12 Juillet 2006
Révision : 5 Octobre 2006

XML et les formats de traitement de texte et de mise en page

Patrick Peccatte – Soft Experience

Le futur de l'industrie des médias se construit en ce moment autour de XML. Ainsi, dans le domaine de l'écrit, tous les traitements de texte et outils de mise en page actuels proposent des fonctionnalités XML diverses qui doivent être comparées et évaluées. Il ne suffit pas en effet de lire la mention "XML" dans la description d'un produit pour connaître ses aptitudes véritables en la matière; il existe par exemple de grandes différences en terme de préservation des contenus, d'interopérabilité et de potentialités, entre un logiciel permettant seulement d'importer certains types de documents comprenant un balisage *ad hoc* et un autre qui repose sur un format de document nativement XML.

Les fonctionnalités XML relatives aux traitements de textes et logiciels de mise en page sont nombreuses : concordance entre styles et balises, entre éléments de mise en page et balises ; méthodes de transformation et d'affichage ; support de DTD, de W3C Schémas ; possibilité de validation ; codage des caractères et support des entités ; flexibilité des dialectes XML supportés ; possibilité de structurations personnalisées ; ou plus avant encore dans les aspects techniques : architecture des formats et API de programmation permettant de les utiliser. L'analyse exhaustive de ces caractéristiques XML dépasse largement le cadre de notre étude. L'objet de cet article est plutôt de décrire les plus importantes d'entre elles en nous intéressant à différentes techniques successivement apparues dans ce domaine pour "travailler des contenus structurés".

Notre grille d'analyse est ainsi orientée vers les formats et la nature des structurations supportées. Ces deux aspects essentiels sont figurés dans le tableau, situé à la fin de l'article, dans lequel

- l'axe horizontal *Ouverture XML*, décrit dans la première partie, distingue différentes étapes dans la façon de travailler en XML jusqu'à l'émergence de véritables formats XML de documents ;
- l'axe vertical *Structuration XML* que nous détaillerons dans la seconde partie s'intéresse plus particulièrement aux possibilités de structurations personnalisées ou orientées métiers.

Nous comparerons donc selon ces deux aspects les caractéristiques XML qui sont présentes dans des outils bien connus : Word et OpenOffice pour les traitements de texte, XPress et InDesign pour les logiciels de mise en page ; nous évoquerons aussi quelques outils

importants du monde de l'édition et plusieurs systèmes éditoriaux actuellement proposés par différents éditeurs[1].

Nous signalerons également au passage quelques caractéristiques des formats XML des traitements de texte généralistes qui pourraient s'avérer importantes à l'avenir pour les applications professionnelles du secteur de la presse.

1. L'ouverture progressive vers XML

1.1. XML est apparu en 1998 et son champ d'application s'est étendu depuis très rapidement jusqu'à toucher pratiquement tous les secteurs de l'informatique. Cependant, il y a peu de temps encore, les outils de bureautique et de mise en page ignoraient tout des techniques XML. L'ouverture s'est réalisée progressivement dans ce domaine, et avec OpenOffice 1.0 et InDesign 2 en 2002 la situation a beaucoup évolué. Parallèlement, dans le secteur de l'édition technique et dans celui des systèmes éditoriaux, plusieurs produits sont devenus comparables aux traitements de texte généralistes tout en exploitant l'avantage fondamental apporté par XML : la séparation du contenu et de ses diverses présentations ou utilisations. Mais avant cela, les besoins d'importer et surtout d'exporter des contenus en XML ont conduit au développement de plusieurs solutions.

La première de ces solutions consiste à sauvegarder un document depuis un format propriétaire vers un format pivot - en RTF (*Rich Text Format*) la plupart du temps pour les traitements de texte - puis à transformer ce format pivot en XML à l'aide d'un convertisseur spécialisé. Quelques produits, souvent développés initialement dans le monde SGML et adaptés ensuite à XML, comme Tétrasys Mosca ou Logictran R2Net, utilisent cette technique. Ils sont surtout déployés dans le domaine de l'édition pour générer des contenus XML à destination de logiciels de mise en forme comme FrameMaker. La plupart du temps cependant il n'est pas possible avec ces convertisseurs d'effectuer le chemin inverse, de convertir vers le format pivot des documents XML corrigés avec l'outil de mise en page puis d'importer dans le traitement de texte ces documents modifiés ; les corrections sur le poste de mise en forme (il y en a toujours…) ne peuvent être remontées via XML sur le poste auteur, ce qui est extrêmement pénalisant pour traiter les publications récurrentes.

