Introduction à XML

Notes de cours

Auteur : Martin Sévigny, *AJLSM* [sevigny@ajlsm.com]

Date : 13 septembre 2002

Présentation

Ce document hypertexte contient des notes, des informations et des liens qui servent de support documentaire pour une formation *Introduction à XML* donnée par **Martin Sévigny** [mailto:sevigny@ajlsm.com] . Il s'agit donc d'un complément à une formation et non d'une formation complète ou un tutoriel.

Ces notes peuvent être reproduites, mais à condition qu'elles demeurent entières et que les mentions d'auteur et de date soient bien indiquées. De plus, il est préférable de toujours référer à l'URL **http://www.ajlsm.com/formation/xml/** [http://www.ajlsm.com/formation/xml/] pour obtenir la dernière version disponible.

A noter qu'une version **PDF** [notes.pdf] de ces notes est disponible.

Objectifs

Les objectifs de cette formation sont les suivants :

1) **Connaître l'origine et le futur de la norme XML**

   Cette norme ou recommandation n'est pas arrivée de nulle part en 1998. Bien comprendre son origine aide à bien comprendre ses différentes utilités. Par ailleurs, XML est voué à un brillant avenir, mais avant de s'engager dans un projet XML il est important de bien connaître les évolutions futures de la norme et des outils associés.

2) **Connaître les différentes applications de la norme XML, en particulier les applications documentaires**

   XML est tellement générique que l'on peut l'utiliser à toutes les sauces. Il est donc important de bien comprendre les différentes applications possibles de XML afin de choisir celles qui pourraient vous être utile.

3) **Situer XML par rapport aux autres technologies**

   Dans certains cas, XML est irremplaçable. Mais en général, d'autres technologies ou approches peuvent être utilisées. Comment choisir ? Quels seront les avantages ou inconvénients ?

4) **Connaître les principaux concepts reliés aux documents XML : jeu de caractères, éléments, attributs, etc.**

   Un jour ou l'autre, on peut se retrouver face à un document XML. Comment réagir ? Comment reconnaître ce qu'on cherche ?

5) **Se faire une idée des principaux types de logiciels XML**

   XML n'est pas un logiciel, mais il existe des logiciels XML. Lesquels ? Que font-ils ? Les démonstrations de logiciel ne font pas partie de ces notes, mais plusieurs sont montrés lors de la formation.

6) **Voir un exemple d'une application documentaire basée sur XML**

   On apprend souvent mieux par l'exemple, c'est ce que nous tenterons de faire ici. A noter que cet exemple n'est pas repris dans ces notes, seulement lors de la formation.

7) **Être en mesure d'évaluer, de façon générale, si XML peut répondre à ses propres besoins documentaires**

   Il s'agit de l'objectif le plus important, mais en même temps le plus difficile à atteindre. L'ensemble de la journée servira de matériel de base pour en arriver là.

Déroulement de la formation

Pendant la journée de formation

Cette formation ne sera pas pratique, c'est-à-dire que les participants ne seront pas face à un ordinateur et ne manipuleront pas d'outils informatiques. **D'ailleurs, un des principaux objectifs de la formation devrait être de comprendre pourquoi une première journée avec XML ne peut être pratique !**

Trois approches pédagogiques seront utilisées tout au long de cette journée :

1) **Présentations magistrales**

   Le but de ces présentations, aussi courtes que possible, est de bien faire comprendre l'approche théorique et philosophique derrière XML.

2) **Démonstrations de documents, de systèmes et de logiciels**

   Ces démonstrations servent à illustrer ce que l'on peut faire avec XML, et les outils qui nous aident à y arriver.

3) **Résolution de problèmes**

   Nous allons discuter de cas ou de problèmes apportés par les participants ou proposés par le formateur, voir si XML est une solution et si oui préciser cette solution.

Après la journée de formation

Les notes que vous consultez vous ont été remises sur support imprimé. De plus, elles seront **disponibles sur le Web** [http://www.ajlsm.com/formation/xml/] . Vous pourrez donc les consulter à votre guise pour revoir certaines parties ou en approfondir d'autres.

De plus, cette documentation contient un grand nombre de liens hypertextes vers le Web où vous pourrez en apprendre plus sur XML et les technologies complémentaires.

Présentation de la formation

Plan de formation

1) XML est partout, mais où est XML ?
2) XML, l'alphabet de l'informatique du XXè siècle
3) L'origine de XML
4) Domaines d'application et limites de la norme XML
5) Les documents XML (Unicode, en-tête, éléments, attributs, entités, DTD ou schémas)
6) Les avantages de XML
   1) Pérennité
   2) Echange de données
   3) Documents structurés
7) Une chaîne de traitement XML
   1) La saisie, la conversion ou la production
   2) Le stockage
   3) Le repérage
   4) La consultation
8) Exemples d'applications XML

Introduction à XML

Quelques mots sur XML

XML [http://www.w3.org/XML/] est une recommandation du **World Wide Consortium (W3C)** [http://www.w3.org] depuis février 1998 (**révision mineure** [http://www.w3.org/TR/2000/REC-xml-20001006] le 6 octobre 2000). Cette norme est simple, et c'est cette simplicité qui fait sa puissance.

XML est en fait un **langage** permettant de définir des **formats de documents** et de créer des documents respectant ces formats. Ainsi, et c'est important, **XML n'est pas un format de document** comme tel.

Avant d'en dire plus sur XML, nous allons d'abord réfléchir à une *application documentaire* vieille de plusieurs siècles, soit l'édition et l'impression de **livres**.

Sites et pages Web

XML étant une recommandation du W3C, le **site Web** [http://www.w3.org] de ce consortium constitue sûrement le meilleur point de départ pour en apprendre plus sur la norme mais surtout sur les normes associées ainsi que sur les différents projets impliquant XML. Parmi les normes associées, citons, en vrac:

- les espaces de noms (**Namespaces in XML** [http://www.w3.org/TR/1999/REC-xml-names-19990114/] , recommandation du W3C depuis le 14 janvier 1999)
- les **schémas** [http://www.w3.org/XML/Schema] pour faire suite aux DTD, recommandation du W3C depuis 2 mai 2001)
- **XML Pointer, XML Base and XML Linking** [http://www.w3.org/XML/Linking] pour la normalisation des fonctions hypertextes (recommandation du W3C depuis le 27 juin 2001)
- **Resource Description Framework** [http://www.w3.org/RDF/] pour la représentation des métadonnées (recommandation du W3C depuis le 22 février 1999)
- **CSS** [http://www.w3.org/Style/CSS/] (recommandation du W3C depuis décembre 1996, en version 2 depuis mai 1998) pour la mise en page des documents (CSS est supporté sur des documents XML dans Internet Explorer 5, 6 et Netscape 6, 7, de même que Mozilla 1.x). La version 3 de la norme est **en cours de préparation** [http://www.w3.org/Style/CSS/current-work] .
- **XSL** [http://www.w3.org/Style/XSL/] pour les transformations (avec **XSLT** [http://www.w3.org/TR/xslt.html]) et pour la mise en page sophistiquée. XSLT 1.0 est une recommandation du W3C depuis le 16 novembre 1999, la version 2.0 est en cours de réalisation. XSL est une recommandation du W3C depuis le 15 octobre 2001 seulement.
- **Synchronized Multimedia (SMIL)** [http://www.w3.org/AudioVideo/] pour la mise en place d'interfaces multimédias (la version 1.0 est une recommandation du W3C depuis le 15 juin 1998, la version 2.0 depuis le 7 août 2001)
- **SVG** [http://www.w3.org/Graphics/SVG/Overview.htm8] pour les ressources graphiques en mode vectoriel (recommandation du W3C depuis le 4 septembre 2001, la version 1.1 est en cours de réalisation).

