UNIVERSITÉ DE SHERBROOKE

Faculté de Génie

Chaire industrielle en infrastructures de communication

La technologie XML

Wajdi Elleuch

Octobre 2004

SOMMAIRE

Content :

Du HTML vers le XML

XML (e Xtensible **M**arkup **L**angage**)**

XML = langage HTML amélioré permettant de définir de nouvelles balises

HTML : un langage limité !

XML : Définir de nouveau domaines de données

Séparer le contenu de la présentation (plus de liberté de présentation)

Historique d'XML…

Descendant de SGML (ISO 8879, 1986)

→ **SGML = Standard Generalized Markup Language**

- **Invention : 1970's (avant le WWW)**
- **Spécification : 150 pages**
- **Intègre tous les langages de «markup» (très puissant)**
- **Application SGML connue : HTML**
- **Très complexe (pas d'implémentation unifiée)**

→ **XML = version «légère» de SGML**

- **Début : 1996**
- **Normalisation par le W3C : 02/1998**
- **Version : 1.0 (jusqu'à maintenant)**

Les avantages de XML

→ **La lisibilité :** aucune connaissance ne doit théoriquement être nécessaire pour comprendre un contenu d'un document XML

→ **Autodescriptif et extensible** : Une structure arborescente permettant de modéliser la majorité des problèmes informatiques

→ **Universalité et portabilité** : les différents jeux de caractères sont pris en compte

→ **Déployable** : il peut être facilement distribué par n'importe quels protocoles capable de transporter du texte, comme HTTP

→ **Intégrabilité :** un document XML est utilisable par toute application pourvue d'un parseur (c'est-à-dire un logiciel permettant d'analyser un code XML)

→ **Extensibilité :** un document XML doit pouvoir être utilisable dans tous les domaines d'applications

L’extensibilité de XML

L'intérêt de disposer d'un format commun d'échange d'information dépend du contexte professionnel dans lequel les utilisateurs interviennent :

VoiceXML : Description des services vocaux intéractifs.

CCXML : ***C****all* ***C****ontrol e****X****tensible* ***M****arkup* ***L****anguage* - Programmation de SERVEUR téléphonique.

HTML : ***H****yper* ***T****ext* ***M****arckup* ***L****anguage* – Pour la description du contenu des pages web.

CML : ***C****hemical* ***M****arkup* ***L****anguage* - Permet de décrire des composés chimiques.

SMIL : ***S****ynchronized* ***M****ultimedia* ***I****ntegration* ***L****anguage* - Permet de créer des présentations multimédia en synchronisant diverses sources : audio, vidéo, texte,...

SOAP : ***S****imple* ***O****bject* ***A****ccess* ***P****rotocol* – Description des Services Web

Structure d'un document XML

La **norme XML** en tant que telle doit être vue comme un outil permettant de définir un langage (on dit alors qu'il s'agit d'un métalangage),

Une balise est une chaîne de caractère du type: <balise>

```
<annuaire>
  <personne class = "etudiant">
     <nom>Desjardins</nom>
     <prenom>Jean-Philippe</prenom>
     <telephone>(819) 234 2343</telephone>
     <email>webmaster@usherbrooke.ca</email>
<!-- insertion de commentaires XML -->
  </personne>
 <personne>
    ...
 </personne>
</annuaire>
```

Structure d’un document XML

Un document XML est structuré en 3 parties:

- La première partie, appelée ***prologue*** : permet d'indiquer la version de la norme XML utilisée pour créer le document ainsi que le jeu de caractères utilisé dans le document :

<?xml version="1.0" encoding="ISO-8859-1"?>

- Le second élément est une déclaration de type de document (déclaration de la DTD ou du Schéma, de la feuille de style)(optionnel)

- Et enfin la dernière composante d'un fichier XML est l'arbre des éléments

La syntaxe des éléments en XML

-Encapsulation dans les balises:

<balise> *contenu de la balise* ***</balise>***

-Utilisation des attributs :

<balise *attribut_01 =* ***"****text****"*** *attribut_02****= "****text****" >***

-Pas de chevauchement de balises :

<balise_01>
<balise_02>
</balise_01>
</balise_02>

- Balise Vide

<balise> </balise> = <balise/>

XML : Une solution d'échange

→ **La nouvelle tendance :** Exploitation des réseaux et développement d'application réparties, (efficacité, réutilisation et accessibilité)

→ **XML** : Solution pour assurer l'échange de données entre systèmes distribués ?