Les "moulinettes maison" qui permettent de convertir un vocabulaire XML simple en langage de balisage "XPress tags" destiné à l'import dans Quark XPress entrent également dans cette catégorie.

Cette première solution peut-être qualifiée de premier pas dans l'*Ouverture XML* des traitements de texte et outils de mise en page (colonne 1 de notre tableau).

---

[1] - Il va de soi cependant que cet article ne constitue en aucune façon un classement hiérarchique ou un comparatif des produits cités. L'évaluation sérieuse d'un système doit évidemment prendre en compte bien d'autres paramètres que les seules caractéristiques des formats et des structurations qui sont évoquées ici. Et d'ailleurs, même sous le seul "angle XML", de nombreux aspects très importants concernant son support ou sa présence dans les systèmes éditoriaux ne sont pas abordés ici : base XML *vs* base relationnelle, exploitation de NewsML, de XMP, support de RSS/Atom, de SOAP, etc.
Nous signalons également que les points de vue exprimés ici n'engagent pas l'IFRA mais uniquement l'auteur de l'étude.
Nous remercions vivement Michela Bertagnolli (EidosMedia), Christian Darroy (Unisys), Jean-Baptiste de Vathaire (Cairn), Carlos Echevarria (Protec), Corinne Girardot (CGI Conseil), Olivier Grenet (Wedia), Ilia Kuznetsov (Syntext), Laurent Le Meur (AFP), José Levices (MVS), Luc Potron (Trias), Maurice Sarrasin (Consultas), Eric van der Vlist (Dyomedea) pour leurs remarques constructives et pour avoir bien voulu répondre à nos questions.

1.2. La seconde solution est liée à la possibilité d'écrire une extension (ou un *plug-in*) du logiciel. La conversion du document vers un format XML est alors réalisée de manière plus directe et plus souple que dans le cas précédent. La colonne 2 de notre tableau montre quelques-unes de ces solutions d'export XML direct à l'aide d'une extension : le produit YAWC Pro pour Word ou les extensions XPress bien connues proposées notamment par les sociétés Easypress, Eurocortex/Wedia, Rosebud, Trias, etc. Il n'est pas toujours possible d'importer des documents XML avec ces solutions qui sont ainsi destinées essentiellement à l'exploitation ou à l'archivage des contenus en fin de production. En aucun cas le XML fourni par ces logiciels ne peut être considéré comme un format de travail se substituant peu ou prou au format de document.

1.3. Certains éditeurs ont développé des solutions bidirectionnelles (import et export XML) à l'aide de cette technologie des extensions permettant ainsi une plus grande liberté de flux et d'envisager des applications intégrant XML tout au long de la chaîne de production plurimedia (colonne 3). L'extension Easypress Atomik Roundrip permet par exemple l'aller et retour (*roundtrip*) entre une mise en page XPress et les contenus structurés en XML qui ne sont plus exploités uniquement en fin de chaîne; XML peut alors devenir un véritable format de travail et non plus seulement un sous-produit du *print*, lorsque seul l'export XML est possible.

Ces premières techniques de conversion vers ou depuis XML s'appuient sur une reconnaissance quasi-automatique de la structure éditoriale (titre, sous-titre, texte, etc.) qui reste limitée car les logiciels de base (Word, XPress) ne prévoient pas une réelle structuration globale des documents. La structure reconnue est donc basée uniquement sur la présentation visuelle (identification des blocs et styles pour l'essentiel, parfois complétée par une analyse fine des changements de polices et de corps) et non sur une construction sémantique contrôlée par l'auteur du document. En conséquence, il est souvent nécessaire d'intervenir *a posteriori* sur les conversions réalisées pour obtenir un résultat satisfaisant, notamment quand les articles sont éclatés en plusieurs blocs dans une même page ou pire dans des pages différentes ; l'exploitation des images ou des tableaux est également souvent problématique avec ce genre de solution. Pour les entreprises de presse, il est alors parfois plus intéressant de sous-traiter à des sociétés spécialisées la production des contenus XML à partir des pages montées ; on le voit là encore, XML n'est pas dans ce cas un format de travail mais un dérivé de la production des pages imposé par le marché pour l'archivage ou la syndication. Il correspond aussi à une conception désormais obsolète des médias où l'édition Web est considérée comme résultant de l'édition imprimée.