Ces normes sont les briques de base des futurs systèmes d'information, et c'est pourquoi il peut être important de connaître leur existence et leur domaine d'application.

Il existe également d'autres site intéressants pour en savoir plus sur la norme, les outils ou les projets XML:

- Le site **XML Cover Pages** [http://xml.coverpages.org/] est probablement le plus important site sur XML (et SGML) sur le Web.
- Pour en savoir plus sur les différents logiciels XML disponibles, consulter le site **Free XML tools and software** [http://www.garshol.priv.no/download/xmltools/] .

Pour des ressources **en français** au sujet de XML, les sites suivants peuvent être intéressants:

- Le site **XMLFr** [http://xmlfr.org/] , avec beaucoup d'information et des listes de discussions, mais aussi une section sur la **documentation et les tutoriels** [http://xmlfr.org/documentations/]
- Le site de l'**ATICA** [http://www.atica.pm.gouv.fr] en discute aussi dans le contexte de l'administration française.
- Une **version française** [http://babel.alis.com/web_ml/xml/] de la norme XML

- **XML Francophone** [http://www.chez.com/xml/] , un autre site sur le sujet
- Informations sur la **plate-forme SDX** [http://sdx.culture.fr] du ministère de la culture et de la communication

Références bibliographiques

XML connaît une telle popularité qu'un nombre impressionnant de publications imprimées existent à ce sujet, y compris en français. La plupart des livres d'introduction se valent plus ou moins, et nous trouvons de plus en plus de livres, y compris en français, qui traitent de certains aspects particuliers de XML, surtout pour les développeurs.

Nous ne fournissons aucune référence précise, étant donné qu'aucun livre ne se distingue particulièrement.

Un système documentaire universel: le livre

Qu'est-ce qu'un livre?

Un livre est un réservoir d'information, information qui a donc été produite mais également consignée sur un support. Le contenu du livre est présenté en respectant un grand nombre de règles et de traditions. Ce sont ces règles qui font en sorte que l'on peut comprendre ce qu'on retrouve à l'intérieur du livre.

Mais le livre en lui-même n'est pas suffisant pour atteindre l'objectif de diffusion de connaissances ou de simple loisir. Une autre série de règles et de conventions, cette fois apparentées à la langue et au langage, sont absolument nécessaires.

Le codage de l'information d'un livre

Le contenu d'un livre est codé à plusieurs niveaux. Nous allons nous servir de ces niveaux de codage pour comprendre où se situe XML dans un système documentaire.

Qu'est-ce qui fait que l'on peut lire et surtout **comprendre** un livre?

Un support que l'on peut lire

Le livre est bien sûr imprimé sur du papier, que l'on peut consulter, porter, toucher, etc. Ce support, sans qui rien ne se produit, n'est pourtant pas la partie la plus importante.

Des symboles élémentaires

Les 26 lettres de l'alphabet, des symboles, des chiffres, de la ponctuation, des diacritiques, etc. Dans notre système d'écriture, ces symboles sont disposés les uns après les autres, de gauche à droite, de haut en bas, sur des lignes droites.

Des objets plus évolués

Des espaces et de la ponctuation qui délimitent des mots, des phrases, des paragraphes, des documents, etc.

La grammaire, le vocabulaire, l'orthographe

Les lettres, puis les mots, les phrases seront compréhensibles pour une personne qui connaît la langue dans laquelle le texte est écrit.

Une navigation aisée

Le support nous permet de passer facilement d'une page à l'autre, les index et les tables facilitent le repérage de l'information.

Une connaissance évoluée

L'ensemble des textes dans une langue, dans une discipline, compris par les personnes, constituent un bagage de connaissances qui peuvent être utilisées dans différentes circonstances.

En résumé

Un imprimeur, avec son savoir-faire, est en mesure d'imprimer un texte à parti de symboles (les lettres) et ce sur un support (le livre) que l'on pourra naviguer en tournant les pages. Pour faire ce travail, nul besoin de connaître la langue dans laquelle le texte est écrit, il doit juste connaître les règles du système d'écriture requis.

Introduction à XML

Les livres et XML

Maintenant que nous comprenons bien les différentes composantes d'un livre, nous allons nous attarder à comparer ces composantes à celles d'un système documentaire basé sur XML.

Livre	XML	Commentaires
Le livre (objet physique) en tant que support	Un fichier informatique	Les documents XML sont **indépendants des supports**. Ils peuvent n'exister que virtuellement. Toutefois, on peut bien entendu les stocker dans un ou des fichiers informatiques. Un document XML peut être stocké dans plusieurs fichiers.
Les caractères et autres symboles	Unicode	Le jeu de caractères des documents XML est **Unicode** [../docs/unicode.xml] (sauf indications contraires). **En principe, le problème du multilinguisme est réglé**.
Les mots, phrases, etc.	Les éléments, les attributs, les entités, les notations, les sections marquées	Ce sont ces composantes qui forment les briques de base des documents XML. Essentiellement, un document XML est une **hiérarchie** d'**éléments** qui sont imbriqués les uns dans la autres. Ces éléments peuvent avoir des **attributs** pour les qualifier.
Langue, grammaire	DTD ou Schéma	Les **DTD**, ou leur successeur les **schémas**, permettent d'exprimer les contraintes grammaticales des documents XML. **Il n'existe pas de DTD prédéfinie en XML.**
Navigation	Application	La norme XML ne sert qu'à structurer et stocker de l'information. **Les activités de navigation, de consultation, de recherche, etc., ne sont pas concernées par la norme**, mais plutôt par des applications qui utilisent des documents XML.
Connaissance	Applications, partage d'information	La création de bases de connaissances est possible voire facilitée avec la norme XML. Toutefois, nous sommes encore une fois à un niveau supérieur qu'il faut implanter à l'aide d'outils spécifiques.

Introduction à XML

Une première conclusion sur XML

Ce que l'on peut retenir de la comparaison précédente, c'est qu'XML définit en fait peu de choses:

- Un jeu de caractères pour représenter les symboles de base
- Des objets qui servent à structurer de l'information
- La possibilité de définir des grammaires pour restreindre l'utilisation des objets

C'est **justement parce que XML définit peu de choses qu'il offre autant de possibilités**. De la même façon on peut dire que toute la connaissance humaine et la richesse culturelle de la planète existent et se communiquent à l'aide de symboles et de différentes règles grammaticales que l'on peut, en général, convertir de l'une à l'autre.