3 concurrents à XML :

- **COM/DCOM** (**Component Object Model / Distributed COM**)
- **CORBA** (**Common Object Request Broker Architecture**)
- **Java RMI** (**Remote Method Invocation**)

XML : Une solution d'échange (comparatif)

Thème	DCOM	CORBA	Java / RMI	XML
types de données	types primitifs et objets	types primitifs et objets	types primitifs et objets	Chaînes de caractère
multi-langages	oui	oui	non (tout Java)	indépendant
multi-systèmes	non (Microsoft – Win 32)	oui	oui	oui
propriétaire	Microsoft	non (norme)	Sun	non
Interfaces	MIDL	IDL	interfaces Java	DTD / XML-schéma
Support des exceptions aux interfaces	Non	Oui	Oui	Non
Transport	RPC	IIOP - GIOP	JRMP ou IIOP	Indépendant

Validation des documents XML

XML fournit un moyen de vérifier la syntaxe d'un document

Utilisation des **XML Schema ou des DTD (*Document Type Definition*)** pour décrire la structure des documents avec les imbrications des éléments possibles

- Document bien formé : Un document suivant les règles de XML

- Document valide : Un document XML possédant une DTD (ou XML Schema) et étant conforme à celle-ci

LES DTD (Document Type Definition)

- Type de données :

Type prédéfini	Description
ANY	L'élément peut contenir tout type de données
EMPTY	L'élément ne contient pas de données
#PCDATA	L'élément doit contenir une chaîne de caractère

- Exemple :

```
<! ELEMENT Nom_élement  (#PCDATA) >
```

LES DTD (Document Type Definition)

- Occurrence des éléments :

Opérateur	Signification	Exemple
+	L'élément doit être présent au minimum une fois	A+
*	L'élément peut être présent plusieurs fois (ou aucune)	A*
?	L'élément peut être optionnellement présent	A?
|	L'élément A **ou** B peuvent être présents (pas les deux)	A|B
,	L'élément A doit être présent et suivi de l'élément B	A,B
()	Les parenthèses permettent de regrouper des éléments afin de leur appliquer les autres opérateurs	(A,B)+

- Exemple :

<!ELEMENT personne (nom,prenom),telephone+,email? >

Carnet_adresse.XML

```
<carnet_adresse>
 <personne>
    <nom>Desjardins</nom>
    <prenom>Jean-Philippe</prenom>
    <telephone>(819)55-23456</telephone>
    <telephone>(514)123-5336</telephone>
    <email>ss@mail.net</email>
 </personne>
 <personne>
     <nom>Cartier</nom>
     <prenom>Jacques</prenom>
     <telephone>(819)235-1234</telephone>
 </personne>
</carnet_adresse>
```

Carnet_adresse.DTD

```
<!ELEMENT carnet_adresse (personne)*>

<!ELEMENT personne(nom,prenom),telephone+,email?>

<!ELEMENT nom (#PCDATA) >

<!ELEMENT prenom (#PCDATA) >

<!ELEMENT telephone (#PCDATA) >

<!ELEMENT email (#PCDATA) >
```

LES DTD (Document Type Definition)

- Déclatation des attributs

Type prédéfini	Description
#IMPLIED	signifie que l'attribut est optionnel, c'est-à-dire non obligatoire
#REQUIRED	signifie que l'attribut est obligatoire
#FIXED	signifie que l'attribut sera affecté d'une valeur par défaut s'il n'est pas défini.

- Exemple :

```
<! ATTLIST disque IDdisk #REQUIRED type (K7|MiniDisc|Vinyl)"CD" >
```

Chaque élément disque est décrit par deux attributs :

- **IDdisk :** Un identifiant obligatoire
- **type :** peut contenir la valeur «K7», «MiniDisc», «Vinyl» ou par défaut «CD »

LES XML-Schema

Avantages par rapport au DTD

→ **Codé sous forme XML :** Les XML Schema utilisent un système de balisage conforme à XML

→ **Meilleur typage des données :** On peut définir de nouveaux types de données (String, Date, Interger, Positif-Integer, Real, Boolean…)