Dans tous les cas, ces solutions de type 2 ou 3 imposent de travailler rigoureusement avec des gabarits et des feuilles de styles. Remarquons toutefois que l'utilisation de feuilles de styles permet d'utiliser des "styles surchargés", c'est-à-dire que le maquettiste peut modifier entièrement le contenu des styles (police, corps, etc.) lors de la mise en page à la seule réserve de conserver leurs noms intacts pour la structuration lors de l'export ; contrairement à ce que l'on entend parfois du côté des créatifs, un maquettiste peut donc parfaitement rester inventif même avec des contraintes strictes concernant l'utilisation des gabarits et des styles.

1.4. Avec la colonne 4 de notre tableau, nous abordons des solutions qui autorisent directement l'import et l'export de contenus XML sans extensions ou produits tiers. Certains systèmes, notamment lorsqu'ils stockent les documents dans une base de données relationnelle, permettent d'importer et d'exporter directement des articles XML conformes à un balisage propriétaire spécifique. C'est le cas par exemple du système éditorial Protec

Milenium à l'aide de son module Insert.
En règle générale, les formats XML autorisés dans ces systèmes sont assez peu paramétrables : choix limités concernant les noms d'éléments, peu ou pas de possibilités de travailler avec des attributs, avec des entités, peu de possibilités d'imbriquer les balises, etc. De même, les structures manipulées sont assez élémentaires car elles constituent une sorte d'image sous forme XML du schéma relationnel utilisé ; on applique une XSL avant l'import afin de se conformer à la structure attendue par défaut et on applique une autre XSL après l'export pour se conformer à la structure du client. Ces possibilités XML sont réservées à l'ajout de documents externes dans une base de données (articles provenant de correspondants par exemple) et à l'archivage ou la syndication des contenus en fin de production. Le concept de la séparation du contenant et du contenu est atteint dans ce genre de système à l'aide de fonctionnalités avancées d'une base relationnelle couplées aux possibilités d'import et d'export XML.

1.5. La maîtrise de l'importation des documents XML nécessite de disposer de véritables outils paramétrables permettant de décrire une structure (colonne 5). C'est le cas par exemple d'InDesign CS2 à l'aide de son mode d'affichage *Structure* et de sa palette *Balises* qui permettent de représenter certaines structures relativement simples [1]. Il est aussi possible d'importer des descriptions externes de structures (sous forme de DTD pour InDesign ; sous forme de DTD ou de W3C Schémas pour FrameMaker) et de valider les documents XML importés. Plusieurs fonctionnalités comme l'affectation de noms de balises aux blocs, la correspondance entre les noms de balises et les noms de styles et la possibilité de travailler avec un espace de noms pour les styles permettent de construire certaines applications de mise en page qui exploitent réellement les possibilités de XML (catalogues à partir de bases de données par exemple). Toutefois, ces applications imposent des contraintes sur les types de XML supportés, et elles utilisent toujours un format de document binaire et propriétaire et non un véritable format de document XML.

D'autres éditeurs ont choisi de laisser à leurs clients le soin de développer à l'aide d'une "boîte à outils" les fonctionnalités d'import et d'export XML dont ils ont besoin. C'est le cas d'Unisys qui livre avec son système Hermès une API (*Application Programming Interface*) nommée MediaApi permettant de développer l'ensemble des imports/exports souhaités en XML. Là également, le format de document utilisé en natif par le système reste propriétaire (Unisys Newsroom ou InCopy/InDesign).