C'est pourquoi il faut se méfier des phrases du genre *Ce logiciel supporte XML* ou encore *Ces documents sont en XML*, ou encore *Apprendre à utiliser XML*. Ca peut être intéressant, mais surtout ça ne veut rien dire en tant que tel, on a besoin de plus d'information pour mieux comprendre.

Origine de XML

La valse des ML

SGML, le père spirituel

Fondamentalement, XML apporte bien peu par rapport à SGML, le *Standard Generalized Markup Language*, ou **ISO 8879:1986**. En fait, on peut formellement dire que XML est un sous-ensemble de la norme ISO 8879. Qui dit sous-ensemble dit nécessairement *moins* de fonctionnalités, et c'est le cas avec XML.

Pourquoi faire tout un plat de XML alors que SGML est disponible depuis plus de 16 ans ? Difficile de trouver une réponse précise à cette question, mais on peut tout de même avancer ces quelques éléments :

- SGML est une norme ISO, et souvent les normes ISO sont considérées comme inaccessibles ou inintéressantes à l'ère Internet ;
- SGML a toujours été utilisé dans des systèmes documentaires très complexes, à très important volume d'information ;
- SGML est d'une souplesse telle que les utilisateurs sont favorisés mais pas les développeurs d'applications ;
- SGML impose certaines contraintes qui limitent sa facilité d'utilisation dans un réseau tel Internet ;
- SGML impose certaines contraintes qui diminuent les performances des systèmes documentaires.

D'autres facteurs ont limité l'utilisation et l'intérêt de SGML récemment, mais ces facteurs **se retrouvent aussi dans XML**.

Pour tous ceux qui s'intéressent à l'**histoire** de la conception et de la normalisation de XML, avec en plus des éléments d'information sur l'histoire de SGML et de ses acteurs, je conseille vivement de lire **The Annotated XML Specification** [http://www.xml.com/pub/a/axml/axmlintro.html] , de Tim Bray, l'un des pères de XML. Il faut tout particulièrement lire les annotations de type *historique ou culturelle*, dénotées par la lettre H.

HTML, le cousin chéri, mal aimé puis renié

On peut dire que la plus belle application documentaire réalisée avec SGML est le Web lui-même, car **HTML est défini de façon formelle comme une application SGML**. Cette affirmation n'a qu'une valeur théorique parce que dans la pratique, aucun outil HTML populaire, des navigateurs aux éditeurs, n'a considéré HTML comme une application SGML.

HTML a permis la mise en place rapide du Web, le plus important système d'information jamais conçu. Grâce à sa simplicité et sa *puissance*, de nombreuses personnes ont pu concevoir des sites Web simples ou complexes.

Fondamentalement, **HTML est une mauvaise utilisation de SGML**. En effet, plusieurs types de structuration sont possibles, et les logiciels et les auteurs ont presque tous choisi les pires formes.

Aujourd'hui, on constate les limites de HTML, fondamentalement parce qu'on réalise qu'il est quasi-

ment impossible de réutiliser de l'information codée en HTML, mais aussi parce qu'on réalise qu'il peut être très utile de réutiliser de l'information électronique!

SGML aurait pu solutionner les problèmes de HTML, mais on a décidé qu'il fallait une norme *à la mode Internet*, et c'est ainsi que XML est né, de la tête et de l'esprit de la plupart des spécialistes SGML dans le monde. Avant que les grandes compagnies informatiques ne s'en mêlent.

De l'ISO au W3C

La normalisation de XML s'est faite dans le cadre du W3C, mais à une époque (1997 essentiellement) ou le W3C était plus petit qu'aujourd'hui et surtout moins tiraillé par la compétition entre les membres. Mais il ne faut pas oublier que pour participer aux discussions et au processus de normalisation du W3C, **il faut en faire partie**.

Aujourd'hui, la normalisation au sein du W3C est difficile. En effet, il s'agit d'une structure suffisamment souple pour que de nombreux projets de normalisation soient lancés, mais en même temps ces projets arrivent difficilement à terme, surtout lorsqu'il s'agit de normes importantes pour un **petit nombre de sociétés qui contrôlent l'essentiel d'un marché**, comme par exemple pour les systèmes de gestion de bases de données relationnelles.

XML est donc normalisé et c'est terminé, mais de nombreuses normes satellites à XML ne sont pas terminées, et les retards s'accumulent.

L'avenir de XML

Lors de la création de XML, on mentionnait régulièrement qu'**il ne devrait jamais y avoir de version subséquente de XML**. Pourquoi ? La réponse devrait être évidente à la fin de la journée de formation. Toutefois, il y a eu une révision, des tentatives de simplification, et depuis peu il existe un brouillon de **XML 1.1** [http://www.w3.org/TR/xml11/] , voire d'une version 2.0.

Il ne faut pas voir dans ces évolutions une quelconque faiblesse dans la norme elle-même, mais plutôt un réel désir de la simplifier encore plus pour rendre son utilisation plus universelle. Ces efforts sont appuyés en particulier par la communauté du commerce électronique, parfois au détriment des applications documentaires. Affaire à suivre.

Domaines d'application de XML

L'universalité de XML

XML permet de définir des formats de documents. Par conséquent, si vous avez de l'information numérique (texte, image, vidéo, son) à stocker ou gérer, XML peut être une solution intéressante. Autrement dit, **XML s'applique à toutes les problématiques documentaires**.

Mais est-ce que XML est une solution intéressante pour toutes les situations ? Surtout pas. Nous pouvons regrouper deux types d'application où XML est un format définitivement intéressant.

Echange de données

Pour différentes raisons, XML s'impose aujourd'hui comme le format d'échange de données par excellence.

L'échange de données consiste à faire communiquer de l'information entre différents systèmes plus ou moins hétérogènes. Par exemple deux bases de données gérées par deux outils différents.

Les transactions de commerce électronique sont un exemple de ce type d'application, et XML joue effectivement un rôle important.

Gestion documentaire hypermédia

Pensez à la façon que l'on construit des documents aujourd'hui, par exemple avec un traitement de texte. Ces documents sont hypermédias, car ils contiennent du texte, de la structure, des objets mul-

timédias comme des images, et des liens entre ces éléments et d'autres documents.

Les pages d'un site Web sont aussi des documents hypermédias. Un site Web, à la rigueur, peut être vu comme un document hypermédia.

Sans SGML ou XML, il est très difficile de gérer efficacement des documents hypermédias, gérer au sens où on l'entend habituellement dans le monde des bases de documents.

Mais avec l'une ou l'autre de ces normes, il est possible de le faire, et il s'agit probablement du domaine d'application de XML le plus important dans le monde documentaire.

Introduction

Dans cette section, nous allons présenter les composantes de base des documents XML. Pour bien suivre, il est conseillé de consulter en parallèle des exemples de documents.

Les descriptions que l'on retrouve dans ces sections ne sont pas formelles, elles visent plutôt à simplifier les concepts. Pour connaître tous les détails, il est préférable de **consulter la norme** [http://www.w3.org/TR/2000/REC-xml-20001006] .

Toutes les composantes des documents XML ne seront pas abordées ici. Nous allons plutôt discuter de celles que l'on rencontre le plus souvent.