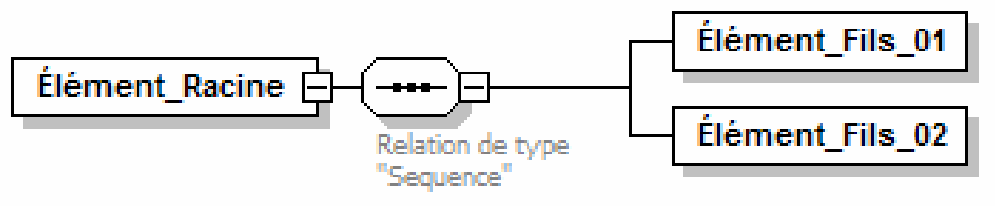
→ **Meilleure définition des occurrences :** Utilisation de modèle de relation entre éléments et définition des occurrences de chaque éléments dans un intervalle pouvant varier de 0 à l'infini

→ **Prise en charge de modèle réutilisable de données** (notion d'objet)

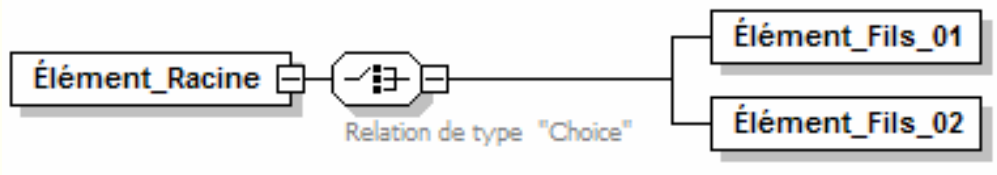
LES XML-Schema

Modélisation graphique (Les relations père-fils)

→ **Relation de type "Sequence" : L'élément père peut inclure un ou plusieurs fils**



→ **Relation de type "Choice" : L'élément père doit inclure seulement un fils**



→ **Relation de type "All" : L'élément père doit inclure tous ses fils**

LES XML-Schema

Modélisation graphique (Occurrence des éléments)

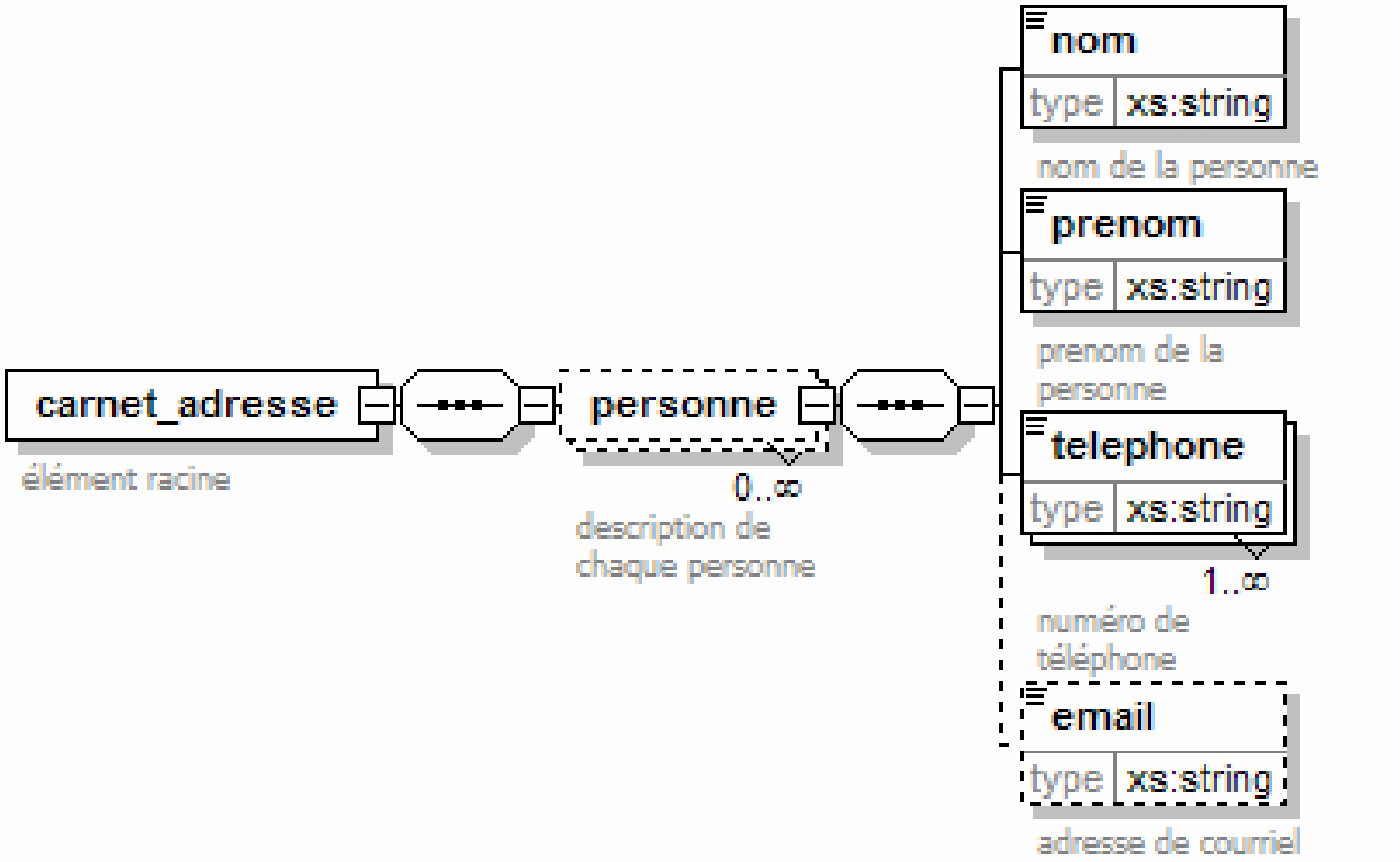
→ **Élément Unique : le nombre maximum et le nombre minimum d'occurrence = 1**