1.6. Une autre approche consiste pour les éditeurs de logiciels à fournir une API permettant au développeur de travailler avec une représentation XML normalisée du document sous forme d'arbre DOM (*Document Object Model* défini par le W3C). Le logiciel permet alors au programmeur de travailler sur le document en XML, mais uniquement à travers l'API proposée par l'éditeur ; autrement dit, il n'est pas possible de réellement s'affranchir du logiciel en question pour construire des solutions. Malgré son nom qui peut laisser penser qu'il s'agit d'un langage XML, QXML (*QuarkXPress Markup Language*) livré avec la version 7 de Quark XPress entre dans cette catégorie et décrit intégralement en DOM le format de document XPress ; QXML permet le développement d'extensions en Java ou en utilisant les langages disponibles sur .Net ou encore à l'aide de plusieurs langages de script en travaillant avec une représentation DOM du document. Mais le format natif de document reste bel et bien binaire et propriétaire.

La fourniture d'une API DOM par Quark constitue certes un progrès vers l'ouverture et dépasse largement les simples possibilités d'import/export que nous avons examiné jusqu'à maintenant. Mais on doit bien reconnaître, comme le remarque l'un des meilleurs spécialistes

des technologies XML, que l'essentiel de l'intérêt de XML échappe à QXML [2] ; cette solution n'est guère plus satisfaisante que la sérialisation DOM utilisée par son concurrent (cf. paragraphe suivant 1.7). L'esprit de XML est en effet de fournir un vocabulaire permettant en théorie à tout un chacun de produire et exploiter des documents, pas d'imposer une boîte noire fournie par l'éditeur pour accéder aux documents.

1.7. Seuls les formats intégralement et nativement XML permettent enfin une véritable ouverture conforme à cet "esprit XML" que nous évoquions. Cependant, si le format XML n'est pas conçu à l'origine comme un format de document mais uniquement comme un format d'échange entre des applications bien spécifiques, il n'est pratiquement utilisable que par l'éditeur des applications en question et en cela il ne peut être considéré comme véritablement ouvert. Le format Adobe INX (*InDesign Interchange Format*) et son dérivé INCX (*InCopy Interchange Format*) appartiennent à cette catégorie ; ils restent totalement propriétaires quand bien même ils s'expriment en XML.
Un fichier INX est une représentation externe du modèle objet d'un document InDesign qui permet de conserver l'intégralité des informations du document sous forme XML afin de permettre un échange transparent [3]. Ce format prend en charge plusieurs fonctionnalités d'InDesign CS2, dont la compatibilité rétroactive avec la version CS précédente et le flux de production de tâches avec le logiciel de rédaction InCopy CS2. Techniquement, le format INX correspond à une sérialisation XML de l'arbre DOM d'un document InDesign exposé au programmeur pour le *scripting* de l'application. Il est donc par nature totalement lié à l'état de la représentation du document dans l'application ; ainsi, un document INX peut par exemple comporter des données maintenues dans la représentation DOM par des *plug-ins* de l'application. Adobe précise même qu'il n'existe pas de structure définitive des fichiers INX ; il n'existe donc pas de DTD ou de schéma décrivant une telle structure DOM sérialisée. L'éditeur ne considère pas d'ailleurs qu'il s'agisse là d'un format d'archivage à long terme et ne recommande pas la création ou la modification de fichiers INX par des applications tierces. On doit donc constater que XML n'apporte rien ici par rapport à un format binaire propriétaire, si ce n'est qu'il s'exprime en plein texte.

1.8. Le format XML des documents peut éventuellement rester propriétaire[2] tout en étant un véritable format de travail ouvert et documenté (rien n'est caché, tout est public et publié). Il est donc possible d'envisager avec ce type de format des solutions capables de générer et transformer des documents XML indépendamment d'un logiciel propriétaire, et même indépendamment d'un éditeur de logiciels privilégié. C'est le cas du format nommé WordprocessingML de Word 2003 (le binaire *.doc* restant le format par défaut dans cette version de Word), de l'ancien format d'OpenOffice version 1.0, mais aussi du format de page utilisé dans le logiciel libre Scribus.

Le système rédactionnel Wedia Open[3] est un des rares systèmes professionnels à utiliser nativement un format XML propriétaire mais public pour décrire les articles, les pages et en général tous les objets utilisés par le système ; les articles par exemple sont décrits selon un format inspiré de NewsML, le langage XML défini par l'IPTC pour l'échange des informations d'actualité. L'éditeur de texte MVS Dixit utilise également un format XML natif et propriétaire personnalisable par les utilisateurs.

