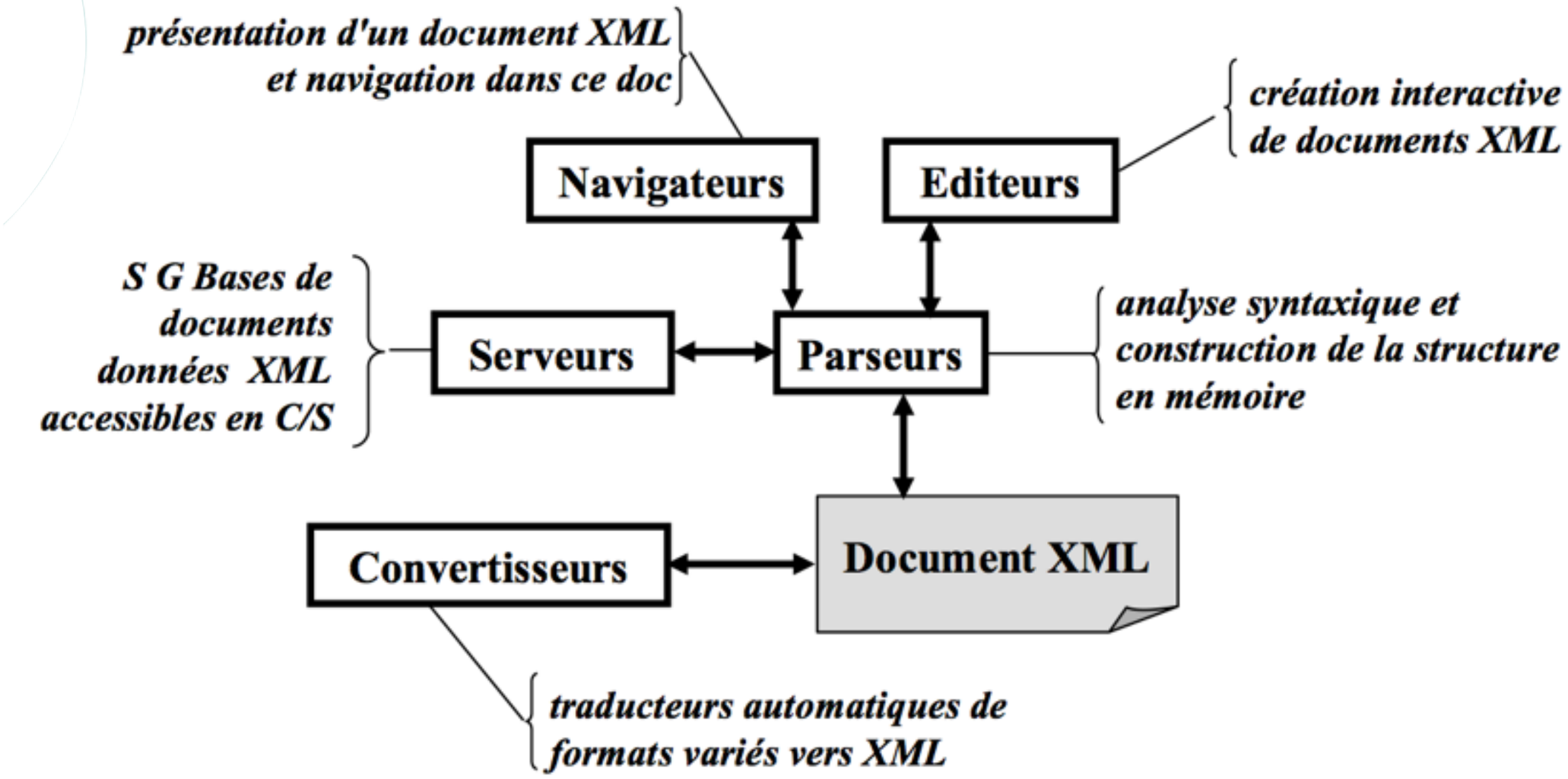
- `<!ELEMENT document(feature,gene,sequence, collection:sequence, collection:list)*>`

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, Chrome, 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...

XSL

- Style-vision...

RDF

- Outils du web sémantique...

SMIL

- Player:REAL...

SVG

- Inkscape, Adobe…

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

Remerciements

Ce cours s’appuie largement sur celui de Yannick Prié, lui même basé sur celui d’Alan Robinson http://industry.ebi.ac.uk/~alan/XMLWorkshop/ et reprenant des éléments du cours CNAM de Tiphaine Accary, lui-même basé sur celui de Jean-Marie Pinon.