→ **Élément Optionnel : le nombre minimum d'occurrence = 0**

→ **Élément Multiple : le nombre maximum d'occurrence > 1**

→ **Éléments Optionnel ou Multiple : le nombre minimum = 0 et le nombre maximum d'occurrence > 1**

Carnet_adresse.xsd



Les espaces de nom (name-space)

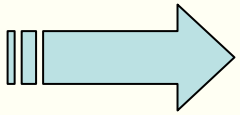
→ **Problème :**

- Possibilité d'un conflit : Si deux langages basés sur XML sont intégrés dans

→ **Solution :**

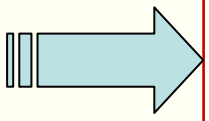
- Définir pour chaque langage XML un espace de nom différent.

```
<table>
<tr>
<td>Pommes</td>
<td>Melons</td>
</tr>
</table>
```

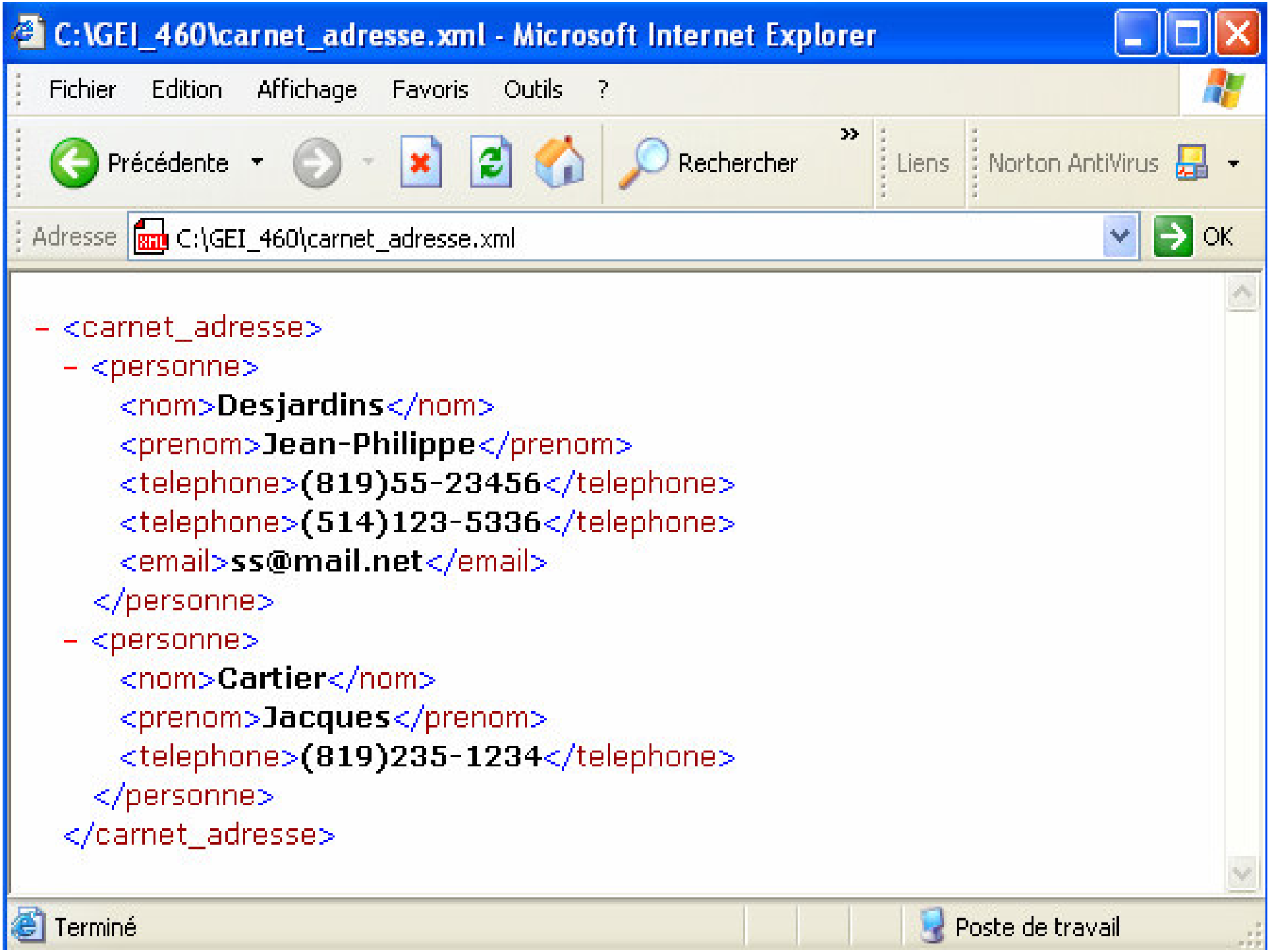


```
<h:table xmlns:h="http://www.w3.org/TR/html4/"> <h:tr>
<h:td>Pommes</h:td>
<h:td>Melons</h:td>
</h:tr>
</h:table>
```

```
<table>
<name>Table chinoise</name>
<width>80</width>
<length>120</length>
</table>
```



```
<f:table xmlns:f="http://www.w3schools.com/fourniture">
<f:name>Table chinoise</f:name> <f:width>80</f:width>
<f:length>120</f:length>
</f:table>
```