Dans l'ordre, nous allons voir le **jeu de caractères Unicode**, **le prologue**, les **éléments**, les **attributs** et enfin les **entités**.

Nous terminerons pas une discussion sur les **DTD et schémas**, qui ne font pas partie des documents comme tel mais qui sont très importantes.

Remarques générales

A noter que la **casse** des caractères dans un document XML est importante, alors que de façon générale les **espaces, tabulations et retours de chariots** ne le sont pas.

De plus, il est possible d'insérer des commentaires dans les documents XML, en utilisant la syntaxe suivante:

```
<!-- Voici le commentaire -->
```


Unicode

Unicode est un standard définissant un jeu de caractères. Sans entrer dans les détails, mentionnons que l'Unicode de base permet le codage des caractères sur 2 octets, soit plus de 65 000 caractères différents. En plus des techniques de codage, Unicode définit un jeu de caractères qui en compte plusieurs dizaines de milliers (la version actuelle d'Unicode est la 3.2 et elle définit plus de 95 000 caractères). Tous les caractères des langues vivantes s'y retrouvent, de même que certaines langues mortes et des symboles.

Unicode ne règle pas tous les problèmes de multilinguisme, mais il constitue la meilleure solution que l'on connaisse actuellement.

Pour en savoir plus sur Unicode, voici quelques sites à consulter:

- Le site Web du **consortium Unicode** [http://www.unicode.org/] , où vous trouverez les tableaux de caractères et beaucoup d'informations techniques sur la norme.
- Un site Web en français, **Écritures du monde** [http://www.culture.gouv.fr/edm/] , réalisé par Michel Bottin et ses collaborateurs au ministère de la culture et de la communication. Ce site présente les différents systèmes d'écriture dans le monde et constitue un bel exemple de l'utilisation

d'Unicode sur le Web.

- Le site **Alan Wood's Unicode Resources** [http://www.alanwood.net/unicode/] , en anglais, mais qui constitue un inépuisable réservoir d'information à propos d'Unicode. Sont particulièrement utiles les exemples de caractères, les informations sur le support logiciel et sur les polices.
- Un endroit pour **télécharger la police Bitstream CybertBit** [ftp://ftp.netscape.com/pub/communicator/extras/fonts/windows/] , qui comporte plusieurs milliers de caractères. Les verions 2000 et XP de la suite Office de Microsoft incluent également la police Arial Unicode MS, 50 377 caractères, soit tout Unicode 2.1.
- Pour Windows, une **extension au gestionnaire de fichiers** [http://www.microsoft.com/typography/property/property.htm] pour obtenir des informations détaillées sur des polices de caractères.

Unicode et XML

Par défaut, les documents XML sont des documents *texte* dont le jeu de caractères est l'Unicode. En quelque sorte, il y a un seul jeu de caractères standard pour l'XML, et ce jeu de caractères est le plus important que l'on connaisse. D'autres jeux de caractères (comme l'ISO Latin 1) peuvent être utilisés, mais une application XML n'est pas obligée de les supporter, et donc de les traiter correctement.

Cette intime association entre Unicode et XML fait en sorte que les systèmes d'information basés sur XML peuvent gérer (en théorie) des documents ou des informations de toutes les langues, y compris dans les mêmes documents.

La question de l'encodage

Il existe plusieurs façons d'encoder des caractères Unicode dans des fichiers informatiques, on parlera de différents *encodages*.

L'encodage nommé **UTF-8** consiste à encoder les caractères ASCII sur un octet, puis les autres caractères sur 2, 3, ou 4 octets, sans ambiguïté. Il s'agit d'un encodage intéressant pour le français, car la plupart des caractèrés pourront s'écrire sur un seul octet (les lettres non accentuées), alors que les autres sont définis de telle sorte qu'en UTF-8 ils sont codés sur deux octets seulement.

L'encodage nommé **UTF-16** consiste à représenter tous les caractères sur deux octets. Cet encodage est plus efficace pour des documents en langues asiatiques par exemple, mais inefficace pour des documents en langues européennes.

Les entités caractères en XML

Dans un fichier XML, les caractères peuvent être directement encodés en Unicode, et normalement les outils le font automatiquement. Toutefois, il est possible d'inscrire tout caractère Unicode à l'aide des entités caractères, qui prennent la forme suivante :

```
&#x4e35;
```

Dans ce cas, il s'agit du caractère qui occupe la position 4E35 en base hexadécimale (20 021 en base décimale) dans le jeu de caractères Unicode. Il s'agit de ce caractère : . Par ailleurs, on peut aussi utiliser cette technique :

```
&#20021;
```

Il s'agit exactement du même caractère, mais cette fois indiqué en base décimale. C'est le 'x' qui suit le '#' qui indique la base hexadécimale.

Les limites d'Unicode

Le standard Unicode est intéressant, mais le problème réside dans le support Unicode que l'on

trouve dans les systèmes d'exploitation, logiciels et polices.

Au niveau des systèmes d'exploitation, les versions récentes de Windows supportent Unicode, de même que plusieurs logiciels récents, dont la suite Office. Sur MacOS, Unicode est supporté depuis la version 8.5 et bien sûr en OS-X. Les systèmes UNIX/Linux récents supportent en général très bien Unicode.

Le prologue

Le prologue des documents XML joue trois rôles importants:

1) Préciser qu'il s'agit d'un document XML
2) Identifier le jeu de caractères utilisé
3) Identifier la grammaire (DTD) utilisée

Les trois éléments sont facultatifs, mais il est en général préférable d'include la déclaration XML qui contient les deux premiers éléments d'information.

Si aucun jeu de caractères n'est spécifié, une application XML doit supposer qu'il s'agit du jeu de caractères Unicode, encodé en UTF-8 ou en UTF-16.

Si aucune DTD n'est identifiée, le document est considéré *bien formé*. Si une DTD est spécifiée, le document est alors *valide* (s'il respecte les règles dictées dans la DTD bien sûr).

Syntaxe

La syntaxe des différents éléments est la suivante.

L'identification XML

L'identification du document XML est nécessairement au début du document et a la forme suivante:

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

L'encodage

L'encodage, s'il est spécifié, se retrouve dans l'instruction de traitement qui identifie le document XML:

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

Dans cet exemple, il s'agit d'un document XML dont le jeu de caractères est l'ISO-Latin 1 (norme ISO 8859-1), et donc pas l'Unicode. Dans l'exemple précédent, le jeu de caractères était l'Unicode, avec encodage UTF-8 ou UTF-16, soit la valeur par défaut.

La DTD

La DTD peut être référencée de la façon suivante:

```
<!DOCTYPE racine PUBLIC "identificateur public" "uri de la DTD">
```

Ici, "racine" identifie l'élément qui contient tous les autres dans le document, la mention PUBLIC et la chaîne de caractères qui suit, optionnelles, précise un identificateur public pour cette DTD, et enfin la chaîne de caractères qui la suit, obligatoire, précise la localisation de la DTD à l'aide d'une URI.

Très souvent, un appel à une DTD sera de la forme suivante:

```
<!DOCTYPE racine SYSTEM "racine.dtd">
```

ce qui signifie que la DTD est dans un fichier nommé "racine.dtd" et qui se situe dans le même répertoire que le document.

