Cahiers GUTenberg

FOIRE AUX QUESTIONS XML

Peter Flynn

Cahiers GUTenberg, nº 33-34 (1999), p. 281-311.

<http://cahiers.gutenberg.eu.org/fitem?id=CG_1999___33-34_281_0>

Foire aux questions XML

Peter FLYNN*

University College Cork, <pflynn@imbolc.ucc.ie>

Questions fréquemment posées sur XML (Version 1.5 de juin 1999)

Mis à jour sous l'égide du Groupe d'intérêt spécial (SIG) XML du Consortium World Wide Web par Peter FLYNN (*University College Cork*), avec la collaboration de Terry ALLEN, Tom BORGMAN (*Harlequin Ltd*), Tim BRAY (*Textuality, Inc*), Robin COVER (*Isogen, Inc*), Bob DUCHARME *(Moody's)*, Christopher MADEN (*O'Reilly & Associates*), Eve MALER (*Arbortext, Inc*), Peter MURRAY-RUST (*Nottingham University*), Liam QUIN (*Groveware*), Michael SPERBERG-MCQUEEN (*University of Illinois at Chicago*), Joel WEBER (*MIT*), Murata MAKOTO (*Fuji Xerox Information Systems*) et avec la participation de nombreux autres membres du Groupe d'intérêt spécial XML du W3C et des lecteurs du monde entier de cette FAQ.

Pour apporter toute correction ou information complémentaire, veuillez envoyer un courriel à l'auteur Peter FLYNN.

Résumé

Ce document reprend les questions les plus fréquemment posées (FAQ ou foire aux questions) – ainsi que leurs réponses – sur le thème du XML (E*xtensible Markup Language*, langage extensible de balisage). Elles ont pour but de fournir aux utilisateurs, développeurs et autres lecteurs intéressés par le sujet un premier niveau d'information, mais ne sauraient en aucune sorte faire partie de la norme XML (page 305).

Organisation

Cette FAQ comporte quatre parties : *a*) Général (page 286), *b*) Utilisateur (page 291), *c*) Auteur (page 293) et *d*) Développeur (page 305). Les questions sont numérotées indépendamment les unes des autres à l'intérieur de ces parties. Dans la mesure où cette numérotation évolue avec chaque version du document, il est préférable d'indiquer le numéro de la version ainsi que le numéro de la partie et de la question dans les commentaires et suggestions apportés.

* Traduction française de Morgane LE BIHAN et Dreves EWEN. Voir page 283.

Un formulaire est à votre disposition en fin du document électronique : vous pouvez y mentionner les bogues que vous auriez décelées, vos suggestions pour améliorer ce document, ou tout autre commentaire concernant *uniquemen cette FAQ* (vous pouvez aussi envoyer un courrier électronique au responsable du document : <pflynn@imbolc.ucc.ie>). Les commentaires concernant la norme XML (page 305) sont à adresser directement au W3C (http://www.w3.org/).

Disponibilité des documents

— Le fichier SGML (page 286) - à utiliser avec tout système conforme à la norme SGML - est disponible sous à http://www.ucc.ie/xml/faq.sgml (ce fichier est aussi accessible en ligne avec des navigateurs SGML comme *Panorama (http://www.sq.com/products/panorama/panor-fe.htm)* ou *Multidoc Pro (http://www.citec.fi/mdp/index.html)* ; vous pouvez aussi télécharger l'auto-extracteur d'installation de la DTD et de la feuille de style (http://www.ucc.ie/xml/xmlview.exe) pour un accès local plus rapide avec ces navigateurs, ou la DTD en ASCII (http://www.ucc.ie/xml/catalog)).

— Ce document existe aussi en version HTML (http://www.ucc.ie/xml/faq.html) - à utiliser avec un navigateur HTML (par exemple *Netscape Navigator (http://www.netscape.com/)*, *Microsoft Internet Explorer (http://www.microsoft.com/)*, *Spry Mosaic (http://www.spry.com/)*, *NCSA Mosaic (http://www.ncsa.edu)*, *Lynx (http://kufacts.cc.ukans.edu/)*, *Opera (http://opera.nta.no)*, *GNUscape Navigator (http://www.cs.indiana.edu/???)*, etc) à l'adresse http://www.ucc.ie/xml/.

— Une version XML sera créée lorsque les DTD et navigateurs compatibles seront plus largement disponibles.

— Une version sans mise en forme (Ascii) est disponible à partir du Web (http://www.ucc.ie/xml/faq.txt) ou par FTP anonyme (http://www.ucc.ie/doc/other/howtoftp.html) sur l'un des nombreux sites de FAQ (ftp://rtfm.mit.edu/). Les versions mentionnées ci-dessus sont aussi disponibles par courrier électronique (mailto:webmail@www.ucc.ie) au serveur *WebMail (http://www.ucc.ie/webmail/)* (pour les utilisateurs n'ayant accès que par courrier électronique). La version sans mise en forme est envoyée à comp.text.xml chaque mois pour archivage.

— Des exemplaires papier au format A4 (http://www.ucc.ie/xml/faqa4.ps) et *Letter* (http://www.ucc.ie/xml/faqlet.ps) sont disponibles en version PostScript^{TM}. Une version PDF sera prochainement disponible pour ces mêmes formats A4 et *Letter*.

— Le document est aussi disponible sur un « support en bois mort applati avec encre à base d'huile », en envoyant 10USD (ou montant correspondant en devises) à l'éditeur (<silmaril@m-net.arbornet.org>) (envoyez d'abord un courrier électronique pour vérifier la devise et l'adresse postale).

Merci à Murata Makoto pour la traduction de ce document en japonais (http://www.fxis.co.jp/DMS/sgml/xml/xmlfaq.html) ; à Jaime Sagarduy pour la traduction en espagnol (http://slug.ctv.es/~olea/sgml-esp/xfaq13.html) ; à Kangchan Lee pour la traduction en coréen (http://xml.t2000.co.kr/faq/index.html) ; à Jiang Luqin pour une version en chinois (en préparation) et à Neko pour une autre (http://zxd.webjump.com/xml.

html) ; à Miloslav Nic pour une version tchèque (en préparation) et à Tim Bray pour les *Annotated Spec* - annotations apportées aux spécifications - (http://www.xml.com/axml/testaxml.htm). Une version grecque est également en cours de rédaction.

Vous pouvez télécharger le logo XML en format GIF (http://www.ucc.ie/xml/xml.gif) ou EPS (http://www.ucc.ie/xml/xmllogo.eps) et une icône pour vos fichiers en format ICO (http://www.ucc.ie/xml/xml.ico) (Microsoft Windows), Mac (http://www.ucc.ie/xml/xml_folder_icon.sit.hqx), ou XBM (http://www.ucc.ie/xml/xml.xpm) (système X Window).

À propos de la version française de cette FAQ

Cette FAQ est traduite de *The XML FAQ* (http://www.ucc.ie/xml/) version 1.5 de juin 1999. La traduction a été commanditée par l'Association GUTenberg (http://www.gutenberg.eu.org/) au CFTTR (Centre de Formation de Traducteurs-Terminologues & Rédacteurs de l'Université de Haute Bretagne - Rennes 2, http://www.uhb.fr/langues/Craie/cfttr/) . Elle a été traduite de l'anglais par Morgane LE BIHAN et Dreves EWEN, étudiants en 5e année de DESS Langes et techniques. Ils se sont basés sur la terminologie utilisée pour la version française du langage XML (voir dans ce *Cahier* pages 191–280 et http://babel.alis.com/web_ml/xml/). Elle a été révisée par Jacques ANDRÉ (Irisa/Inria-Rennes), Bernard GAULLE (Association GUTenberg) et Michel GOOSSENS (CERN, Genève).

Les questions

A. Questions d'ordre général

B. Utilisateurs de SGML (HTML compris)

C. SGML (y compris HTML) vu des auteurs

D. Développeurs et implémenteurs (y compris les webmasters et les responsables serveurs)

Les réponses

A. Questions d'ordre général

A.1 *Qu'est-ce-que XML ?*

XML signifie *Extensible Markup Language* (langage extensible de balisage.). Ce langage est dit extensible car, contrairement à HTML (page 287), il ne s'agit pas d'un format fixe. Son but est de permettre d'utiliser SGML (page 286) sur le *World Wide Web*.

XML n'est pas un simple langage de balisage prédéfini. C'est un méta-langage – un langage qui permet de définir d'autres langages – qui vous permet de concevoir votre propre balisage. Un langage de balisage prédéfini comme HTML définit une façon de décrire les informations pour une certaine classe de documents : XML vous permet de définir vos propres langages de balisage personnalisés pour un ensemble de classes de documents. Ceci est possible parce que XML est écrit en SGML, le méta-langage international normalisé de balisage.

A.2 *À quoi sert XML ?*

XML est conçu pour « rendre l'utilisation de SGML (page 286) sur le Web facile et directe : il sera facile de définir des types de documents, facile de créer et de gérer des documents définis en SGML et facile de les transmettre et de les partager sur le Web. »

XML définit un « dialecte » ou sous-ensemble de SGML extrêmement simple, que la spécification XML (page 305) décrit complètement. Avec XML, l'objectif est d'envoyer, de recevoir et de traiter du SGML générique comme il est aujourd'hui possible avec HTML (page 287)

« C'est pourquoi XML a été développé pour sa facilité de mise en oeuvre et son interopérabilité avec SGML et HTML » [citations extraites de la spécification XML (page 305)].

A.3 *Qu'est-ce-que SGML ?*

SGML signifie Standard Generalized Markup Language (`http://www.oasis-open.org/cover/sgml-xml.html`) (ISO 8879 (`http://www.iso.ch/`)), langage normalisé de balisage généralisé ; c'est la norme internationale pour décrire la structure et le contenu de différents types de documents électroniques. La FAQ sur SGML (`http://www.infosys.utas.edu.au/info/sgmlfaq.txt`) est envoyée chaque mois au newsgroup `comp.text.sgml` ; les pages Web en SGML sont disponibles à `http://www.oasis-open.org/cover/sgml-xml.html`.

Les normes ISO sont gouvernées par l'Organisation Internationale de Normalisation à Genève (Suisse). Leur existence ou disparition est votée par les représentants des organismes de normalisation de chaque pays.

- Pour toute requête concernant une norme internationale, veuillez contacter l'organisme de normalisation de votre pays qui vous donnera le nom du représentant de votre pays intervenant dans le comité ou groupe de travail ISO concerné.
- Pour toute requête concernant la représentation de votre pays à Genève, ou la bonne conduite de l'organisme national de normalisation vous représentant, veuillez contacter le service approprié de votre gouvernement, ou vous adresser directement à votre représentant ou tout autre homme politique.

La représentation des pays à l'ISO ne relève pas de cette FAQ. Les requêtes sur le pourquoi et le comment du vote ou de l'absence de vote de vos représentants à l'ISO ne doivent donc en aucun cas être adressées au responsable de cette FAQ.