C:\GEI_460\carnet_adresse.xml - Microsoft Internet Explorer
Fichier Edition Affichage Favoris Outils ?
Précédente
Rechercher
Liens
Norton AntiVirus
Adresse
C:\GEI_460\carnet_adresse.xml
OK
- <carnet_adresse>
- <personne>
<nom>Desjardins</nom>
<prenom>Jean-Philippe</prenom>
<telephone>(819)55-23456</telephone>
<telephone>(514)123-5336</telephone>
<email>ss@mail.net</email>
</personne>
- <personne>
<nom>Cartier</nom>
<prenom>Jacques</prenom>
<telephone>(819)235-1234</telephone>
</personne>
</carnet_adresse>
Terminé
Poste de travail

Introduction à XSL

La mise en page des données est assurée par un langage de mise en page tiers.

A l'heure actuelle il existe deux solutions pour mettre en forme un document XML :

- ***CSS*** (***C****ascading* ***S****tyle****S****heet*), l'ancienne solution des années fin 90. c'est un standard qui a déjà fait ses preuves avec HTML mais qui présente beaucoup de limitations.

- XSL (*e****X****tensible* ***S****tylesheet* ***L****anguage*), un langage de feuilles de style extensible développé spécialement pour XML. Il offre plus de souplesse d'accès aux fichiers XML

Introduction à XSL

XML est un langage de structuration des données, et non de représentation des données. Ainsi **XSL** (*e**X**tensible **S**tyleSheet **L**anguage*) est un langage recommandé par le W3C pour effectuer la représentation des données de documents XML.

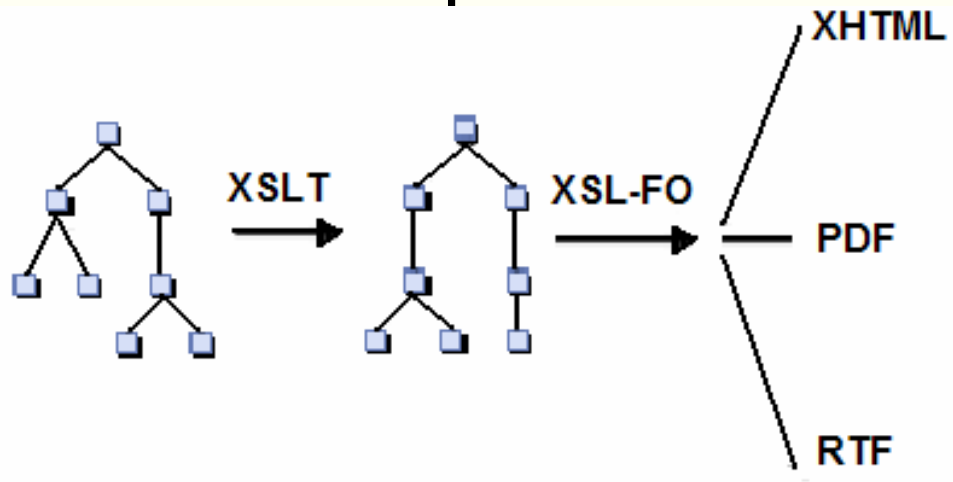
XSL permet aussi de:

→ Retraiter un document XML afin d'en modifier sa structure

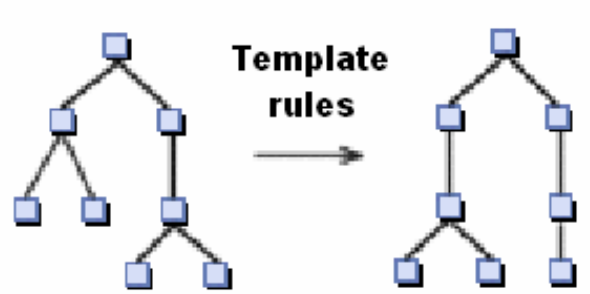
→ Générer d'autres types de documents (PostScript, XHTML, PDF, RTF, DOC, ...) ou bien un autre fichier XML de structure différente.

Les normes de XSL

XSL possède deux composantes :



- Le langage de transformation des données (**XSLT**, *e**X**tensible **S**tylesheet **L**anguage **T**ransformation*) permettant de transformer la structure des éléments XML



Les normes de XSL

→ **Le langage de formatage des données (XSL/FO),**

- Un langage permettant de définir la mise en page de ce qui a été créé par XSLT.

- Une fois l'arbre source créé, *XSL/FO* permet de formater le résultat, c'est-à-dire d'interpréter l'arbre résultat, ou plus exactement les objets de flux le composant en leur appliquant des *objets de mise en forme* afin d'en faire une représentation visuelle (papier, écran (PDF, RTF, PS,HTML...)

Structure d'un document XSL

Carnet_adresse.XSL

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet
xmlns:xsl="http://www.w3.org/TR/WD-xsl"
xmlns="http://www.w3.org/TR/REC-html40"
result-ns="">

<xsl:template ... >
<!-- traitements à effectuer -->
</xsl:template >

</xsl:stylesheet>
```

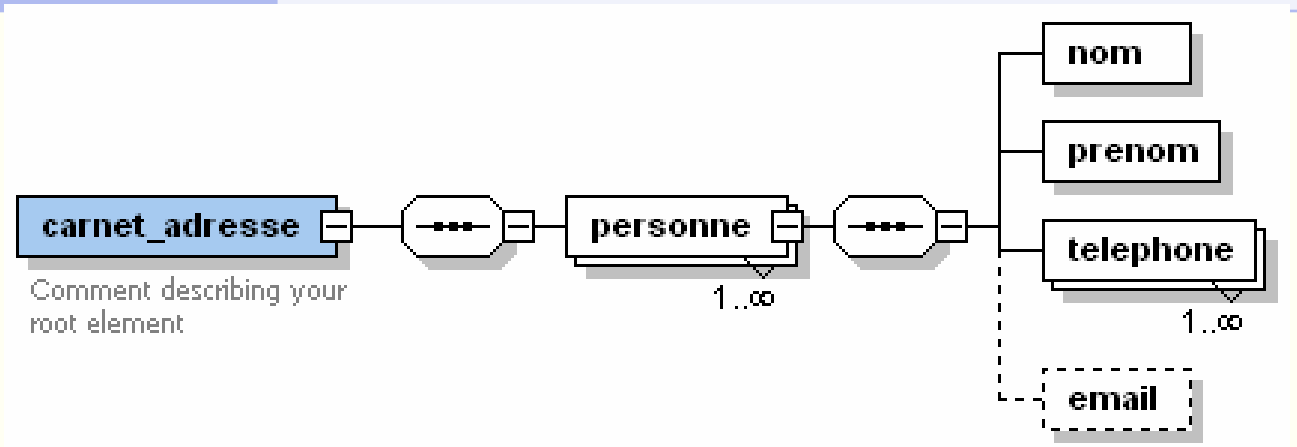
Association d'une feuille XSL à un document XML

carnet_adresse.XML

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet href= "carnet_adresse.xsl" type="text/xsl"?>
<carnet_adresse>
 <personne>
    <nom>Desjardins</nom>
    <prenom>Jean-Philippe</prenom>
    <telephone>(819)55-23456</telephone>
    <telephone>(514)123-5336</telephone>
    <email>ss@mail.net</email>
 </personne>
 <personne>
     <nom>Cartier</nom>
     <prenom>Jacques</prenom>
     <telephone>(819)235-1234</telephone>
 </personne>
</carnet_adresse>
```