Citons aussi dans cette catégorie certains éditeurs rédactionnels destinés aux correspondants

[2] - L'éditeur peut modifier la spécification d'un format XML *propriétaire* comme il l'entend, ce qui n'est pas le cas lorsque le format est *standardisé* c'est-à-dire contrôlé par un organisme indépendant (voir paragraphe suivant 1.9).

ou aux "petites" rédactions comme Trias TriasEdit, Soft Experience KaliNews et Wedia YA2.

Mais qu'entend-on précisément par *format ouvert de document* ? La Commission Européenne s'est penché sur cette question en 2003 pour ce qui concerne les formats utilisés en bureautique et ses conclusions sont intéressantes. La Commission a en effet mis en place un programme appelé IDA (*Interchange of Data between Administration*) concernant l'"e-gouvernement", c'est-à-dire l'utilisation des Technologies de l'Information et de la Communication (TIC) par les administrations publiques pour faciliter les échanges de données. À la demande du comité TAC (*Telematics between Administrations Committee*) de l'IDA, le groupe d'audit Valoris a effectué une étude comparative sur les différents formats de documents disponibles sur le marché : Word .doc, RTF, Tex/LaTex, PostScript, PDF, etc. Le rapport Valoris définit ainsi ce qu'est un format ouvert :

> « Les conditions minimales pour constituer un standard ouvert sont les suivantes : le format de document doit être complètement décrit dans des documentations accessibles publiquement ; c'est-à-dire que ces descriptions peuvent être distribuées librement et le format de document peut être implémenté dans des programmes sans restrictions, gratuitement et sans contraintes légales. » ([4], p. 20).

Le rapport énumère d'autres critères importants : en particulier, le format ne doit pas être binaire et il doit supporter les fonctionnalités des traitements de texte actuels ainsi que les besoins futurs. Seuls deux des formats examinés parviennent à remplir ces critères : celui d'Office 2003 et celui d'OpenOffice.org. Tout en ayant une préférence pour le format ODF (*Open Document Format*) d'OpenOffice - à l'époque en cours de standardisation par le consortium OASIS (*Organization for the Advancement of Structured Information Standards*) -, le rapport ne donne toutefois pas de "vainqueur" définitif et met en avant les qualités de chacun des deux formats[3].

1.9. À la suite du rapport Valoris, le comité TAC de la Commission Européenne a publié en 2004 une série de recommandations. Il a demandé en particulier de soumettre le standard émergent ODF à une organisation de normalisation officielle telle que l'ISO.
La véritable ouverture XML est en effet atteinte lorsque le format natif des documents est un format de travail *ouvert* (rien n'est caché, tout est public et publié) et *standardisé* (le format est contrôlé par un organisme indépendant et n'est pas propriétaire).
Après avoir défini ODF, conçu à partir du format utilisé dans la version 1.0 de la suite OpenOffice, le consortium OASIS l'a adopté comme standard en Mai 2005 et accepté la requête du comité TAC ; ODF est depuis Mai 2006 un standard ISO (ISO/IEC 26300).
ODF est le format de document natif de la suite OpenOffice.org version 2.0. Il est aussi utilisé par d'autres traitements de texte moins connus (AbiWord, KWord, TextMaker) et des services en ligne comme ajaxWrite, Zoho Writer ou Writely (racheté par Google) ; il sera aussi présent dans la future version du client IBM Lotus Notes.

Microsoft de son côté a fait évoluer les formats XML utilisés dans Office 2003 (dont WordprocessingML) et développé son propre format nommé Open XML qui sera utilisé par défaut dans tous les composants de la future suite Office 2007. Ce format est en cours de standardisation auprès de l'ECMA International (*European Computer Manufacturers*

[3] - Le sujet des formats ouverts est presqu'aussi polémique que celui des logiciels libres, surtout lorsqu'il fait intervenir un format proposé par Microsoft ; la définition du rapport Valoris a le mérite de se concentrer sur le sujet et d'être suffisamment "modérée" pour ne pas exclure un acteur au seul motif de sa position dominante.