A.4 *Qu'est-ce-que HTML ?*

HTML signifie HyperText Markup Language (`http://www.w3.org/MarkUp`, RFC 1866 `http://ds.internic.net/rfc/rfc1866.txt`), ou langage de balisage avec liens hypertextes. C'est une application spécifique de SGML (page 286) utilisée sur le *World Wide Web* (`http://www.w3.org/`).

A.5 *XML, SGML et HTML n'est-ce pas la même chose ?*

Pas exactement. SGML (page 286), c'est la « langue maternelle » utilisée pour décrire des milliers de types de documents différents dans de nombreux champs de l'activité humaine, de la transcription des anciens manuscrits irlandais à la documentation technique des bombardiers furtifs et des dossiers médicaux de patients jusqu'aux partitions de musique.

HTML (page 286) n'est qu'un de ces types de documents, celui qui est le plus utilisé sur le Web (`http://www.w3.org/`). HTML définit un type de document unique, fixe, dont le balisage a été défini pour une classe commune de rapports simples de type « bureau » ou de rapports techniques, avec des entêtes, des paragraphes, des listes, des illustrations, etc. et avec la possibilité d'éléments hypertextes et multimedia.

XML est une version abrégée de SGML, qui permet de définir ses propres types de documents plus facilement, et qui permet aux programmeurs de développer plus facilement des programmes permettant de traiter ces documents. Les parties les plus complexes et les moins utilisées de SGML ont été supprimées. Du coup, ce produit a gagné en avantages : écriture d'applications plus facile, compréhension plus aisée et produit mieux adapté à sa mise en place et à son interopérabilité sur le Web. Mais il s'agit toujours de SGML et les fichiers XML peuvent toujours être analysés et validés comme tout autre fichier SGML (voir la question sur les logiciels XML (page 308)).

Les programmeurs trouveront peut-être pratique de considérer XML comme un SGML−− plutôt qu'un HTML++.

A.6 *Quelle différence y-a-t'il entre SGML/XML et C ou C++ ?*

C ou C++ (et autres langages comme Fortran, Pascal, Basic, Java et des dizaines d'autres) sont des *langages de programmation* qu'on utilise pour spécifier des calculs, des actions ou des décisions à prendre :

```
do when @front(@date,6) is equal "01-Apr"
     print "April Fool!\n"
else
     print @days(@datesub("25-Dec",@date)),\
           " shopping days to Christmas\n"
done
```

SGML et XML sont des langages de spécification de balisage qu'on utilise pour décrire des informations, en général pour le stockage, la transmission ou pour être traités par un programme :

```
<para>It was the week after <event class="festival">Christmas</event>
 but <name class="person">Max</name>s mind was still running on the
 prank he had played on <name class="person">Louise</name> the previous
 <name class="month">April</name>.</para>
```

Tout seuls, des fichiers-textes en SGML ou XML (y compris HTML) ne *font* rien du tout : il faut un programme qui *en* fasse quelque chose.

A.7 *Qui est responsable de XML ?*

XML est un projet duWorld Wide Web Consortium (W3C) (`http://www.w3.org/`) et le développement de sa norme est supervisé par son Groupe de travail XML. Un Groupe d'intérêt spécial composé de collaborateurs admis par cooptation et d'experts issus de divers domaines a apporté commentaires et révisions via le courrier électronique.

XML est un format public : aucune société n'est propriétaire de son développement. La spécification v1.0 (page 305) a été entérinée par le W3C (recommandation du 10 février 1998).

A.8 *Pourquoi le développement de XML est-il si important ?*

XML permet de supprimer deux contraintes qui limitent les développements du Web :

1. dépendance envers un type de document unique et non flexible (HTML, voir page 287) ;
2. complexité du SGML (page 286) intégral, dont la syntaxe autorise un grand nombre d'options puissantes mais difficiles à programmer.

XML simplifie le degré d'optionalité de SGML tout en permettant de développer sur le Web des types de documents créés par l'utilisateur.

A.9 *En quoi XML simplifie-t-il SGML tout en permettant de définir des types de documents personnalisés ?*

Pour simplifier SGML, XML redéfinit certaines fonctions et certains paramètresSGML (page 286) et supprime un grand nombre des fonctions les plus complexes et parfois les moins utilisées qui rendent l'écriture de programmes de traitement difficile (voir `http://www.w3.org/TR/NOTE-sgml-xml-971215`).

Même si XML garde toutes les fonctions de structure de SGML qui permettent de définir son propre type de document, XML introduit aussi une nouvelle classe de documents qui n'oblige pas à utiliser un type de document *prédéfini* (en gros, on peut définir son propre balisage à condition de suivre rigoureusement les règles syntaxiques). Voir les questions sur les documents « valides » et « bien formés » (page 305) et la question Comment définir vos propres types de documents ? (page 298) dans la Partie Développeur (page 305).

A.10 *Pourquoi ne continue-t-on pas à développer HTML ?*

HTML (page 287) est déjà surchargé de dizaines d'inventions de divers constructeurs intéressantes mais souvent incompatibles car il ne prévoit qu'une seule manière de décrire vos informations.

XML permettra à des groupes de personnes ou à des organisations de créer leurs propres langages de balisage personnalisés afin d'échanger des informations dans leurs domaines (musique, chimie, électronique, randonnée, finance, surf, géologie pétrolière, linguistique, cuisine, tricot, cartographie stellaire, histoire, ingénierie, élevage de lapins, mathématiques (page 301), etc).

HTML a atteint les limites de son utilité en tant que descripteur d'informations. Et même s'iil continue à jouer un rôle important pour le contenu qu'il représente aujourd'hui, de nombreuses autres applications nécessitent une infrastructure plus robuste et plus flexible.

A.11 *A quoi servent tous ces trucs en SGML ? Pourquoi ne pas tout simplement utiliser* Word *or* Notes*?*

Les informations qui circulent sur un réseau qui relie différents types d'ordinateurs doivent être utilisables par tous ces ordinateurs. Les informations publiques ne peuvent pas se restreindre à une seule marque ou à un seul modèle ou à un seul fabricant, ni ne peuvent céder le format de leurs données à des mains privées. De plus il est pratique que ces informations soient dans un format qui puisse être réutilisé de différentes façons, de manière à réduire temps et effort. Les formats de données propriétaires, même lorsqu'ils sont bien documentés, ne sont pas utilisables : leur contrôle reste dans les mains de privés et ils peuvent être changés ou retirés arbitrairement sans préavis.

SGML (page 286) est la norme internationale utilisée pour définir ce type d'applications. Mais les personnes ayant besoin d'une option différente basée sur un autre logiciel sont entièrement libres de mettre en place des services similaires en utilisant ce système et plus particulièrement s'ils sont conçus pour un usage privé.

A.12 *Où puis-je obtenir plus d'informations sur XML ?*

Divers documents émanant du W3C sont disponibles en ligne : la Spécification XML (page 305) et la documentation qui s'y rapporte ; une section XML (`http://www.oasis-open.org/cover/xml.html`) avec une longue liste de matériels de référence en ligne sur les pages SGML (`http://www.oasis-open.org/cover/sgml-xml.html`) de Robin Cover ; un résumé (`http://www.textuality.com/xml/`) et une FAQ condensée (`http://www.textuality.com/xml/faq.html`) de Tim Bray.

Les manifestations ci-dessous sont celles dont on m'a parlées : merci de me contacter par courrier électronique (`mailto:pflynn@imbolc.ucc.ie`) si vous en trouvez d'autres.

— La Conférence annuelle XML est organisée par la *Graphic Communications Association*. XML99 (`http://www.gca.org/conf/xml99/xm99ndex.htm`) aura lieu à Philadelphie du 5 au 9 décembre et comprendra, comme l'an dernier, deux conférences en une : *XML Conference 99* et *Markup Technologies 99* (`http://www.gca.org/conf/mt99/mtndex.htm`).
— SGML/XML Asia/Pacific (`http://www.allette.com.au/asia99/xml.html`) a eu lieu à Sydney du 18 au 21 octobre.

De plus amples informations sur ces manifestations sont disponibles sur le site Web du GCA (`http://www.gca.org/`).

Les pages SGML et XML (`http://www.oasis-open.org/cover/sgml-xml.html`) de Robin Cover proposent une liste des livres et articles sur XML : ce site devrait toujours être votre principale ressource : merci de le consulter avant d'utiliser le formulaire de cette FAQ pour poser des questions sur les logiciels et la documentation.

A.13 *Où puis-je discuter de développement et d'implémentation de XML ?*

> Merci de lire le fichier *Read The Fine Documentation* (Lire la documentation détaillée) que vous recevez en vous abonnant à une liste de discussion. En effet, il contient des informations importantes, notamment sur ce qu'il faut faire si votre adresse de courrier électronique change.

Une liste de diffusion appelée `xml-dev` est accessible aux personnes impliquées dans le développement de composants pour XML. Pour vous abonner à cette liste, envoyez un courrier électronique d'une ligne à `<majordomo@ic.ac.uk>` en indiquant :

```
subscribe xml-dev votre_nom@votre_adresse
```

(en substituant votre adresse électronique). Pour vous désabonner, envoyez un message d'une seule ligne à la même adresse, disant :

```
unsubscribe xml-dev votre_nom@votre_adresse
```

On peut aussi accéder à cette liste à l'adresse `http://www.lists.ic.ac.uk/hypermail/xml-dev/`. Attention : cette liste s'adresse aux personnes réellement concernées par le développement de ressources pour XML. Il ne s'agit *pas* d'informations générales sur XML (voir la FAQ et les autres sources (page 289)) ni de discussions générales sur la mise en place de SGML ni sur les ressources SGML (voir ci-dessous).

Une liste de diffusion du domaine général appelée `XML-L` est accessible pour des discussions publiques. Pour vous abonner à cette liste, envoyez un courrier électronique d'une ligne à `<LISTSERV@listserv.heanet.ie>` en indiquant :

```
subscribe XML-L prénom nom
```

(remplacez ces libellés par vos propres prénom et nom). Pour vous désabonner, envoyez un courrier électronique d'une ligne à la même adresse en indiquant :

```
unsubscribe XML-L
```

À noter que les listes comme `LISTSERV` ne demandent pas votre adresse électronique : elles la trouvent dans l'entête de votre message. On peut accéder à XML-L et à ses archives, ou s'abonner et se désabonner interactivement, par `http://listserv.heanet.ie/xml-l.html`.

Note Notez qu'il y a un grand nombre d'informations erronées ou incomplètes qui trainent dans des documents papier ou sur le Web à propos des listes de discussions. Les informations données ici sont correctes : utilisez les !