Soulignons également qu'il est possible d'inclure la DTD à l'intérieur du document lui-même, de la façon suivante:

```
<!DOCTYPE racine [
  <!ELEMENT racine EMPTY>
]>
```

Dans cet exemple, la DTD (fort simple) est incluse dans le document. il s'agit d'une pratique assez rare, mais cette technique est aussi utilisée pour déclarer des entités.

Le schéma

Les schémas ne sont pas références dans l'en-tête du document, mais plutôt en attribut d'un élément, comme dans cet exemple :

```
<BookCatalogue
     xmlns="http://www.publishing.org"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://www.publishing.org BookCatalogue.xsd">
  <Book>...</Book>
  <Book>...</Book>
</BookCatalogue>
```

Dans cet exemple, on identifie l'espace de nom du schéma et sa localisation.

Les éléments

Les éléments sont les objets les plus importants des documents XML. En effet, fondamentalement, les documents XML sont des hiérarchies strictes d'éléments. Ainsi, il existe toujours un (et un seul) élément supérieur qui contient tous les autres. De plus, un élément peut contenir d'autres éléménts ou du texte (un type particulier d'élément), et se situe à l'intérieur d'un seul élément.

Voici un document XML bien formé et complet:

```
<?xml version="1.0"?>
<livre>
  <titre>Introduction à XML</titre>
  <édition>Deuxième édition</édition>
</livre>
```

Trois éléments composent ce document:

1) Un élément supérieur **livre**
2) Un élément **titre**
3) Un élément **édition**

Les éléments se reconnaissent par l'utilisation d'une syntaxe particulière, soit une balise d'ouverture constituée du caractère "<" suivi du nom de l'élément suivi du caractère">", et enfin d'une balise de fermeture qui reprend la même syntaxe mais cette fois en ajoutant "/" entre le "<" et le nom de l'élément.

On remarque donc que dans l'exemple précédent l'élément **livre** contient deux sous-éléments, alors

que les éléments **titre** et **édition** ne contiennent que du texte. Cet exemple illustre également que les caractères accentués peuvent être utilisés sans problèmes dans les noms d'éléments.

Les éléments ne peuvent pas se chevaucher. Par exemple, ceci est interdit dans un document XML:

```
<b>gras<i>italique</b></i>
```

C'est interdit parce que l'élément **i** n'est pas complètement à l'intérieur de l'élément **b**. Il ne s'agit donc pas d'une véritable hiérarchie.

Les éléments vides, qui ne possèdent si nous-élément ni texte, peuvent être représentés ainsi:

```
<img/>
```

Il y a une seule balise, qui ouvre et ferme en même temps l'élément, et cette balise respecte une syntaxe particlière, inspirée à la fois des balises d'ouverture et des balises de fermeture.

Les attributs

Les attributs sont toujours associés aux éléments. Ils viennent en quelque sorte les qualifier.

Un attribut est toujours une paire *nom=valeur*. Un élément ne peut pas avoir deux attributs de même nom. Un attribut doit toujours avoir une valeur, même si celle-ci est une chaîne vide.

Les attributs sont toujours spécifiés **dans la balise d'ouverture** de l'élément. Voici quelques exemple:

```
<chapitre id="ch1" titre='Introduction à XML'></chapitre>
```

Nous avons ici un élément **chapitre** qui contient deux attributs, un premier nommé **id** dont la valeur est **ch1** et un deuxième nommé **titre** dont la valeur est **Introduction à XML**. La syntaxe est simple et intuitive: le caractère "=" séparer le nom et la valeur, les guillements (simples ou doubles) délimitent la valeur, et des espaces séparent les attributs entre eux.

Les entités

Les entités sont des substituts pour des séquences d'information. Ces entités doivent être définies dans l'en-tête du document XML, ou dans la DTD, et peuvent être référencées à une ou plusieurs reprises dans le document.

Il existe plusieurs types d'entités, nous verrons les plus simples ici.

Entité XML définie dans le document

Une telle entité sert en général d'abbréviation pour des informations répétitives ou qui peuvent être modifiées en bloc:

```
<?xml version="1.0"?>
<!DOCTYPE démo [
  <!ELEMENT démo (#PCDATA|important)>
  <!ELEMENT important (#PCDATA)>
  <!ENTITY cie "Les aliments de Bordeaux SA">
  <!ENTITY imp "<important>Attention!</important>">
]>
<démo>
  &cie;
```

```
  &imp;
</démo>
```

Ce document déclare les entités **cie** et **imp**, et les utilise. Pour ce faire, on utilise une balise formée du caractère "&", suivi du nom de l'entité, suivi du caractère ";".

Le document précédent est équivalent au suivant:

```
<?xml version="1.0"?>
<!DOCTYPE démo [
  <!ELEMENT démo (#PCDATA|important)>
  <!ELEMENT important (#PCDATA)>
  <!ENTITY cie "Les aliments de Bordeaux SA">
  <!ENTITY imp "<important>Attention!</important>">
]>
<démo>
  Les aliments de Bordeaux SA
  <important>Attention!</important>
</démo>
```

Un parseur XML fournira à une application exactement la même information dans les deux cas.

Entité XML définie à l'externe

Les entités définies à l'extérieur des documents sont utilisées de la même façon mais sont déclarées ainsi :

```
<?xml version="1.0"?>
<!DOCTYPE doc [
  <!ENTITY chap1 SYSTEM "../chapitres/chap1.xml">
  <!ENTITY chap2 SYSTEM "../chapitres/chap2.xml">
]>
<doc>
  &chap1;
  &chap2;
</doc>
```

Dans cet exemple, on suppose que les fichiers "chap1.xml" et "chap2.xml" contiennent les chapitres marqués en XML.

Entité non XML

Les entités non XML peremettent de déclarer des documents (ou plus généralement de l'information) qui n'est pas en format XML, par exemple des composantes multimédia (images, son, vidéo, etc.). Ils nécessitent l'utilisation des notations, ce que nous ne verrons pas ici.

Il est important de noter que les entités non XML ne sont pas nécessaires pour inclure des composantes multimédias dans les documents XML.

Entités prédéfinies

Il existe 5 entités prédéfinies en XML, les voici avec leur signification:

```
lt: caractère '<'
gt:      caractère '>'
amp:     caractère 'é'
quot:    caractère '"' (guillement double)
apos:    caractère "'" (quillement simple ou apostrophe)
```

On comprend que ces entités ont été définies afin de permettre l'utilisation de ces caractères sans qu'ils puissent être confondus avec les caractères spéciaux utilisés par les documents XML pour le balisage.

Les DTD et les schémas

Les DTD, ou Document Type Definition, contiennent les règles que doivent respecter les éléments et attributs d'un document XML pour que celui-ci soit valide. De la même façon qu'on ne conçoit pas une base de données sans avoir déterminé sa structure, on ne devrait pas concevoir un système d'information basé sur XML sans avoir conçu la ou les DTD.

Comme son nom l'indique, on utilisera une DTD par type de documents que l'on gère. Il s'agit d'une notion très floue mais qui peut servir de règle générale.