Le XPath :

Pattern	Exemple	Signification
|	**Gauche|Milieu**	Indique une alternative (un noeud ou bien l'autre (ou les deux))
/	**personne/nom**	Chemin d'accès aux éléments (*personne/bras/gauche*) au même titre que l'arborescence utilisée généralement pour les fichiers (*/usr/bin/toto*)
*	*	Motif "joker" désignant n'importe quel élément
//	**//personne**	Indique tous les descendants d'un noeud
.	.	Caractérise le noeud courant
..	..	Désigne le noeud parent
@	**@valeur**	Indique un attribut caractéristique (dans l'exemple l'attribut *value*



Le fichier XSL

Carnet_adresse.XSL

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl"
xmlns="http://www.w3.org/TR/REC-html40" result-ns="">

<xsl:template match="/">
    <HTML> <HEAD> <TITLE>Titre de la page</TITLE> </HEAD> <BODY
    BGCOLOR="#FFFFFF">

     <xsl:apply-templates/>

    </BODY> </HTML>
 </xsl:template >

<xsl:template match="personne" >
    <ul> <li>
    Voici le nom : <xsl:value-of select = "nom"  />
    Voici le prénom :  <xsl:value-of select = "prenom" />
    </li> </ul>
</xsl:template >
</xsl:stylesheet>
```

Les parseurs

Définition : Un parseur est un outil logiciel permettant de parcourir un document et d'en extraire des informations. Pour la technologie XML, il existe deux types d'approche :

→ **DOM** (*Document Object Model*) : API utilisant une approche **hiérarchique** pour construire une structure contenant des objets représentant les éléments du document, et dont les méthodes permettent d'accéder aux propriétés.(*GetRootElement(), GetChildren(), GetElement()…*)

→ **SAX** (*Simple API for XML* : API basés sur un mode **événementiel** permettent de réagir à des événements comme le début d'un élément ou la fin d'un élément (*startDocument() , startElement(), characters(), endElement()…*)

Les parseurs

	Avantages	Inconvénients
DOM	- Parcours libre de l'arbre - Possibilité de modifier la structure et le contenu de l'arbre	- Gourmand en mémoire - Doit traiter tout le document avant d'exploiter les résultats
SAX	- Peut gourmand en ressources mémoire - Plus rapide pour les documents volumineux - Permet de ne traiter que les données utiles	- Traite les données séquentiellement - Un peu plus difficile à programmer, il est souvent nécessaire de sauvegarder des informations pour les traiter