Certaines listes de discussions concernent des langues autres que l'anglais :

— Gianni Rubagotti signale : « Une nouvelle liste italienne de discussion sur XML est née : pour s'abonner, envoyer un message (sans sujet) disant

```
subscribe XML-IT
```

à `<majordomo@ananas.usr.dsi.unimi.it>`. Envoyez vos messages de discussion à : `<xml-it@ananas.usr.dsi.unimi.it>` (seuls les abonnés peuvent envoyer un mesage). Toute personne, italienne ou non, désirant débattre de XML en italien est bien venue. »

— JP Theberg écrit : « Une liste française sur XML a été créée. Pour s'abonner, envoyez `subscribe` à `xml-request@trisome.com`. Envoyez vos messages ensuite à `xml@trisome.com`. »

Le *newsgroup* USENET `comp.text.xml` (`news:comp.text.sgml`) sert aux discussions sur XML. S'il n'est pas disponible sur votre serveur local, demandez à votre fournisseur d'accès à d'Internet de l'installer, ou utilisez une interface Web comme déjàNews (`http://www.dejanews.com`).

B. Utilisateurs de SGML (HTML compris)

B.1 *Que dois-je faire pour utiliser XML ?*

Pour l'utilisateur du Web moyen, rien sauf d'utiliser un navigateur qui marche avec XML (voir les questions sur les navigateurs (page 292)). Rappelons que XML est encore en cours d'implémentation, aussi certaines possibilités ne sont pas encore définies ou sont encore à écrire.

On peut utiliser des navigateurs XML pour voir des produits naissants, comme les pièces de Shakespeare de Jon Bosak (`ftp://sunsite.unc.edu/pub/sun-info/standards/xml/eg/`) ou les expériences sur les molécules du CML (*Chemical Markup Language*, un langage de balisage pour la chimie) (`http://www.xml-cml.org`). On trouvera d'autres exemples dans à `http://www.oasis-open.org/cover/xml.html#examples`.

Pour commencer à vous préparer à écrire votre propre XML, voir les questions dans les parties Auteurs (page 293) et Développeurs (page 305).

B.2 *Pourquoi devrais-je utiliser XML plutôt que HTML ?*

— Les auteurs et les fournisseurs d'accès peuvent concevoir leurs propres types de documents (page 298) à l'aide de XML, au lieu de dépendre de HTML. Les types de documents peuvent explicitement être personnalisés par rapport à un public donné. Ainsi les casse-têtes inhérents à HTML (page 287) lors de la mise en place d'effets spéciaux devraient bientôt appartenir au passé : les auteurs et les concepteurs seront libres d'inventer leurs propres balises.
— Le contenu informatif peut être plus riche et plus facile à utiliser parce que les fonctions de liens hypertextes de XML (page 300) sont bien supérieures à celles de HTML.
— XML peut offrir des fonctions meilleures et plus nombreuses en termes de présentation et de performance du navigateur.

— Il supprime beaucoup des complexités inhérentes à SGML, en échange d'un modèle plus flexible. Ainsi, l'écriture de programmes pour manipuler XML sera bien plus facile que l'écriture de programmes similaires pour manipuler un SGML complet.
— Les informations seront accessibles et réutilisables, parce que le langage le plus flexible de XML peut être utilisé par n'importe quel logiciel XML plutôt que d'être restreint à des fabricants spécifiques comme c'est devenu le cas avec HTML.
— Les fichiers XML valides (page 306) sont des fichiers SGML conformes à la norme, c'est-à-dire qu'ils peuvent aussi être utilisés hors du Web, dans un environnement SGML.

B.3 *Où puis-je trouver un navigateur XML ?*

Souvenez-vous que la spécification XML (page 305) est encore récente, donc tout ce que vous voyez est encore expérimental. Comme avec HTML (page 287), il n'y aura pas qu'un seul navigateur mais plusieurs. Cependant, parce que le nombre potentiel d'applications XML différentes est illimité, aucun navigateur donné ne peut traiter tous les cas à 100%.

Certaines parties génériques de XML (par exemple l'analyse, la gestion d'arbres, la recherche, le formatage, etc) sont combinées dans des bibliothèques de navigateurs ou boîtes à outils (toolkits) pour rendre la tâche plus facile et rigoureuse aux développeurs (page 305) lorsqu'ils développent des applications XML. De telles applications peuvent ensuite être personnalisées par l'ajout de sémantique propre à des marchés donnés, ou par l'utilisation de langages tels que *Java* pour développer des *plugins* pour navigateurs génériques et pour délivrer des modules spécialisés transparents pour le Web.

— MSIE5 traite XML mais l'affiche par le biais de CSS, en utilisant un modèle très largement emprunté à HTML, aussi toutes les options de feuilles de style ne marchent-elles pas. Microsoft a aussi défini l'architecture d'une solution hybride dans laquelle on peut mélanger des morceaux de XML dans un fichier HTML puisque les navigateurs spécifiques à HTML ignorent simplement les balises qu'ils ne reconnaissent pas.
— Le code Netscape à diffusion publique (Mozilla (`http://www.mozilla.org/`)) a débouché sur une réalisation test XML comprenant une application de RDF et l'analyseur XML *expat* (`http://www.jclark.com/xml/expat.html`) de James Clark.
— Les auteurs du navigateur SGML *MultiDoc Pro*, CITEC (`http://www.citec.fi/`), ont fait un gros effort avec Mozilla pour produire un navigateur multi-tout appelé *DocZilla* qui lit HTML, XML et SGML, avec XSL et les feuilles de style CSS. Ça marche sous NT et linux et est actuellement en version alpha (voir `http://www.doczilla.org` pour plus de détails). C'est vraiment du début d'alpha, mais c'est de loin la version la plus ambitieuse et la seule qui soit basée sur une réelle expertise SGML.

Voir aussi les notes sur les logiciels pour les auteurs (page 308) et les développeurs (page 309) et la liste plus détaillée de pages XML disponible sur le site Web SGML à l'adresse `http://www.oasis-open.org/cover/xml.html`.

B.4 *Est-ce-que je dois passer de SGML ou HTML à XML ?*

Non, les logiciels existants pour applications SGML (page 286) et HTML (page 287) continueront à fonctionner avec les fichiers existants. Mais comme c'est le cas avec toute fonction améliorée, si vous voulez visualiser ou télécharger et utiliser des fichiers XML, il vous faudra ajouter dès qu'il y en aura sur le marché, un logiciel qui reconnaisse XML.

C. SGML (y compris HTML) vu des auteurs

Les auteurs devraient également lire la partie Développeur (page 305) : elle contient de plus amples informations sur le contenu des fichiers XML.

C.1 *Est-ce que XML remplace HTML ?*

Non, XML en soi ne remplace pas HTML (page 287) : il constitue en fait une alternative qui vous permet de définir vos propres éléments de balisage. HTML devrait encore être utilisé pendant un certain temps. Les DTD (page 298) pour HTML seront ensuite disponibles au format XML ainsi qu'au format original SGML. XML est conçu pour rendre l'écriture de DTD plus facile qu'avec SGML (page 286) complet (voir les questions sur les DTD (page 298) pour savoir ce que c'est et pourquoi vous n'en voulez pas).

Des travaux en cours (page 297) devraient conduire à la conception de versions XML de DTD HTML et autres DTD connues, mais ceci ne pourra se faire avant que des logiciels plus fiables ne soient disponibles. Surveillez les annonces sur `comp.text.sgml`, `comp.text.xml`, `XML-L` et `xml-dev`.

C.2 *A quoi ressemble l'intérieur d'un document XML ?*

La structure de base ressemble énormément à celle de la plupart des applications de SGML y compris HTML. Les documents XML peuvent être très simples, sans déclaration de type de document, et avec un balisage de votre conception directement imbriqué :

```
 <?xml version="1.0" standalone="yes"?>
   <conversation>
     <greeting>Hello, world!</greeting>
     <response>Stop the planet, I want to get off!</response>
   </conversation>
```

Ces documents peuvent également être plus complexes, avec une DTD donnée, et peut-être un sous-ensemble interne et une structure plus complexe :

```
<?xml version="1.0" standalone="no" encoding="UTF-8"?>
  <!DOCTYPE titlepage SYSTEM "http://www.frisket.org/dtds/typo.dtd"
  [<!ENTITY % active.links "INCLUDE">]>
  <titlepage>
    <white-space type="vertical" amount="36"/>
    <title font="Baskerville" size="24/30"
           alignment="centered">Hello, world!</title>
    <white-space type="vertical" amount="12"/>
    <!-- In some copies the following decoration is
          hand-colored, presumably by the author -->
    <image location="http://www.foo.bar/fleuron.eps" type="URL"
           alignment="centered"/>
    <white-space type="vertical" amount="24"/>
    <author font="Baskerville" size="18/22" style="italic">
      Vitam capias</author>
  </titlepage>
```

Ils peuvent également se situer à mi-chemin entre ces deux options : cela dépendra essentiellement de la façon dont vous définirez votre type de document (ou celui que vous utiliserez) et de son usage. Voir la question sur les fichiers valides et bien formés (page 305).

C.3 *Comment XML traite-t-il les espaces blancs présents dans mes documents ?*

Les règles de SGML concernant les espaces blancs ou séparateurs ont été modifiées pour XML. Ainsi, *tous* les séparateurs, y compris les sauts de lignes, les tabulations et les espaces normaux, *même entre les éléments où aucun texte ne peut apparaître*, passent *tels quels* dans l'application (navigateur, logiciel de formatage, visionneur, etc). Cela signifie que :

— les séparateurs « insignifiants » entre les éléments de structure (les éléments qui ne peuvent contenir que d'autres éléments et non pas du texte, parfois appelés « contenus élémentaires ») *passeront* dans l'application (avec du SGML intégral, les séparateurs sont supprimés) ;
— les séparateurs « signifiants » entre les éléments qui *peuvent* contenir à la fois du texte et du balisage (« contenu mixte » ou PCDATA [*parsed character data*, données textuelles analysées]) *passeront* dans l'application comme auparavant.

```
<chapter>
    <section>
      <title>
        My title for Section 1.
      </title>
      <para>
        ...
      </para>
    </section>
  </chapter>}
```

Le parser doit toutefois indiquer à l'application si tel séparateur est apparu dans tel contenu élémentaire, s'il est connu. (Les utilisateurs de SGML intégral reconnaîtront peut-être que cette information n'apparaît pas dans le ESIS *Element Structure Information Set)* (`http://www.oasis-open.org/cover/WG8-n931a.html`), mais elle figure bel et bien dans le *grove (Graph representation of Property Values)* (`http://www.oasis-open.org/cover/topics.html#groves`).) Dans l'exemple ci-dessus, l'application recevra tous les sauts de ligne, tabulations et espaces entre les éléments *ainsi que* ceux emboïtés dans le titre de section. Il revient à l'application (navigateur, logiciel de formatage, visionneur, etc.) de décider du type de séparateur à supprimer ou à garder.

C.4 *Quelles parties d'un document XML dépendent de la casse ?*

Tous les éléments de fichiers XML, balisage comme texte, dépendent de la casse. Ceci diffère considérablement du HTML et de la plupart des autres types de documents SGML. Cette contrainte a été mise en place pour permettre le balisage dans des écritures en alphabet non Latin et pour éviter des problèmes avec le traitement des différences entre minuscules et majuscules pour des écritures qui ne connaissent pas une telle différence.