Il existe un grand nombre de DTD disponibles publiquement, soit pour des applications spécifiques, soit pour de grandes catégories de documents. Quelques sites essaient d'en faire le recensement, dont le **XML.ORG Registry** [http://www.xml.org/xml/registry.jsp] .

Nous passerons sous silence les multiples projets de spécifications XML dans le domaine du commerce électronique et de l'échange de données qui aboutissent à autant de DTD publiques.

Nous allons simplement mentionner les DTD **TEI** [http://www.tei-c.org/] et **Docbook** [http://www.oasis-open.org/docbook/] pour des documents comme des livres, **BiblioML** [http://www.culture.gouv.fr/BiblioML/] pour les références bibliographiques, et quelques-une des DTD du ministère de la culture comme la DTD pour l'inventaire, les dossiers d'artistes, les bilans scientifique régionaux en archéologie, etc.

Composantes d'une DTD

Une DTD a pour objectif de spécifier quatre types d'information:

1) Quels sont les éléments permis dans les documents
2) Quel contenu peuvent posséder les éléments
3) Quels attributs peuvent être associés à quels éléments
4) Quelles sont les valeurs permises pour les attributs

Les deux premiers types d'information s'effectuent à l'aide des déclarations d'éléments, alors que les deux autres s'effectuent à l'aide des déclarations d'attributs.

Les déclarations d'éléments

Les éléments sont déclarés de cette façon dans une DTD:

```
<!ELEMENT chapitre (titre?, (para|liste)*, section+)>
```

Plusieurs informations sont fournies sur une telle ligne. D'abord, on déclare le nom de l'élément, soit **`chapitre`** dans notre exemple. Ensuite, on détermine quel est son modèle de contenu, entre parenthèses.

Ces modèles de contenu peuvent être des éléments (déclarés par ailleurs) ou encore des éléments et du texte (on les appellera des modèles de contenu mixtes). Pour comprendre l'exemple précédent, il faut connaître la signifiation de ces symboles:

,

Indique une séquence, c'est-à-dire que l'élément à gauche de la virgule doit précéder l'élément à droite.

|

Indique un choix, c'est-à-dire que l'on choisit soit l'élément à gauche de la barre verticale, soit l'élément à droite.

?

Indique un élément optionnel et qui ne peut pas être répété. Autrement dit, il a une cardinalité de 0 ou 1.

*****

Indique un élément qui est optionnel mais qui peut être répété. Autrement dit, il a une cardinalité de 0 ou plusieurs, sans limite.

+

Indique un élément obligatoire et qui peut être répété. Il a donc une cardinalité de 1 ou plusieurs, sans limite.

Un élément peut également contenir du texte. Dans un tel cas, son modèle de contenu doit nécessairement être de la sorte:

```
<!ELEMENT para (#PCDATA|ville|etat|personne)*>
```

Ainsi, le mot "#PCDATA" doit être au début, les différents éléments doivent être séparés par des "ou", et l'ensemble doit être optionnel et répétable. La signification d'un tel modèle de contenu est la suivante:

L'élément **para** peut contenir du texte ou des éléments **ville**, **etat** ou **personne** dans n'importe quel ordre, autant d'occurrences que nécessaires, mais sans obligation de les utiliser.

Ces possibilités de structuration peuvent paraître limitées, et elles le sont. Entre autres, il n'est pas possible de contrôler le contenu textuel des champs, par exemple pour limiter à une liste de termes ou pour un format de date. Pour y arriver, il faut agir au niveau des applications XML ou utiliser des schémas, pas encore normalisés.

Il est toutefois important de souligner que ces possibilités sont suffisantes pour les documents textuels ou hypermédias.

Les déclarations d'attributs

Les DTD permettent de décler des attributs et leurs valeurs, et de les associer aux éléments. Voici un exemple de déclaration d'attributs:

```
<!ATTLIST chapitre
     id          ID        #REQUIRED
     niveau      NMTOKEN   #IMPLIED
     etiquette   CDATA     #FIXED "chapitre"
     acces       (public|restreint) "public"    >
```

Cette déclaration peut paraître complexe, mais il faut la regarder composante par composante. Tout d'abord, on indique que l'on déclare les attributs associés à l'élément **chapitre**. Ensuite, on retrouve quatre attributs déclarés, un par ligne dans notre exemple.

Le premier attribut se nomme **id**, son type de données est ID, c'est-à-dire qu'il doit répondre à certains critères (essentiellement composé de lettres et de chiffres, doit débuter par une lettre) et surtout que sa valeur doit être unique parmi tous les attributs de type ID dans le document, et enfin cet attribut est obligatoire, c'est-à-dire qu'il doit se retrouver dans toutes les balises d'ouverture des éléments **chapitre**.

Ensuite, l'attribut **niveau** est un nom, semblable à un ID mais pas nécessairement unique, il n'est pas obligatoire et il n'y a pas d'autres restrictions sur son contenu.

L'attribut **etiquette** est de type CDATA, c'est-à-dire qu'il peut contenir du texte quelconque, il a toujours la même valeur fixe, soit **chapitre**.

Enfin, l'attribut **acces** peut posséder deux valeurs, soit **public** ou **restreint**, et par défaut il a la valeur **public**.

Les schémas XML

Pour pallier aux limites des DTD, en particulier dans le domaine de l'échange de données, le W3C a entrepris un effort pour normaliser les **schémas XML** [http://www.w3.org/XML/Schema] , soit une nouvelle forme de grammaires et des types de données que l'on peut utiliser pour valider les documents XML. Les schémas XML jouent donc exactement le même rôle que les DTD dans une application XML, mais avec des possibilités plus grandes de structuration.

Ces possibilités sont réelles et peuvent s'avérer très utiles dans certains contextes, y compris des applications documentaires. Toutefois, pour une première journée de formation à XML, il n'est pas nécessaire d'entrer dans ces détails.

Outils disponibles

Les DTD sont stockées dans des fichiers texte, alors n'importe quel éditeur de texte peut permettre de les créer. Toutefois, certains outils peuvent faciliter la création ou la visualisation des DTD.

XML Spy [http://www.xmlspy.com/]

XML Spy est un éditeur XML mais qui comporte un mode spécial pour l'édition de DTD, avec des fonctions spécialisées et efficaces. Son mode d'affichage est de type tableau.

TurboXML [http://www.tibco.com/solutions/products/extensibility/turbo_xml.jsp]

Ce logiciel est le plus complet en ce qui concerne l'édition de DTD ou de schémas. D'ailleurs, on peut passer de l'un à l'autre. Il offre un mode graphique pour la visualisation des DTD.

DTDParse [http://nwalsh.com/perl/dtdparse/]

Il s'agit d'un programme Perl qui analyse une DTD et construit une version HTML facile à naviguer et à comprendre.

L'approche XML

Une philosophie XML ?