XML pour le stockage ?

→ **Stocker dans des fichiers XML tout simplement** (problèmes : intégrité des données, gestion des accès concurrents, performances …)

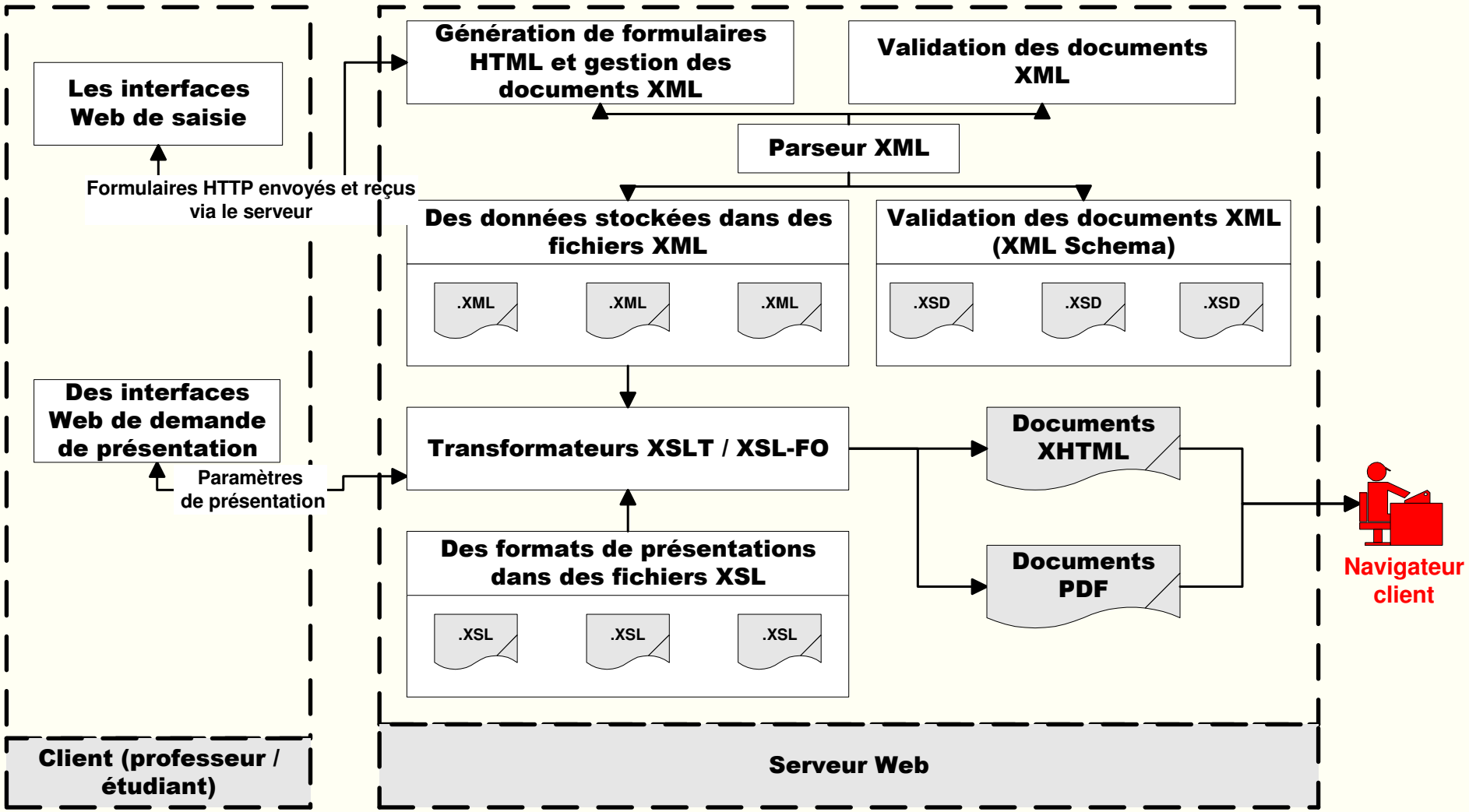
→ **Utiliser des bases de données relationnelles (SGBD)** (Problèmes : beaucoup de tables et de jointures, performances, complications, ….)
Prévoir un champ BLOB (Binary Large OBject) ou un CLOB (Character Large OBject)

→ **Utiliser des bases de données orientées objet (SGBDOO)** (Des bases de données coûteuses, couplage faible entre objet et XML)

→ **Utiliser des bases de données XML Native (NXD)**

Quelques produits : Tamino (SoftwareAG), Ipedo XML Database (IPedo), TextML (IXiasoft)

Étude d'un cas pratique



Pour résumer !

→ **Principe de l'utilisation de XML :** Un excellent format d'échange et de codage de données (Création dune structure descriptive adaptée aux données à traiter)

→ **Validation d'un document XML**

-**DTD** (ancienne technologie)

-**XML Schema** (très populaire en ce moment, plus flexible, plus riche que la DTD)

→ **Transformation et présentation d'un document XML par XSL**

- **XSLT** (Parcourir pour transformer)
- **XSL-FO** (Parcourir pour présenter)

→ **Transformation et exploitation d'un document XML par parseur**

- **DOM** : Utilisation d'une approche Hiérarchique
- **SAX** : Utilisation d'une approche Événementielle

→**Stockage de documents XML**

- Dans des **fichiers** (s'il n'y a pas d'accès multiples en écriture)
- Dans des **bases de données XML natives** (nouvelle solution en cours d'amélioration)

Conclusion

La spécification XML va-t-elle remplacer HTML ?

Qu'apporte donc de plus le choix d'un format XML, comparé à un simple fichier texte ?

- **Lisibilité et interopérabilité**
- **Caractère standard**
- **Souplesse**

Quels sont les logiciels nécessaires pour travailler avec XML ?

- **Parseur et un environnement de programmation compatible**
- **Editeur XML (XMLSPY, XML Viewer, Microsoft XML) (Éditeur en mode « *tableau* » recommandé).**