— Les noms de type d'éléments (utilisés dans les balises de début et les balises de fin) dépendent de la casse : vous devez être très rigoureux dans leurs libellés (utilisation ouDTD (page 298)) que vous choisissiez les minuscules ou les majuscules ;
il est donc impossible de dire `<BODY> ... </body>` : les majuscules et minuscules *doivent* correspondre ; par conséquent, `<IMG/>` et `<img/>` sont *deux types d'élément différents* ;
— dans le cas des fichiers bien formés sans DTD, la *première occurrence*d'un nom de type d'élément en définit la casse
— les noms d'attributs dépendent eux aussi de la casse et correspondent aux éléments : par exemple, `<PIC width="7in"/>` et `<PIC WIDTH="6in"/>` dans un même fichier définissent deux attributs distincts, puisque la casse de `width` et `WIDTH` les différencie ;
— les valeurs des attributs dépendent elles aussi de la casse. Les valeurs des données textuelles (ex. : `HRef="MyFile.SGML"`) n'ont pas changé, mais les attributs ID et IDREF dépendent désormais de la casse et ne basculent plus en majuscules pour permettre la comparaison ;
— tous les noms d'entité (p.ex. `&Aacute;`) et le contenu de vos données (votre texte) sont dépendantes de la casse, comme avant.

C.5 *Comment faire pour que mes fichiers HTML fonctionnent en XML ?*

Ils doivent être bien formés (page 305) (voir ci-dessous) et associés à une feuille de style. Une DTD (page 298) est facultative avec XML, mais les fichiers HTML convertis au format XML doivent aujourd'hui de toute façon n'avoir aucune DTD : il n'existe encore aucune version XML des actuelles DTD HTML de type SGML (en cours de développement parce qu'elles doivent être considérablement modifiées pour ne plus dépendre des fonctions du SGML exclues du XML.

Il est nécessaire de convertir les fichiers HTML existants afin qu'ils soient bien formés, XML interdisant la suppression des balises de fermeture (absence de </p>, etc.), autorisée dans la plupart des DTD HTML. De nombreux éditeurs HTML produisent déjà du XML presque *bien formé* (mais pas tout à fait). Pour se préparer à XML, le programme HTML Tidy (`http://www.w3.org/People/Raggett/tidy/`) est recommandé pour nettoyer certaines erreurs de formatage tolérées ou produites par des éditeurs HTML trop tolérants.

Si vous voulez transformer vos fichiers HTML en un autre type de DTD, le site pilote de CommerceNet (`http://www.xmlx.com/`) permet l'échange des DTD XML et le serveur pilote FPI (`http://www.ucc.ie/cgi-bin/public`) vous propose plusieurs DTD courantes à partir desquelles commencer.

Si vous avez créé vos fichiers HTML (page 287) conformément à une des nombreuses Définitions de Type de Document (DTD) (page 308) HTML et qu'ils sont valides, ces fichiers peuvent alors être convertis de la façon suivante :

— remplacez la déclaration `DOCTYPE` et tout sous-ensemble interne (en fait tout ce qui se trouve dans le premier ensemble de chevrons `<!DOCTYPE HTML...>`) par la déclaration XML `<?xml version="1.0" standalone="yes"?>` ;
— changez tous les éléments `EMPTY`(ex. : `<ISINDEX>`, `<BASE>`, `<META>`, `<LINK>`, `<NEXTID>` et `<RANGE>` dans l'en-tête, et `<IMG>`, `<BR>`, `<HR>`, `<FRAME>`, `<WBR>`,

`<BASEFONT>`, `<SPACER>`, `<AUDIOSCOPE>`, `<AREA>`, `<PARAM>`, `<KEYGEN>`, `<COL>`, `<LIMITTEXT>`, `<SPOT>`, `<TAB>`, `<OVER>`, `<RIGHT>`, `<LEFT>`, `<CHOOSE>`, `<ATOP>` et `<OF>` dans le corps) de façon à ce qu'ils se terminent par "`/>`", par exemple `<IMG SRC="mypic.gif" alt="Picture"/>` ;
- assurez-vous que les balises de fin de tous les éléments non vides sont bien présentes et correctes ; ex. : chaque `<P>` doit avoir un `</P>`, etc. Si votre HTML a été créé par un éditeur conformant, ceci peut être automatisé par un programme normaliseur comme *sgmlnorm* (partie de *SP (http://www.jclark.com/sp/)*) ou par la fonction `SGML-normalize` d'un éditeur comme *Emacs*/*psgml* ;
- masquez tous les caractères qui ne servent pas au balisage `<` et `&` (c-à-d les *literal*), en `&lt;` et `&amp;` respectivement ;
- assurez-vous que toutes les valeurs des attributs sont entre apostrophes (") ;
- assurez-vous que toutes les occurrences de tous les noms d'élément entre balises de début *et* de fin respectent la même casse (respect des majuscules et des minuscules) et ce dans tout le fichier ;
- assurez-vous que toutes les occurrences de tous les noms d'attribut possèdent la même casse et ce dans tout le fichier.

Soyez conscients que de nombreux navigateurs HTML pourraient ne pas accepter les éléments `EMPTY` XML avec barre oblique ; les changements ci-dessus ne sont donc pas rétro-compatibles (lors de la conversion de XML en HTML). La solution est d'ajouter une balise de fin fictive à tous les éléments `EMPTY`, de sorte que `<IMG src="foo.gif">` devienne `<IMG src="foo.gif"></IMG>`.

Si vous avez beaucoup de fichiers HTML valides, vous pourriez écrire un script dans un système de conversion SGML (tel que *Omnimark (http://www.omnimark.com)*, *Balise (http://www.balise.com)*, *SGMLC (http://www.dircon.co.uk/sgml/)*, ou un système utilisant une des bibliothèques SGML de *Perl*, *Python* ou *Tcl*) ; vous pourriez également utiliser des macros (d'un éditeur comme emacs) si vous savez ce que vous faites.

Si vos fichiers HTML ne sont pas valides (les fichiers HTML créés par la plupart des éditeurs WYSIWYG ne sont pas valides), ils devront être convertis manuellement. Même si les défauts sont réguliers et constants, les fichiers pourraient être en fait bien formés, et vous pourriez alors écrire un programme ou un script comme on l'a dit ci-dessus. Pour juger de l'invalidité ou de la non-conformité, vérifiez les points suivants :

- les fichiers contiennent-ils des erreurs de syntaxe de balisage ? Par exemple, y a-t-il des barres obliques inverses à la place des barres obliques dans les balises de fin ? ou des éléments qui s'imbriquent incorrectement (ex. : `<B>un élément qui commence<I>dans un élément</B> mais se termine en dehors</I>`) ?
- les fichiers contiennent-ils un balisage qui entre en conflit avec les DTD HTML, tel que des en-têtes dans des paragraphes ou des listes d'éléments en dehors des environnements de liste ?
- les fichiers utilisent-ils des éléments qui ne figurent dans aucune DTD ? Bien que la transformation en fichier bien formé sans DTD soit simple (parce que vous n'avez pas à définir les éléments à l'avance) la plupart des extensions de programmes propriétaires

- propres à chaque navigateur - n'ont jamais été formellement définies ; il est donc souvent impossible de savoir où elles peuvent être utilisées à bon escient

Le balisage valide mais insignifiant ou inutile devra peut-être être modifié avant la conversion (répétitions de paragraphes vides, de sauts de ligne, de tableaux vides, d'images GIF vides servant à l'espacement, etc.) car XML utilise des feuilles de style, vous n'en aurez donc pas besoin.

Voir les règles pour des fichiers XML « bien formés (page 305) » pour plus amples informations sur les points à vérifier en XML lors de la conversion.

C.6 *Y a-t-il une version XML de HTML ?*

Il y a des versions XML de la DTD HTML en préparation, mais aucune n'est encore prête :

— Ben Trafford (`<btrafford@worldnet.att.net>`) est en train de développer une version XML de HTML 3.2
— J'ai personnellement démarré l'écriture d'une version XML d'HTML Pro, mais ce n'est pas facile et j'aimerais que l'on me convinque que c'est utile !
— XHTML (Extensible HyperText Markup Language) (`http://www.w3.org/TR/xhtml1/`) est un projet du W3C. « Les spécifications définissent XHTML 1.0, une reformulation de HTML 4.0 comme une application XML 1.0, et trois DTD correspondant à celles définies par HTML 4.0. La sémantique des éléments et de leurs attributs est définie dans les *W3C Recommendation for HTML 4.0*. Cette sémantique sert de base à de futures extensions d'XHTML. La compatibilité avec des agents HTML existants est possible à condition de respecter un petit ensemble de recommandations. »

C.7 *Si XML est un sous-ensemble de SGML, puis-je directement utiliser les fichiers XML avec des outils SGML ?*

Oui, à condition que vous utilisiez un logiciel qui reconnaît les nouvelles *WebSGML AQdaptations* à la norme ISO 8879 (fonctions nécessaires pour supporter XML : forme particulière des éléments `EMPTY` ; certains aspects de la déclaration SGML tels que `NAMECASE GENERAL NO` ; déclarations d'attributs multiples, etc.).

Une autre façon est d'utiliser une DTD SGML qui vous permette de créer un fichier SGML (mais une DTD qui n'utilise pas d'éléments vides) ; supprimer ensuite la *DocType Declaration* de façon qu'il devienne un fichier XML sans DTD bien formé.

Aujourd'hui, peu d'outils permettent de garder intacts les fichiers XML à cause du format de ces éléments `EMPTY`, mais ça bouge. L'analyseur *nsgmls* dispose d'une option de conformité XML, introduite pour être utilisée avec Jade par ailleurs, les premiers éditeurs et analyseurs spécifiques pour XML sont en usage (voir la question sur les logiciels (page 308)).

C.8 *Je suis habitué à utiliser HTML. Est-ce que je peux apprendre XML facilement ?*

Oui, très facilement. Mais aujourd'hui, il manque toujours des cours, des outils simples et plus d'exemples de documents XML. Des documents XML bien formés (page 305) peuvent paraître similaires à HTML (page 287) excepté pour de petits mais très importants points de syntaxe.

La principale différence est que XML doit coller aux règles. Les navigateurs HTML en revanche vous permettent de créer du HTML dégradé car ils ignorent les informations erronées : avec XML, ou bien votre fichier est correct, ou bien ile ne marche pas !

C.9 *XML pourra-t-il utiliser des alphabets non-latins ?*

Oui, les spécifications XML (page 305) indiquent explicitement que XML utilise la norme internationale de codage des caractères sur 31 bits ISO 10646 (`http://www.iso.ch/`), qui couvre la plupart des langages humains (et quelques langages non-humains). Ceci est actuellement conforme à Unicode.