Quel est l'intérêt de la norme XML? En fait, il y en a plusieurs, et cette section va les résumer. Mais pour bien les comprendre, il faut savoir qu'XML est une norme permettant de définir des **formats de documents**, et que ces formats possèdent les caractéristiques suivantes:

- Ce sont des documents textuels utilisant la norme Unicode
- Le modèle de base est hiérarchique
- XML est une norme de l'industrie
- L'information dans les documents est auto-descriptive

L'ensemble de ces caractéristiques fait en sorte que la norme XML permet de représenter des **documents structurés**, elle assure la **pérennité de l'information** et facilite les **échanges de données**.

L'approche XML

La pérennité de l'information

Les documents XML sont de bons candidats pour la conservation à long terme, et ce pour les raisons suivantes:

- Le format est normalisé
- Les documents sont indépendants des plates-formes ou systèmes ou logiciels
- Les informations sont autodécrites
- Les documents peuvent être lisibles à l'oeil

Ce dernier point est assez important. En effet, pour consulter et comprendre un document XML, il suffit d'avoir en sa possession un ordinateur et un simple éditeur de texte. Ce ne sont pas des documents binaires ou à codage complexe, ce sont des documents qui utilisent des balises compréhensibles par les humains, pour autant que l'on connaissance la langue des balises et le sens des mots utilisés.

Le format XML est donc la pierre angulaire de l'archivage des données électroniques, et le gouvernement français l'a **récemment identifié** [http://www.atica.pm.gouv.fr/servicesenligne/guide/guide_sommaire.shtml] formellement comme tel.

L'approche XML

L'échange de données

Les documents XML sont complètement indépendants des plates-formes, des logiciels, des systèmes d'exploitation, etc. De plus, leur nature hiérarchique permet de représenter des structures de données fort complexes. Ces deux raisons font en sorte que cette norme est utilisée comme le principal format d'échange de données entre systèmes hétérogènes, par exemple des bases de données.

Nous n'entrerons pas dans ces détails qui débordent largement le contexte documentaire. Mais soulignons toutefois que la plupart des applications XML aujourd'hui entrent dans cette catégorie, c'est-à-dire qu'elles sont prévues pour faciliter les échanges d'information.

L'approche XML

Les documents structurés

Les documents XML, par leur nature hiérarchique mais aussi par la normalisation et le mécanisme de validation, peuvent constituer de très bons documents structurés.

Qu'est-ce qu'un document structuré?

Les documents structurés sont des documents qui contiennent de l'information à propos de leur **structure logique, sémantique et intellectuelle**. Pour mieux comprendre, voici un exemple d'un document non structuré suivi du même document mais cette fois-ci structuré:

```
<?xml version="1.0"?>
<document>
  <p><font size="20pt"><b>Introduction</b></font></p>
  <p><i>George Washington</i> n'a jamais gouverné le
        <b>Washington</b> mais a résidé à
        <b>Washington</b>.</p>
</document>
```

```
<?xml version="1.0"?>
<document>
  <section>
    <titre>Introduction</titre>
    <p><personne>George Washington</personne> n'a jamais
          gouverné le <etat>Washington</etat> mais a résidé à
          <ville>Washington</ville>.</p>
  </section>
</document>
```

La différence entre ces documents est très grande. Dans le premier cas, il s'agit d'un document formaté pour une application particulière. Dans le deuxième cas, il s'agit d'une structure d'information, et c'est tout.

Supposons que nous bâtissions deux collections de documents semblables au précédent. Si nous créons les documents selon le premier modèle, nous pourrions obtenir un système d'information intéressant, mais limité. Pour voir ces limites, essayons de voir ce que l'on peut faire avec la deuxième forme:

- On peut afficher les documents en plusieurs formats
- On peut créer un index des personnes, des états, des villes
- On peut construire une table des matières automatiquement

Ces opérations sont permises parce que le deuxième document, contrairement au premier, ne contient pas d'informations sur les traitements, mais bien sur **la nature des informations**. Et il s'agit de l'intérêt principal des documents structurés.

Pourquoi XML pour les documents structurés?

La norme XML est appropriée pour définir des documents structurés pour deux raisons principales:

1) Le modèle de données est hiérarchique, ce qui est également le cas des documents textuels, hypermédias, et de bon nombre de structures de données.
2) Les documents XML peuvent être validés selon une grammaire, permettant ainsi de définir, et par le fait même limiter, les structures possibles et donc les traitements à effectuer.

En fait, il n'existe pas d'autres formats appropriés pour les documents structurés, hormis SGML qui est un surensemble d'XML. Dans le monde documentaire, les documents structurés occupent encore une petite place, mais cette place devient de plus en plus grande.

Aperçu d'une chaîne de traitement

Il est possible de bâtir un système d'information autour de la norme XML. Celui-ci comprendra les fonctions habituelles des systèmes d'information, à savoir la **saisie**, le **stockage**, le **repérage** et la **consultation**.

Pour ce faire, il est nécessaire d'utiliser une suite logicielle complète ou un ensemble d'outils. Dans cette section, nous allons reprendre tous les aspects et discuter des points particuliers à XML et des outils disponibles.

La saisie

La saisie ou la **production** des documents XML peut se faire de plusieurs façons. Nous allons distinguer la productrion par conversion de la productrion par saisie ou édition.

La production par conversion

Est-il possible de convertir automatiquement des documents ou des bases de données en format XML?

La réponse est bien sûr positive, mais à certaines conditions. En effet, il faut réaliser qu'un traitement par ordinateur peut difficilement ajouter de l'intelligence à de l'information. Par exemple, un ordinateur ne pourra pas, de façon générale, convertir des caractères gras en noms de personnes.

C'est pourquoi il est très difficile de passer d'une base de documents de traitement de texte à une base de documents structurés.

Par contre, le passage d'une base de données (relationnelle ou textuelle) vers un format **XML**

http://www.w3.org] ne pose en général aucun problème, il existe même des outils qui permettront d'introduire un peu de hiérarchie (par regroupement) dans les structures XML équivalentes aux structures de bases de données.

La plupart des fournisseurs de systèmes de gestion de bases de données relationnelles offrent maintenant des produits pour importer et exporter des données en format XML. La conversion est donc automatique, mais le gain n'est pas vraiment important.

La production par saisie

La production par saisie implique des personnes et la plupart du temps des logiciels adaptés. Il est possible, mais non recommandé, de créer des documents XML à l'aide d'un simple éditeur de texte.

Mais l'outil par excellence pour saisir des documents XML est l'*éditeur XML*. Il peut y en avoir plusieurs types, mais tous possèdent ces caractéristiques de base:

- Ils vont créer des documents bien formés (pas d'erreur de syntaxe)
- S'il y a DTD, ils vont créer des documents valides
- S'il y a DTD, il vont faciliter la saisie en ne proposant que les objets valides dans le contexte d'édition

Au-delà de ces caractéristiques communes, les éditeurs XML peuvent se regrouper en deux grandes catégories, qui sont présentées ici brièvement. On peut en obtenir une lsite exhaustive sur le site de **XML Software** [http://www.xmlsoftware.com/editors.html] .

Les éditeurs orientés tableaux ou données

Ce sont les plus communs, ils présentent les éléments sous la forme de tableaux ou d'arbres. Ces éditeurs sont particulièrement destinés à la saisie de données et non de documents.