Association), et - conformément là aussi aux recommandations émises par le comité TAC - Microsoft annonce son intention de le proposer également comme standard ISO.

ODF et Open XML sont fonctionnellement similaires. Mais malgré quelques ressemblances architecturales, les implémentations techniques en XML[4] de leurs fonctionnalités communes sont très différentes. Autrement dit, il s'agit de deux formats XML qui font pour l'essentiel la même chose mais qui sont incompatibles. OpenOffice 2.0 permet cependant déjà de lire et d'écrire des documents au format Word 2003 et supportera probablement à l'avenir Open XML. Microsoft de son côté participe à un projet *open source* en partenariat avec d'autres sociétés pour proposer un *plug-in* d'Office 2007 permettant d'exploiter directement des documents ODF [6]. Ce développement est librement téléchargeable tout comme le *plug-in* permettant de produire des documents PDF depuis Office ; un convertisseur ODF/Open XML en mode ligne de commande est aussi proposé pour effectuer des conversions de documents en masse ; Microsoft reconnaît d'ailleurs qu'il s'agit de satisfaire aux demandes d'organisations gouvernementales de plus en plus nombreuses qui souhaitent travailler avec le format ODF.

Les deux formats ODF et Open XML sont clairement concurrents et il n'est guère possible de dire actuellement si l'un des deux prendra le pas sur l'autre. Il se pourrait d'ailleurs qu'ils soient à l'avenir tous deux standardisés par l'ISO, et il est très probable en tout cas que l'on soit amené à travailler en bureautique avec ces deux formats de la même manière que l'on utilise plusieurs variantes de RSS et Atom dans la syndication de contenu Web. Chaque partie tente de convaincre des sociétés importantes et des institutions prestigieuses de l'excellence de son format, en particulier en ce qui concerne son ouverture, son interopérabilité et sa compatibilité avec l'existant ; Microsoft argumente ainsi en ce moment sur le fait que son nouveau format Open XML est meilleur qu'ODF car il permettrait de préserver au mieux les anciens formats propriétaires d'un certain … Microsoft.

Microsoft a également défini XPS (*XML Paper Specification*, anciennement connu sous le nom de "Metro"), un nouveau format XML destiné à la fois à l'affichage des documents dans Windows et à leur impression. Ce format est dédié à l'échange des documents sous leur forme finale. Il est donc potentiellement concurrent de XSL-FO et surtout de PDF bien que sa spécification actuelle ne soit pas encore au même niveau ; un document XPS reste statique tandis que PDF possède de nombreuses caractéristiques permettant de rendre un document actif. Nul doute cependant que XPS évoluera s'il rencontre le succès ; des imprimantes, des RIP et des copieurs XPS sont déjà annoncés. XPS sera lancé avec Windows Vista et Office 2007 proposera une possibilité d'exporter tout document selon ce format.
XPS ne doit pas être confondu avec un format natif comme Open XML ou ODF car il ne prétend pas représenter toute la richesse des documents. C'est pourquoi seul l'export vers XPS est possible dans Office 2007 ; l'import de documents XPS n'est pas possible. Il est à noter également que XPS n'est pas un format ouvert à la différence d'Open XML ; il reste un

[4] - Un exemple de ressemblance architecturale ; Open XML emprunte à ODF la technique des conteneurs zip (*zip package*) ; un document Word au format Open XML (suffixe *.docx*) comme un document du traitement de texte Write d'OpenOffice (suffixe *.odt*) sont constitués d'un assemblage de différents fichiers XML compactés en zip – il suffit pour s'en convaincre de changer le suffixe d'un document *.docx* ou *.odt* en *.zip* et de le décompacter pour accéder à l'ensemble de ces fichiers XML répartis dans différents dossiers.
En ce qui concerne les implémentations différentes, nous renvoyons à l'article *Format comparison between ODF and MS XML* [5] qui décrit simplement quelques choix techniques divergents entre les deux formats. Ce papier est toutefois assez partisan en faveur d'ODF et a été rédigé avant la spécification d'Open XML qui a un peu "corrigé" certains défauts relevés, notamment en ce qui concerne la réutilisation des standards (Dublin Core, par exemple).

format propriétaire et soumis à copyright, bien que sa licence soit gratuite afin d'encourager son développement.