Les spécifications indiquent (2.2) : « Tous les processeurs XML devront accepter les codages UTF-8 et UTF-16 de la norme ISO 10646 . . . » UTF-8 est un codage d'Unicode en caractères 8-bit : les 128 premiers caractères sont identiques au codage ASCII, les suivants sont utilisés pour coder le reste d'Unicode en séquences de 2 à 6 octets. Dans sa forme mono-octet UTF-8 est donc identique à la norme ISO 646 IRV (ASCII) ; vous pouvez donc continuer à utiliser le code ASCII pour l'anglais ou toute autre langue non accentuée en utilisant l'alphabet latin. À noter que UTF-8 est incompatible avec la norme ISO 8859-1 (ISO Latin-1) au-delà du code 126 décimal (la fin d'ASCII). UTF-16 est semblable à UTF-8 mais avec une méthode de représentation des 16 plans suivants de 64 000 caractères comme deux caractères de 16 bits.

« . . . Les mécanismes qui signalent lequel des deux est utilisé et qui mettent d'autres codages en jeu, [. . .] sont traités lors des débats sur le codage des caractères. » Les spécifications XML (page 305) expliquent comment indiquer dans un fichier XML le codage du jeu de caractères utilisé.

L'utilisation d'UCS-4 peut seulement être spécifiée en SGML ou XML quand les adaptations WebSGML à la norme ISO 8879 sont mises en place : ceci permet d'utiliser des nombres supérieurs à huit octets dans la déclaration SGML.

« Quel que soit le codage utilisé, tous les caractères du jeu de caractères de la norme ISO 10646 peuvent être définis avec l'équivalent décimal ou hexadécimal de leur chaîne d'octets. » Donc, quel que soit le jeu de caractères que vous utilisez, vous pouvez toujours définir des caractères spécifiques d'un autre endroit du répertoire de codage en utilisant `&#`*`dddd`*`;` (code de caractères décimal) ou `&#x`*`HHHH`*`;` (code de caractères hexadécimal, en majuscule). La terminologie, comme les nombres, peut porter à confusion : voir ISO 10646 Concept Dictionary (`http://cns-web.bu.edu/pub/djohnson/web_files/i18n/ISO-10646.html`). Rick Jelliffe a "XML-isé" les ensembles d'entité caractères ISO (`http://www.oasis-open.org/cover/xml-ISOents.txt`).

C.10 *Qu'est-ce qu'une définition de type de document (DTD) ? Où en obtenir une ?*

Une DTD est un fichier (ou plusieurs fichiers à utiliser ensemble), écrit en XML, qui contient une définition formelle d'un type particulier de document. Elle définit les noms qui peuvent être utilisés pour les types d'éléments, où ils peuvent apparaître et comment ils s'arrangent les uns par rapport aux autres. Par exemple, si vous voulez qu'un type de document décrive des `<LIST>` qui contiennent des `<ITEM>`, une partie de votre DTD devra contenir quelque chose comme :

```
<!ELEMENT list (item)+>
<!ELEMENT item (#pcdata)>
```

Ceci définit une liste comme un type élément contenant un ou plusieurs items (c'est le rôle du signe plus) et des items commes des types éléments ne contenant que du texte. XML est le langage formel de spécification utilisé par les processeurs pour analyser automatiquement

une DTD et utiliser ensuite cette information pour identifier la place de chaque type d'élément et la relation qui lie ces éléments, afin de permettre l'utilisation des feuilles de style, des navigateurs, des moteurs de recherche, des bases de données, des sous-programmes d'impression, ou d'autres applications. Les instructions ci-dessus vous permettent de créer des listes qui sont conservées sous la forme :

```
<List><Item>Chocolate</Item><Item>Music</Item><Item>Surfing</Item></List>
```

La façon dont la liste apparaît sur l'écran ou sur papier dépend de votre feuille de style : vous n'avez normalement rien à mettre en XML pour affecter la mise en page comme il fallait le faire en HTML avant les feuilles de style.

En fait une DTD donne à l'avance aux applications l'information des noms et structures qui peuvent être utilisés dans un type de document particulier. Utiliser une DTD signifie que tous les documents appartenant à un type particulier seront construits et nommés de façon conforme.

Il existe déjà des milliers de DTD SGML dans tous les domaines (voir les pages Web SGML (`http://www.oasis-open.org/cover/sgml-xml.html`) pour des exemples). Nombre de ces DTD peuvent être téléchargées et utilisées librement. Vous pouvez aussi écrire vos propres DTD. Pour cela, comme pour tout langage, vous devez auparavant l'apprendre (voir ar exemple : *Developing SGML DTDs* par Maler et el Andaloussi, Prentice Hall, 1997, 0-13-309881-8). Cependant XML est beaucoup plus simple que SGML intégral : voir la liste de restrictions (page 308) qui montre ce qui a été supprimé. Les DTD SGML existantes doivent être converties au format XML pour être utilisées dans des systèmes XML : lire la question concernant la conversion des DTD SGML au format XML. (page 310) Attendez-vous à voir des annonces de DTD connues disponibles au format XML.

C.11 *Je reste attentif à des alternatives aux DTD. Qu'est-ce qu'un schéma ?*

Bob DuCharme écrit : « Beaucoup de développeurs XML ne sont pas satisfaits de la syntaxe des déclarations de balises décrites dans les spécifications, pour deux raison. Premièrement, ils pensent que si les documents XML sont si bons pour décrire la structure des informations, alors la description d'une structure de type de document (ses « schémas ») devrait être dans un document XML et non écrite avec sa propre syntaxe. En plus d'être plus consistent, ceci permettrait plus facilement d'éditer et manipuler le schéma avec des outils habituels de manipulation de documents. Deuxièmeement, ils pensent que la notation traditionnelle des DTD ne donne pas aux définisseurs de schémas la puissance nécessaire pour imposer assez de contrainte sur les données – par exemple, la possibilité de dire qu'un certain type élément doit toujours avoir une valeur positive, qu'il ne peut pas être vide ou qu'il peut faire partie d'une liste au choix. Ceci faciliterait le développemennt de logiciels utilisant ces données car le développeur aurait moins de code de détection d'erreurs à écrire. »

Note Les utilisateurs ayant une formation en informatique ou en bases de données devraient prendre conscience que les systèmes SGML – y compris XML – ne sont pas des systèmes de gestion de bases de données : ce sont des systèmes de balisage de textes. Bien qu'il y ait beaucoup de points communs, comme ceux décrits ici, certains concepts des uns peuvent ne pas exister dans les autres : XML n'a pas les mêmes possibilités que les bases de données, tout comme DBMS n'a pas de possibilités de balisage..

« Plusieurs groupes ont soumis au W3C des propositions alternatives pour exprimer les schémas de type de document. En plus d'offrir des contraintes de schémas comme le typage des données et les autres décrites ici, beaucoup s'appuient sur les tendances actuelles en logiciel telles que les méthodologies orientées-objets. Le groupe de travail sur les schémas du W3C est en train d'analyser ces propositions et de développer sa propre proposition basée sur les meilleurs possibilités suggérées dans les propositions existantes ou par les membres du *Schema Working Group.* »

C.12 *Comment XML modifiera-t-il les liens de mes documents ?*

Les possibilités de liens hypertextes des systèmes XML sont beaucoup plus développées que celles du HTML, ce qui vous permettra de faire plus de choses avec eux. Les liens existants de type `HREF` continueront à être utilisables, mais la nouvelle technologie de liaison est basée sur les enseignements tirés du développement d'autres standards utilisant des hypertextes tels que TEI (`http://www-tei.uic.edu/orgs/tei/`) et HyTime. (`http://www.oasis-open.org/cover/hytime.html`) Cette technologie autorise des liens bi-directionnels et multi-branches, ainsi que des liens vers une sélection de texte (dans vos documents ou autres) et non vers un point unique. SGML connaît déjà ces applications dans des navigateurs tels que *DynaText*, *Panorama* et *Multidoc Pro* et ce depuis de nombreuses années en les utilisant, on gagne expérience et compétence considérables.

Les documents *XML Linking Specification (XLink)* (`http://www.w3.org/TR/WD-xlink`) et *XML Extended Pointer Specification (XPointer)* (`http://www.w3.org/TR/1998/WD-xptr`) contiennent des spécifications provisoires détaillées. Un lien XML peut être soit une URL, soit un pointeur étendu de style TEI : *TEI-style Extended Pointer (XPointer)*, soit les deux. Une URL seule est supposée être une ressource (comme avec HTML) ; si un pointeur étendu suit cette URL, on suppose qu'il est une sous-ressource de cette URL ; un pointeur étendu seul est supposé s'appliquer au document courant

Un XPointer est toujours précédé par un des symboles `#`, `?`, ou `|`. Les symboles `#` et `?` ont les mêmes fonctions que dans les applications HTML ; le symbole `|` signifie que la sous-ressource peut-être trouvée en appliquant le XPointer à la ressource, mais la méthode à utiliser dépend de l'application.

La TEI Extended Pointer Notation (`http://etext.virginia.edu/bin/tei-tocs?div=DIV2;id=SAXR`) (EPN) est beaucoup plus puissante que « l'adresse fragmentée » à la fin de certaines URL. Elle permet en effet de spécifier l'emplacement d'une fin de lien en utilisant la structure du document et/ou les points fixes, connus comme les ID. Par exemple, l'expression la seconde occurrence liée du mot « XPointer » deux paragraphes plus haut pourrait avoir pour référence `http://www.ucc.ie/xml/faq.sgml#ID(faq-hypertext)CHILD(2,*)(6,*)`, ce qui signifie le sixième sous-objet dans le deuxième sous-objet après l'élément dont l'ID est `"faq-hypertext"`.

Compter les objets depuis le début de cette question dans la version SGML (`faq.sgml`) (qui a l'ID `"faq-hypertext"`) :

1. le titre de la question :

```
<SECT2 ID="faq-hypertext">
<TITLE>Comment XML modifiera-t-il ... ?</TITLE>
```

2. le deuxième paragraphe :
   (a) la donnée textuelle depuis le début du paragraphe jusqu'au premier élément de balisage :
   ```
   <PARA>Les documents
   ```
   (b) l'élément de balisage :
   ```
   <ULINK URL="http://www.w3.org/TR/WD-xlink">XML Linking
          Specification (XLink)</ULINK>
   ```
   (c) l'élément de texte :
   ```
   et
   ```
   (d) l'élément de balisage :
   ```
   <ULINK URL="http://www.w3.org/TR/WD-xptr">XML Extended
          Pointer Specification (XPointer)</ULINK>
   ```
   (e) le paragraphe suivant de donnée textuelle :
   ```
   contiennent des spécifications...
    ...  Extended Pointer (
   ```
   (f) et l'élément de balisage suivant :
   ```
     <LINK LINKEND="tei-link">XPointer</LINK>
   ```

Si vous lisez ce fichier avec *Panorama* ou *MultiDoc Pro* vous pouvez cliquer sur le bouton de renvoi en surbrillance au début de la phrase d'exemple. Les emplacements de tous les liens qui s'y réfèrent s'afficheront alors en *EPN (Extended Pointer Notation)*, y compris le mot « XPointer » mentionné. Réaliser ceci avec un navigateur HTML n'est pas très significatif, le navigateur HTML ne supportant pas les liaisons bidirectionnelles ou les EPN. David Megginson a développé une fonction supplémentaire pour *Emacs/psgml* qui déduira un XPointer pour tout emplacement dans un fichier SGML ou XML.