Il existe trop d"éditeurs orientés données pour tous les lister ici, mais mentionnons **XML Spy** [http://www.xmlspy.com] de Altova, utilisé dans le cadre de cette formation.

Les éditeurs orientés documents

Ces éditeurs sont en fait des traitements de documents XML, par analogie avec les traitements de texte. Ils sont moins nombreux et plus dispendieux, mais ils peuvent être d'une très grande utilité. Ils possèdent tous la caractéristique d'être très configurable, permettant ainsi de définir des environnements de saisie très efficaces.

Dans cette catégorie, on retrouve principalement **WordPerfect** [http://www.corel.com] de Corel, **XMetaL** [http://www.softquad.com] de Softquad, **Epic** [http://www.arbortext.com/Products/Epic/epic.html] de Arbortext, **Documentor** [http://www.excosoft.se/] de Excosoft.

Depuis le début septembre 2001, un nouvel éditeur entre dans cette catégorie, soit **XML Spy** [http://www.xmlspy.com] de Altova qui a aussi un module d'édition de documents. Cet outil s'affirme donc comme un acteur important dans le monde de l'édition de documents XML de toute nature.

Le stockage des documents

Les documents XML peuvent être stockés dans des fichiers, alors la problématique du stockage peut se résoudre simplement. Toutefois, il est peu pratique pour un système d'information de se fier au système de fichiers pour le stockage.

Les documents XML peuvent être stockés facilement dans des champs de bases de données relationnelles. Mais il existe également des bases de données XML, qui stockent et exploitent les documents XML de façon native.

Dans ce domaine, soulignons **Tamino** [http://www.softwareag.com/taminoplatform/] de Software AG, **XHive** [http://www.xhive.com/] de XHive Corporation ou encore **XIS** [http://www.exceloncorp.com/] de Excelon Corp. Ce sont toutefois des produits très chers et plutôt orientés vers le commerce électro-

nique. Il existe maintenant des solutions libres : **Xindice** [http://xml.apache.org/xindice/] du projet Apache et **Exist** [http://exist.sourceforge.net/] .

Les bases de données orientées objet constituent de bons candidats pour stocker des documents XML, qui peuvent être transposés facilement (et de façon normalisée, avec le Document Object Model) en objets. Les logiciels libres **Ozone** [http://www.ozone-db.org/] et **Zope** [http://www.zope.org] peuvent le faire assez bien.

Enfin, certaines bases de données relationnelles, en particulier **Oracle** [http://www.oracle.com] , commencent à pouvoir stocker les documents XML de façon intéressante, en exploitant leur structure.

Le repérage

Le repérage des documents XML constitue une voie de recherche encore très active. En effet, il existe des solutions, mais également des problèmes et surtout des opportunités dans ce domaine.

Un document XML est une base de données en soi, alors il est normal de vouloir exploiter la structure de l'information lors du repérage. Par exemple, sortir tous les noms de personne d'une collection de documents.

Dans un contexte orienté données, il existe plusieurs solutions basées sur le langage **XPath** [http://www.w3.org/TR/xpath] ou encore le langage de requêtes **XQL** [http://www.ibiblio.org/xql/xql-proposal.html] . Par ailleurs, il existe plusieurs propositions au W3C pour normaliser les langages de requêtes, recensées dans la section **XML Query** [http://www.w3.org/XML/Query] , et en particulier le futur standard **XQuery** [http://www.w3.org/TR/xquery/] qui accompagnera XPath 2 comme composante fondamentale des systèmes XML.

Malheureusement, ces langages sont souvent limités pour des bases de documents à forte dominante textuelle. Ainsi, on ne retrouve pas de tri de pertience, de proximité des mots, d'analyse linguistique, rarement de la troncature, etc. Pourtant, il serait intéressant de pouvoir chercher des documents XML avec des mots, à l'aide d'un outil qui accorderait plus d'importance aux documents qui contiennent les mots dans les titres plutôt que dans les paragraphes.

La puissance d'XML dans le domaine des bases de données documentaires vient du fait que l'on peut amalgamer des informations très structurées avec des informations plutôt textuelles, et donc moins structurées. Mais il faut que les outils suivent cette approche, ce qui est malheureusement peu le cas.

Par ailleurs, il est possible de concevoir la recherche d'information dans les documents structurés à l'aide d'une indexation intelligente, plutôt qu'un langage de recherche sophistiqué. C'est ainsi que fonctionne par exemple **SDX** [http://sdx.culture.fr] , ce qui permet d'utiliser n'importe quelle caractéristique de contenu ou de structure des documents XML, à condition que le concepteur de la base de documents ait prévu le besoin au départ.

La consultation

La consultation des documents doit se faire avec un navigateur. Il existe des navigateurs XML natifs, mais ils sont rares et plutôt inacessibles.

Heureusement, il existe de nombreuses autres solutions, que l'on peut regrouper en deux catégories: la conversion dans des formats de consultation et l'utilisation de navigateurs Web.

La conversion dans des formats de consultation

Le principal format de consultation d'information électronique est aujourd'hui le format HTML, avec les normes Javascript et CSS qui l'accompagnent. Il est facile de convertir les documents XML en format HTML, alors pourquoi s'en priver?

La plus grande partie des documents XML aujourd'hui disponibles sont convertis, à un moment ou un autre, en format HTML pour être consultés. Pour ce faire, la norme XSLT joue un rôle important.

Cette norme est suffisamment complexe pour faire l'objet d'une formation à part et ne fera donc pas l'objet d'une documentation ici. Toutefois, on peut consulter le **FAQ** [http://www.dpawson.co.uk/xsl/xslfaq.html] et les différents **tutoriels** [http://www.dpawson.co.uk/xsl/sect1/N856.html] qu'il recense.

Les navigateurs Web

Internet Explorer version 5 et ultérieure (avec une mise à jour spécifique toutefois) permet de consulter des documents XML, soit avec une feuille de style CSS associée, soit avec une transformation XSLT qui donne du HTML. Il est donc possible, avec cet outil, de consulter des bases de documents XML sans conversion extérieure.

Par ailleurs, Netscape 6 permet la consultation de documents XML avec une feuille de style CSS associée, de même que la navigateur Opera.

Ces outils sont intéressants, mais la multitude de navigateurs, les différents bogues rencontrés et les différences entre plate-formes font en sorte que la plupart du temps, la consultation des documents XML sur le Web se fait suite à une conversion en format HTML côté serveur.

L'impression

L'impression de qualité professionnelle à partir de documents XML est possible de différentes façons. Encore une fois, on peut convertir les documents dans un format approprié, par exemple les XPress Tags pour le logiciel Quark XPress.

Il existe maintenant d'autres solutions basées sur la norme **XSL-FO** [http://www.w3.org/TR/xsl/] , qui vise a définir des objets de mise en page universels. Des outils permettent de convertir des objets de mise en page en format PDF directement : **FOP** [http://xml.apache.org/fop/] (libre), de même que **XEP** [http://www.renderx.com/FO2PDF.html] et **XSL Formatter** [http://www.antennahouse.com/xslformatter.html] , deux produits commerciaux. Il s'agit de la voie d'avenir pour l'impression des documents XML.