Pour conclure cette première partie, nous estimons qu'il y a bien *progression* dans les différentes étapes décrites selon l'axe que nous avons nommé *Ouverture XML*, et l'apparition de formats XML natifs et standardisés constitue l'aboutissement actuel d'une évolution importante pour le secteur du prémedia; en effet, ce mouvement vers l'ouverture permet non seulement de mieux assurer la pérennité des contenus et leur indépendance vis-à-vis des fournisseurs de technologies mais aussi de concevoir des solutions de plus en plus performantes. Toutefois, cela ne signifie pas bien sûr que les produits qui ne s'appuient pas sur un format XML natif soient dénués d'intérêt. Chez de nombreux éditeurs de presse, les besoins en contenus XML restent pour l'instant relativement limités ; ils interviennent en début et/ou fin de chaîne de production ou ne nécessitent pas de "retour" d'un format XML vers un format d'article ou de mise en forme propriétaire. Ces besoins peuvent alors être satisfaits avec des solutions d'import/export simples mais maîtrisées. Nous estimons pourtant que la progression décrite vers les formats XML natifs, ouverts et non propriétaires, constitue un réel mouvement de fond que les éditeurs de presse doivent suivre avec attention et accompagner.

2. Les possibilités de structuration XML

2.1. Avant de nous intéresser aux structurations représentées dans un format de document, rappelons quelques définitions habituelles concernant les types de contenus XML :

- Un élément XML possède un *contenu mixte* (*mixed content*) quand il peut contenir à la fois des données textuelles et des sous-éléments. Dans l'exemple suivant :
  <salutation>Bonjour, cher<name>Jean Dupont</name></salutation>
  l'élément *salutation* est mixte car il contient à la fois du texte ("Bonjour, cher") et un sous-élément *name*.
- « Les contenus *orientés données* [*data-centric* ou *record-like*] sont caractérisés par une structure assez régulière, des données qui présentent une granularité fine (c'est-à-dire que la plus petite unité indépendante de donnée est située au niveau d'un élément de type PCDATA unique ou d'un attribut), et peu ou pas du tout de contenus mixtes. L'ordre dans lequel les éléments enfants d'un même parent [les *sibling elements*] et les PCDATA apparaissent n'est en général pas significatif, excepté lorsque l'on valide le document au sens XML […]
- Les contenus *orientés document* [*document-centric* ou *narrative-like*] sont habituellement conçus pour être utilisés par des humains. Les livres, messages électroniques, annonces, ainsi que presque toutes les pages XHTML écrites à la main constituent des exemples de tels documents. Ils sont caractérisés par une structure moins régulière ou même franchement irrégulière, des données qui présentent une granularité plus grande (c'est-à-dire que la plus petite unité indépendante de donnée peut être située au niveau d'un élément mêlant différents contenus, voire même au niveau du document lui-même), et beaucoup de contenus mixtes. L'ordre dans lequel les éléments enfants d'un même parent et les PCDATA apparaissent est presque toujours significatif […]
  La distinction entre contenus *orientés données* et contenus *orientés document* n'est pas toujours claire en pratique. Un contenu *orienté données* comme une facture par exemple peut contenir aussi des données de granularité forte et irrégulièrement structurées telles que des descriptions. Et inversement, un contenu *orienté document*

comme un manuel utilisateur peut contenir des données de granularité fine et régulièrement structurées, telles que le nom de l'auteur ou une date de révision (il s'agit la plupart du temps de métadonnées). » (Ronald Bourret, [7]).