C.13 *Puis-je faire des maths avec XML ?*

Oui, si le type de document (page 298) que vous utilisez est prévu pour les mathématiques. La communauté des utilisateurs de mathématiques développe actuellement un logiciel et il existe une proposition MathML au W3C (`http://www.w3.org/Math/`), qui est une application XML native. Il serait également possible de créer des fragments XML à partir de l'application obsolète HTML3, HTML Pro (`http://www.arbornet.org/~silmaril/dtds/html/htmlpro.html`), ISO 12083 Math (`http://www.oasis-open.org/cover/gen-apps.html#iso12083DTDs`), ou OpenMath (`http://www.can.nl/~abbott/OpenMath/`), ou d'une application de votre propre conception. Il existe déjà des navigateurs qui affichent des maths intégrées à SGML (ex. : *DynaText*, *Panorama*, *Multidoc Pro*).

La complexité de ces expressions mathématiques peut varier d'expressions telles que x_i à des équations dans le texte telles que $E = mc\hat{}2$ voire pour afficher en pleine-page des équations telles que :

$$b_4 \sqrt[i]{u} \frac{ru}{16} qt\pi$$

Si vous utilisez un navigateur HTML pour lire ceci, les équations ci-dessus n'apparaissent peut-être pas correctement. Le plugin *Techexplorer* d'IBM peut être utilisé avec les navigateurs HTML courants pour afficher les maths de TeX. Le navigateur *Amaya* sur le W3C (`http://www.w3.org/Amaya/`) dispose d'un affichage MathML expérimental.

C.14 *Comment XML gère-t-il les méta-données ?*

Parce que XML permet de définir votre propre langage de balisage, vous pouvez utiliser pleinement les caractéristiques d'hypertexte étendu (voir la question sur les liens (page 300)) de XML pour stocker ou lier des méta-données dans n'importe quel format (p. ex.ISO 11179, Dublin Core, Warwick Framework (`http://purl.oclc.org/metadata/dublin_core/`), Resource Description Framework (RDF) (`http://www.dstc.edu.au/RDU/RDF/`) et Platform for Internet Content Selection (PICS) (`http://www.w3.org/PICS/`)).

Étant une architecture et non une application, XML ne comporte aucun élément prédéfini. Ce n'est donc pas le rôle de XML de spécifier si et comment les auteurs devraient ou ne devraient pas implémenter des méta-données. Vous êtes donc libre d'utiliser la méthode qui vous convient, des simples attributs à l'inclusion de tous les registres de méta-données de Dublin Core/Warwick Framework. Les éditeurs de navigateurs pourraient également proposer leurs propres recommandations en matière d'architecture ou leurs méthodes.

C.15 *Puis-je utiliser Java, ActiveX, etc. dans des fichiers XML ?*

Ceci dépend des caractéristiques propres au navigateur. XML a pour fonction de décrire l'information ; les langages de script et les langages pour fonctionnalités intégrées sont les logiciels qui permettent à l'information d'être traitée par l'utilisateur final.

XML lui-même permet de définir le balisage nécessaire à l'implémentation de langages de script : en tant que standard neutre il n'encourage ni ne décourage leur utilisation et ne favorise pas un langage plutôt qu'un autre ; la porte est donc ouverte. Des développements sont en cours : voir les suggestions de John Tigue pour la normalisation de l'API pour Java (`http://www.datachannel.com/ChannelWorld/XML/dev/`) à propos d'XML.

Les langages de script *sont* donnés dans le cadre d'une proposition pour un Extensible Style Language (XSL) (`http://www.w3.org/TR/NOTE-XSL-970910`) (voir la question sur les feuilles de style (page 303)).

C.16 *Puis-je utiliser Java pour créer ou gérer des fichiers XML ?*

Oui, n'importe quel langage de programmation peut être utilisé pour sortir des données de n'importe quel source au format XML. Il y a un nombre croissant de produits amont ou aval pour environnements de programmation et de gestion de données qui permettent d'automatiser ceci.

— Mark Watson écrit, article `344c3443.4494773@news.infonex.net`: « J'ai affiché les spécifs d'une boîte à outils Java pour la création de dcouments XML à partir de requètes de bases de données relationelles et pour les sauvegardes/restaurations de documents XML sur/depuis des fichiers locaux, et pour le transport à travers des *sockets* RMI, CORBA, IIOP. Les spécifs sont à `www.markwatson.com/XMLdb_0_1.htm`. »
— Il y a une série de tutoriels Java (avec code source et explications) disponibles à `http://developerlife.com`. Ces tutoriels montrent aux développeurs Java2 comment utiliser les parseurs IBM, Sun et OpenXML pour écrire des programmes Java qui utilisent XML.

C.17 *Comment contrôler la présentation ?*

L'utilisation d'une feuille de style est obligatoire en XML. Certains navigateurs offrent peut-être de simples styles par défaut pour des éléments courants comme `<PARA>` ou `<LIST>` contenant des `<ITEM>` ; mais en général une feuille de style permet à l'auteur de mieux contrôler la mise en page. Or, comme dans tout système où les fichiers peuvent être visualisés au hasard par des utilisateurs arbitraires, l'auteur ne peut pas connaître les ressources (polices par exemple) disponibles sur le système de l'utilisateur ; des précautions sont donc de rigueur.

— La norme internationale de feuilles de style pour les documents SGML est DSSSL, Document Style and Semantics Specification Language (`http://www.oasis-open.org/cover/dsssl.html`) (ISO 10179 (`http://www.iso.ch/`)). Elle offre des langages du genre de Scheme pour les feuilles de style et la conversion de documents et est implémenté dans le logiciel de formatage *Jade* (`http://www.jclark.com/jade/`).
— Les CSS (Cascading Stylesheet Specifications) (`http://www.w3.org/Style/css`) offrent une syntaxe simple pour attribuer des styles aux éléments et ont été implémentées en partie dans certains navigateurs HTML.
— Une nouvelle ébauche de (XSL) Extensible Style Language (`http://www.w3.org/TR/WD-xsl`) a été conçu pour une utilisation spécifique avec XML . Elle utilise la syntaxe XML (une feuille de style XSL est en fait un fichier XML) mais combine les fonctions de formatage de DSSSL et de CSS (HTML) et bénéficie déjà du soutien de nombreux distributeurs importants.

Le XML Styler (`http://www.arbortext.com/xmlstyler/`) expérimental d'Arbortext donne des détails sur la façon de l'utiliser avec XML. Il faut aussi les commandes ActiveX et XSL codebase (`http://www.microsoft.com/xml/xsl/msxsl.cab`).

Plusieurs systèmes et implémentations de feuilles de style propriétaires pré-existantes sont aussi disponibles. La plupart sont largement intégrés à la communauté de documentation technique (et par conséquent supportés par un ou plusieurs produits) :

— chez Inso Corps (dont la compagnie originelle, EBT, a inventé la majorité des concepts de la technologie actuelle des feuilles de style), les navigateur et serveur *DynaText* et *DynaWeb* ;
— la DTD pour feuille de style Synex telle qu'utilisée dans *Panorama* et *MultiDoc Pro* ;
— la norme militaire américaine FOSI *(Formatted Output Specification Instance)* implémentée dans *ADEPTůEditor* d'Arbortext et ailleurs;
— *Author/Editor* de SoftQuad utilise un autre système de feuille de style contrôlable par l'utilisateur.

La plupart des fournisseurs de navigateurs et de serveurs semblent migrer vers XML mais, vu la grande base de systèmes déjà installés, cela ne se fera sans doute pas très vite.

C.18 *Comment utiliser les graphiques en XML ?*

Les graphiques sont simplement des liens vers une image plutôt que vers un texte, ils peuvent donc être créés de n'importe quelle manière supportée par les spécifications XLink et XPointer (voir une question précédente (page 300)), même en utilisant une syntaxe similaire aux images HTML existantes. On peut aussi en faire en utilisant le mécanisme intégré d'XML `NOTATION`

et ENTITY de la meme façon qu'en SGML standard. Cependant, les spécifications de liaison offrent un meilleur contrôle sur les liens de traversée et leur activation. Un auteur peut donc, par exemple, choisir de faire apparaître une image lors du chargement de la page, sur simple clic de l'utilisateur, ou dans une autre fenêtre, *sans avoir à modifier le script*. Il revient aux concepteurs de navigateurs de choisir les formats de fichiers graphiques qui seront supportés : XML ne comporte aucune restriction. L'utilisation des formats GIF, JPG, TIFF et CGM au minimum seraient logiques : il y a certaines tendances à la création d'une norme de graphiques vectoriels pour réseau (voir paragraphe suivant).

Peter Murray-Rust précise : « Les formats GIF et JPEG traitent des bitmaps (représentation en pixels des images). Les graphiques vectoriels (échelonnables) sont discutés par l'activité graphique du W3C (voir http://www.w3.org/Graphics/Activity). Quand un consensus aboutira, il sera alors possible de transmettre une représentation graphique *au sein même du fichier XML*. Ceci signifie, pour la plupart des objets graphiques, un temps de téléchargement considérablement réduit et une mise à l'échelle sans perte de définition. »

Note On ne peut pas emboîter un fichier graphique (pas plus qu'un autre fichier binaire [non textuel]) directement dans un fichier XML car n'importe quel octet ressemblant à une marque serait mal interprété. Il faut s'y référer par des liens (voir ci-dessous).

Bob DuCharme ajoute : « Toutes les données dans une entité de document XML doivent être du XML parsable. Vous pouvez définir une entité externe soit en tant qu'entité parsée (XML parsable) soit en tant qu'entité non parsée (n'importe quoi d'autre). Les entités non parsées peuvent être utilisées pour les fichiers images, les fichiers sons, les fichiers vidéo ou tout ce que vous voulez. Elles ne peuvent être appelées qu'à partir d'un document en tant que valeur d'un attribut (de la même façon qu'une image Bitmap d'une page web en HTML est l'attribut src de l'élément <img>) et n'ont pas en tant que partie du document courant. Dans un document XML, cet attribut doit être déclaré de type ENTITY et la déclaration d'entité doit spécifier une NOTATION déclarée. En effet, si l'entité n'est pas du XML, le processeur XML doit savoir ce dont il s'agit. Par exemple, dans le document suivant, l'entité colliepic est déclarée comme ayant une notation JPEG et est utilisée en tant que valeur de l'attribut picfile de l'élément vide toutou.

```
<?xml version="1.0"?>
    <!DOCTYPE toutou [
    <!NOTATION JPEG SYSTEM "Joint Photographic Experts Group">
    <!ENTITY colliepic SYSTEM "lassie.jpg" NDATA JPEG>
    <!ELEMENT toutou EMPTY>
    <!ATTLIST toutou picfile ENTITY #REQUIRED>
    ]>
    <toutou picfile="colliepic"/>
```

Les spécifications de liaison XLink et XPointer décrivent d'autres manières de pointer vers un fichier non-XML tel qu'un graphique. Ces spécifications permettent de mieux contrôler la position, le traitement et l'apparence d'une entité externe dans un document XML. »

On pourrait toutefois inclure un fichier binaire transformé et codé en texte comme une section CDATA, en utilisant quelque chose comme UUencode avec des marques] et > supprimées du jeu de façon qu'elles ne puissent ni être utilisées, ni mal interprétées.

D. Développeurs et implementeurs (y compris les webmasters et les responsables serveurs)

D.1 *Où se trouve la spécification ?*

La spécification se trouve à `http://www.w3.org/TR/REC-xml`. Elle inclut l'EBNF. Il existe également des versions en japonais (`http://www.fxis.co.jp/DMS/sgml/xml/`) ; en espagnol (`http://www.ucc.ie/xml/faq-es.html`) ; en coréen (`http://xml.t2000.co.kr/faq/index.html`) et une version annotée Java-isée à l'adresse (`http://www.xml.com/axml/testaxml.htm`).

Eve Maler a lancé la DTD (`http://www.w3.org/XML/1998/06/xmlspec-19980910.dtd`) et la documentation (`http://www.w3.org/XML/1998/06/xmlspec-report-19980910.htm`) propres à la spec : il s'agit d'une nouvelle version utilisée pour coder les spécifications XML, XLink, XPointer, DOM, etc. Attention, cette version n'est plus compatible avec la version qu'utilise XML 1.0. Merci d'envoyer vos commentaires et questions à Eve (`<elm@arbortext.com>`).

D.2 *Que signifient les termes « sans DTD », « valide » et « bien formé » ?*

Le SGML intégral utilise une définition de type de document (DTD) pour décrire le balisage (éléments) disponible dans tout type de document spécifique. Cependant la conception d'une DTD peut s'avérer complexe et peu évidente. XML a donc été conçu pour être utilisé avec ou sans DTD. Opérer sans DTD permet d'inventer un balisage sans avoir à le définir de façon formelle, avec le risque de perdre le contrôle automatique de la structuration de documents additionnels de même type.

En fait un fichier sans DTD « définit » son propre balisage, de manière informelle, par l'existence et la localisation d'éléments là où ils sont créés. Mais lorsqu'une application XML, telle qu'un navigateur, rencontre un fichier sans DTD, elle doit être capable d'en comprendre la structure *à la lecture* car aucune DTD ne l'informe de ce qu'elle doit faire, aussi quelques modifications ont dues être apportées.

Par exemple, l'élément HTML `<IMG>` est défini comme « `EMPTY` » : il n'a pas de balise de fin. Une application XML lisant un fichier *sans DTD* et rencontrant `<IMG>` n'aurait aucun moyen de savoir si elle doit ou non attendre une balise de fin, aussi le concept « bien formé » a-t-il été introduit. Ceci rend complètement univoque le début et la fin de chaque élément et l'occurrence d'éléments `EMPTY` .

D.2.1 Les documents « bien formés »

Tous les documents XML, sans DTD ou valides, *doivent* être bien formés :

— Si aucune DTD n'est utilisée, le document doit démarrer avec une Déclaration de document autonome *(Standalone Document Declaration, SDD)* :

```
<?xml version="1.0" standalone="yes"?>
<foo>
  <bar>...<blort/>...</bar>
</foo>
```

David Brownell signale : « du XML "seulement" bien formé n'a pas du tout *besoin* d'utiliser de *Standalone Document Declaration.* De telle déclarations sont là pour per-

mettre certaines accélérations quand on traite des documents en ignorant les entités paramètres externes – en gros, on ne peut pas compter sur les déclarations externes dans des documents indépendants. Les types concernés sont les entités et les attributs. Les documents indépendants ne doivent pas demander de valeurs d'attribut par défaut ou normales, sinon ils sont invalides. »

— Toutes les balises doivent être équilibrées : tous les éléments qui pourraient contenir des données textuelles doivent avoir une balise de début et une balise de fin (l'omission est interdite sauf pour les éléments vides (voir le deuxième paragraphe qui suit) ;
— Toutes les valeurs d'attribut doivent être entre apostrophes (le caractère guillemet simple [apostrophe `'`] peut être utilisé si la valeur contient un caractère double `"` et *vice versa*) : si vous avez besoin des deux signes, utilisez `&apos;` et `&quot;`
— Toute balise d'éléments `EMPTY` (ex. : ceux qui n'ont pas de balise de fin comme les `<IMG>`, `<HR>` et `<BR>` et autres) d'HTML doit soit se terminer par « `/` » ; sinon il faut ajouter une véritable balise de fin aux éléments qui deviennent alors non-`EMPTY`.
Exemple : `<BR>` devient soit `<BR/>` soit `<BR></BR>`.
— Il ne doit pas y avoir de caractère de balisage isolé (`<` ou `&`) dans vos données textuelles (c'est-à-dire qu'ils doivent apparaître comme `&lt;` et `&amp;`) et la séquence `]]>` doit être donnée sous la forme `]][amp ]gt;` si elle n'est pas à la fin d'une section `CDATA` ;
— Les éléments doivent s'imbriquer correctement les uns dans les autres (pas de chevauchement de balisage, de même qu'avec SGML) ;
— Les fichiers bien-formés sans DTD peuvent utiliser des attributs pour tout élément, mais tous les attributs doivent être de type CDATA par défaut.

Les fichiers XML sans DTD sont supposés avoir les éléments `&lt;`, `&gt;`, `&apos;`, `&quot;` et `&amp;` prédéfinis, qui sont donc utilisables même sans DTD. Les fichiers XML valides doivent les déclarer explicitement s'ils les utilisent. Si on veut utiliser plus que ces cinq entités caractères par défaut, mais si on veut éviter d'avoir à écrire une DTD complète, il est possible de ne déclarer que les entités caractères dans le sous-ensemble interne d'un autre fichier XML indépendant (merci à Richard Lander pour cette information).

```
<?xml version="1.0" standalone="yes"?>
<!doctype example [
<!entity nbsp "&#032;">
]>
<example>Three   blanks.</example>
```

D.2.2 XML valide

Les fichiers XML valides sont ceux qui possèdent une Définition de Type de Document (DTD) (page 308), comme toutes les applications SGML (page 286) et qui y adhèrent. Ils doivent également être bien-formés (page 305).

Un fichier valide, comme tout autre fichier SGML, commence par une Déclaration de Type de Document, qui peut être précédée d'une déclaration XML facultative :

```
<?xml version="1.0"?>
<!DOCTYPE advert SYSTEM "http://www.foo.org/ad.dtd">
```

```
<advert>
  <headline>...<pic/>...</headline>
  <text>...</text>
</advert>
```

La Spécification XML (page 305) définit une déclaration SGML pour XML qui est définie pour toutes les instances (la déclaration ne figure plus dans le texte de la spécification et se trouve maintenant dans un document à part (`http://www.w3.org/TR/NOTE-sgml-xml-971215#null`)). Une version XML (page 308) de la DTD spécifiée doit être accessible par le processeur XML, soit localement (c'est-à-dire que l'utilisateur a déjà une copie sur son disque), ou via le réseau. Vous pouvez le indiquer une DTD par une URL dans l'identificateur de système (comme dans l'exemple précédent). Il est possible (certains diraient préférable) de fournir un identificateur public formel (`http://www.ucc.ie/cgi-bin/public`), mais s'il est utilisé, il doit *précéder* l'identificateur de système, qui doit toujours être donné (et seul le mot clé `PUBLIC` est utilisé).

```
<!DOCTYPE advert PUBLIC "-//Foo, Inc//DTD Advertisements//EN"
  "http://www.foo.org/ad.dtd">
```

Les valeurs par défaut pour les autres attributs de la déclaration XML sont `version="1.0"` et `encoding="UTF-8"`.

D.3 *Que dois-je utiliser dans ma DTD : des attributs ou des éléments ?*

Il n'y a pas de réponse unique à cette question : cela dépend beaucoup de la finalité de votre type de document. Les deux extrêmes s'expliquent mieux par des exemples :

— la pratique « traditionnelle » est de mettre le texte « réel » (celui que l'on veut imprimer) comme des données de type caractère et de garder les méta-données (comme les numéros de ligne) en attributs, où elles peuvent être plus facilement isolées pour analyse ou pour des traitements spéciaux comme la mise en marge ou dans une *mousevoer* :

```
<l stance="oui" nb="13">Et je m'en vais</l>
```

— sauf du point de vue système, il n'y a rien de « faux » à garder les données d'une autre façon, notamment lorsque le volume des données textuelles est plutôt petit à chaque occasion :

```
<l stance="oui" texte="Et je m'en vais">13</l>
```

Cela dépend donc pour beaucoup de ce que vous voulez faire des informations et de quelles parties sont le plus facilemement accessibles par chaque méthode. Une règle de bon sens pour les documents textuels usuels est que si l'on supprime toutes les balises, le texte dépouillé doit rester lisible et utilisable, même si c'est difficile. Mais, cependant, pour les sorties de bases de données ou autres documents secondaires, « lire » n'a plus de sens et il est alors tout à fait immaginable d'avoir des documents où *toutes* les données sont en attributs et où les documents ne contiennent pas du tout de caractère dans le modèle de contenu. Voir `http://www.oasis-open.org/cover/elementsAndAttrs.html` pour plus de détails.

D.4 *Quoi d'autre a changé de SGML à XML ?*

Les principales modifications portent sur ce que vous pouvez écrire dans une Définition de Type de Document (DTD). Pour simplifier la syntaxe et simplifier l'écriture de logiciels de traitement, de nombreuses options de déclaration de balisage SGML ont été supprimées (voir la référence à la liste des caractéristiques omises (page 288)).

Un délimiteur supplémentaire (les deux-points) est autorisé dans les noms et est utilisé dans les expériences avec les espaces de nom (ils permettent aux DTD de distinguer la source de l'élément, le propriétaire ou l'application). Les deux-points peuvent n'apparaître qu'au milieu d'un nom et pas au début ou à la fin. Des travaux en cours visent à définir comment ils peuvent être déclarés et référencés en utilisant le balisage des éléments et des attributs.

D.5 *Quels logiciels XML puis-je utiliser aujourd'hui ?*

Les précisions concernant cette question évoluent trop rapidement ; elles n'apparaissent donc pas dans cette FAQ : voir les pages XML à l'adresse suivante `http://www.oasis-open.org/cover/xml.html`.

Pour un guide détaillé d'exemples de programmes SGML et XML et les concepts sous-jacents, voir le livre de l'auteur de cette FAQ : *Understanding SGML and XML Tools* (`http://imbolc.ucc.ie/~pflynn/books/sgmltools.html`) (Kluwer, 1998, 0-7923-8169-6).

Concernant les navigateurs, voir la question sur les navigateurs XML (page 292) et les précisions de la `liste de diffusion ml-dev (page 290)` pour les développeurs de logiciels. Bert Bos maintient une liste de quelques développements XML (`http://www.w3.org/XML/notes.html`) en bison, flex, perl et Python.

Des informations pour les développeurs de XML chinois peuvent être trouvées dans la page web chinoise *WML Now!* de l'Academia Sinica : `http://www.ascc.net/xml/` (`http://www.ascc.net/xml/`) Ce site propose une FAQ et des fichiers de tests.

D.6 *Dois-je changer mes serveurs pour travailler avec XML ?*

Seulement pour servir les fichiers `.xml` au type MIME correct (`application/xml`, voir RFC2376 (`ftp://ftp.isi.edu/in-notes/rfc2376.txt`)), donc pour servir les documents XML il suffit de modifier le fichier `mime-types` (ou son équivalent) et d'ajouter la ligne :

```
application/xml        xml XML
```

Dans certains serveurs (ex. : *Apache*), les utilisateurs peuvent modifier le type MIME pour les types de fichiers spécifiques à partir de leurs propres répertoires en utilisant un fichier `.htaccess`. Le type de contenu MIME `text/xml` peut seulement être appliqué aux fichiers en ASCII pur (ISO 646 IRV) à cause d'une restriction du jeu de caractères dans le RFC ; en usage normal, aller à `application/xml`.

XML étant conçu pour supporter les feuilles de style et l'hyperliaison sophistiquée, les fichiers XML seront accompagnés de fichiers auxiliaires, tout comme les fichiers SGML : DTD, fichiers d'entité, catalogues, feuilles de style, etc. qui nécessiteront d'autres entrées type de contenu MIME, comme `text/css` pour les feuilles de style CSS. Le XUA *(XML User Agent)*, qui est l'un des développements envisagés du XML WG, pourrait fournir un système pour englober les documents XML et les styles XSL en un seul message.

Si vous exécutez des scripts générant du HTML que vous voulez faire fonctionner en XML, vous devrez les modifier pour produire le type de document pertinent.

D.7 *Puis-je encore utiliser les `INCLUDE` côté serveur ?*

Oui, dans la mesure où le produit généré s'insère dans un fichier conforme au XML (c'est-à-dire valide (page 306) ou simplement bien formé (page 305)).

D.8 *Puis-je (ainsi que mes auteurs) encore utiliser les `INCLUDE` côté clients ?*

C'est la même règle qu'avec les `INCLUDE` serveur (page 309) qui s'applique ici. Vous devez donc vous assurer qu'aucun code intégré qui passe dans un moteur tiers (ex. : requêtes *SDQL*, `codes` *Java*, requêtes *LiveWire*, contenu « streamed », etc.) ne contient de caractère qui pourrait être interprété comme du balisage XML (c'est-à-dire ni guillemets-chevrons ni esperluète) : utilisez une section `CDATA` afin d'éviter que votre application XML ne parse le code intégré, ou utilisez à la place les références d'entité de caractère standard `<`, `>` et `&`.

D.9 *J'essaie de comprendre la spécif XML : pourquoi la terminologie de SGML (et de XML) est-elle si compliquée ?*

Une terminologie précise est nécessaire à une bonne implémentation de XML. L'objectif numéro 8 *(design goal 8)* des spécifications nous dit que « la définition de XML doit être formelle et concise ». Pour décrire XML en termes formels, sa spécification utilise le langage concis de l'informatique, ce qui déroute les non informaticiens : en effet, il utilise des mots anglais bien connus mais avec un sens bien précis, parfois éloigné de leur sens habituel – par exemple : *grammar* (grammaire), *production*, *token* (marque) ou *terminal* (terminal).

La spécification explique rarement ces termes à cause de l'autre aspect de cette définition : la spécification doit être concise. Elle ne répète pas les explciations que l'on peut trouver ailleurs. Ça veut dire que pour dominer les subtitlités des spécifications, il vous faut être compétent en informatique et en sgml !

Une terminologie inappropriée dans les spécifications est source d'interprétations erronnées. Les normes formelles doivent donc être rédigées avec une terminologie formelle. Cette FAQ n'est pas un document formel et le lecteur avisé aura sans doute remarqué qu'on y fait référence à des « noms d'éléments » et non pas à des « noms de types d'éléments », formule qui est certes plus correcte mais mal comprise et moins répandue.

Les novices du SGML (page 286) pourront peut-être lire le chapitre de TEI (`http://www.uic.edu/orgs/tei/`) intitulé *Gentle Introduction to SGML (`http://etext.virginia.edu/bin/tei-tocs?div=DIV1;id=SG`)*.

Merci à Bod DuCharme pour quelques suggestions et quelques extraits de son livre sur la spécif XML (`http://www.snee.com/bob/xmlann`).

D.10 *Existe-t-il un kit API de développeurs pour XML ?*

Plusieurs kits sont disponibles ou en cours de développement. Des précisions, ainsi que des informations sur les autres logiciels XML, sont disponibles sur lespages Web SGML/XML (`http://www.oasis-open.org/cover/sgml-xml.html`) .

Les gros moteurs de développement d'application et de conversion SGML tels que *Balise*, *Omnimark* et *SGMLC* développent tous des versions XML. Des précisions sur les logiciels SGML de

tous types sont disponibles sur les pages Web SGML/XML (`http://www.oasis-open.org/cover/sgml-xml.html`).

D.11 *Comment XML réagit-il avec les DOM ?*

Le *Document Object Model* (DOM) (`http://www.w3.org/TR/PR-DOM-Level-1`) fournit des API abstraites pour la construction, l'accès et la manipulation de documents XML et HTML. Une « réalisation » du DOM dans un langage de programmation donné offre une API concrète.

D.12 *Existe-t-il des tests de conformité pour les processeurs XML ?*

James Clark propose plusieurs tests pour parsers XML à l'adresse `http://www.jclark.com/xml/`. Il propose également un test de conformité.

D.13 *Comment inclure une DTD (ou un morceau) dans une autre ?*

Exactement de la même façon que pour SGML normal. Vous déclarez d'abord l'entité que vous voulez inclure, puis vous la référencez par son nom :

```
<!ENTITY % mylists PUBLIC
         "-//Foo, Inc//ENTITIES Common list structures//EN"
         "dtds/listfrag.ent">
...
%mylists;
```

Normalement de telles déclarations figurent toutes au début du fichier DTD principal, où elles peuvent être gérées ; mais ce n'est pas essentiel. Il faut utiliser une syntaxe d'entité paramètre (signe pourcent) parce que le fichier sera intégré au moment de la compilation de la DTD et non pas lorsque l'instance du document sera parsée.

Avec XML, il est obligatoire de spécifier une URL pour toutes les références à un fichier externe : les règles classiques pour supprimer la référence à une URL s'appliquent (méthodes, serveurs et répertoires identiques à ceux du document les contenant). L'URL peut être indiquée comme un identificateur système seul :

```
<!ENTITY mydtd SYSTEM "http://www.foo.bar/~blort/my.dtd">
```

ou comme le deuxième paramètre d'un identificateur public formel (`http://www.ucc.ie/cgi-bin/public`) comme dans l'exemple précédent (page 306).

D.14 *Je possède déjà des DTD en SGML : comment les convertir pour les utiliser avec XML ?*

De nombreux projets sont en cours pour convertir les DTD SGML courantes ou connues au format XML (Patrice Bonhomme par exemple travaille sur une version XML non officielle de la DTD TEI Lite : les détails sont discutés sur la liste de diffusion TEI-L).

La checklist suivante vous est donnée grâce à l'aimable autorisation de Seán McGrath (auteur de *XML By Example*, Prentice Hall, 1998) *[l'italique est mis par l'auteur de la FAQ]* :

1. Aucun équivalent de la déclaration SGML. Les mots clés, jeux de caractères, *etc.* sont donc essentiellement fixes ;

2. La minimisation des balises est interdite, donc `<!ELEMENT x - O (A,B)>` devient `<!ELEMENT X (A,B)>` et `<!ELEMENT x - O EMPTY>` devient `<!ELEMENT X EMPTY>` ;
3. `#PCDATA` ne doit apparaître qu'à l'extrême gauche d'un modèle `OR` ; ex. `<!ELEMENT x (A|B|#PCDATA|C)>` devient `<!ELEMENT x (#PCDATA|A|B|C)>` et `<!ELEMENT x (A,#PCDATA)>` est illégal ;
4. Aucun élément `CDATA`, `RCDATA` *[declared content]* ;
5. Certains types d'attributs SGML ne sont pas autorisés en XML ; ex. `NUTOKEN`. Il n'y a pas non plus d'attributs `NOTATION` (attributs de données) ;
6. Certains défauts d'attributs SGML ne sont pas autorisés en XML ; ex. `CONREF` ;
7. Les commentaires et les déclarations ne peuvent pas figurer sur la même ligne ; ex. *[légal en SGML]* :

   ```
   <!ELEMENT x (A,B) -- commentaire dans une déclaration SGML -->
   ```

8. Tout un ensemble de fonctions facultatives de SGML a disparu avec XML : *a*) toutes les formes de minimisation de balises (`OMITTAG`, `DATATAG`, `SHORTREF`, etc) ; *b*) les Link Process Definitions ; *c*) les DTD multiples dans un même document, et bien d'autres : voir la question sur les points de SGML n'apparaissant pas dans XML (page 288) pour en obtenir la liste complète ;
9. Dernier point, et non le moindre, `CONCUR` ! Il existe d'énormes différences entre la portion de sous-ensemble interne et externe d'une DTD en XML : *a*) les sections ne peuvent apparaître que dans un sous-ensemble externe ; *b*) les entités paramètres doivent être utilisées pour remplacer des déclarations entières dans la portion d'un sous-ensemble interne d'une DTD ; ex. ce qui suit n'est pas valide en XML :

   ```
   <!DOCTYPE x [
           <!ENTITY % modelx "(A|B)*">
           <!ELEMENT x %modelx;>
           ]>
           <x></x>
   ```

D.15 *Qu'en est-il de XML et de l'EDI ?*

L'échange électronique de documents est utilisé depuis de nombreuses années pour échanger des documents entre partenaires commerciaux dans le cadre d'une transaction. Ceci a nécessité des logiciels propriétaires spécifiques, mais des efforts sont réalisés aujourd'hui pour permettre aux données EDI de circuler dans XML. Des précisions sur les développements en cours sont disponibles à l'adresse `http://www.xmledi.com/`, ainsi que des directives à l'adresse `http://www.geocities.com/WallStreet/Floor/5815/guide.htm`.