Au delà de l'aspect visuel des documents, un format XML de traitement de texte ou de mise en page doit aussi représenter une multitude de fonctionnalités caractéristiques telles que les différentes versions du texte, l'historique des corrections, les champs de formulaires, les données linguistiques au long du texte, les annotations, la gestion des droits d'accès sur certaines sections, les métadonnées, etc. ; il doit aussi permettre de stocker des macros, représenter les fonctionnalités spécifiques à une certaine plate-forme (ActiveX par exemple sur Windows), etc. Un tel format est donc bien évidemment complexe[5] et quand on entre dans le détail, la distinction entre contenus orientés données et contenus orientés document y apparaît moins limpide qu'elle ne semble au premier abord. La distinction est encore moins claire dans un format comme Open XML de Microsoft qui tend à éviter les éléments mixtes en sérialisant par exemple la représentation des styles de paragraphes et de caractères [9 et 10]. Mais malgré cette grande complexité, la structuration décrite à l'aide de ces formats XML est, du point de vue global qui est celui du document, essentiellement tournée vers sa présentation visuelle : formats des pages, géométrie et positionnement des blocs et éléments graphiques, représentation des attributs de caractères, des styles de caractères et de paragraphes, description des tableaux, images, etc. Cette structuration globalement gouvernée par la présentation est bien sûr typique d'un contenu *orienté document*.

Dans les traitements de texte récents comme OpenOffice 2.0, les possibilités de filtres depuis une structure XML externe vers la structure interne supportée par le traitement de texte (et vice-versa) sont souvent considérées comme un point important de l'ouverture au monde XML ; ces filtres sont réalisés à l'aide de transformations XSL paramétrables déclenchées de façon transparente, et permettent d'accéder directement en lecture et écriture à certains formats de documents comme par exemple une version simplifiée de TEI (*Text Encoding Initiative*, un standard pour l'échange de textes électroniques dans le domaine universitaire). Il est bien évident cependant que ces possibilités intéressantes restent limitées par la correspondance nécessaire des deux structures ; lors d'une lecture, que faire des éléments de la structure importée qui ne correspondent pas à la structure de présentation prévue dans le traitement de texte ? Un format éditorial comme NITF (*News Industry Text Format*, défini par l'IPTC pour l'échange des articles de presse) posera problème pour les traitements de texte, car, par exemple, les codes sujets, la date de publication, l'édition, les droits, etc. ne possèdent pas d'équivalents dans Word ou OpenOffice sauf à utiliser des variables de documents évidemment propriétaires. On voit bien que ces logiciels tout comme Quark et InDesign d'ailleurs ne constituent pas des éditeurs XML mais restent attachés à leur "métier de base" : présenter et mettre en forme des documents. C'est pourquoi ils sont classés dans la ligne 1 intitulée *Structuration selon la présentation* en suivant l'axe vertical *Structuration XML* de notre tableau.

[5] - Le format ODF utilisé par OpenOffice est décrit dans un document de 700 pages ; Open XML de Microsoft nécessite plus de 4000 pages. Ces chiffres sont d'ailleurs actuellement utilisés par les supporters de chacun des deux formats pour disqualifier le concurrent. L'*Open Document Alliance* [8] créée pour promouvoir ODF estime ainsi qu'Open XML sera très difficile à implémenter en raison de cette complexité ; de fait, les développements actuels autour de ce format utilisent les API et l'environnement de programmation fournis par Microsoft, ce qui rétrograde *ipso facto* le format Open XML dans la catégorie n° 6 de notre tableau… Microsoft de son côté affirme que les volumes des deux documentations sont simplement proportionnels aux fonctionnalités des formats ; à quoi les partisans d'ODF rétorquent que les spécifications de leur format sont plus courtes parce que celui-ci réutilise au mieux les standards existants…

2.2. Pour saisir un document XML selon une structuration personnalisée indépendante de la présentation affichée par les traitements de texte habituels, il est bien sûr possible d'utiliser un éditeur XML tel qu'Altova XML Spy ou Authentic. Mais il s'agit là de "couteaux suisses" techniques qu'il est pratiquement impossible d'envisager sur un poste auteur. Il existe toutefois certains outils bien mieux adaptés au travail du rédacteur dans le domaine de l'édition technique où les documents sont très fortement structurés. Ces logiciels, issus du monde SGML pour la plupart, possèdent des facilités permettant de simplifier le travail de rédaction en conformité avec les DTD ou W3C Schémas orientés métiers qu'ils supportent. Ces *authoring environments* comme XMetal Author, Syntext Serna, XXE ou PTC Arbortext Editor (anciennement Epic) proposent en effet une ergonomie de travail assez semblable à celle des traitements de texte habituels tout en présentant le plus simplement possible la syntaxe XML des documents et en affichant clairement la rigueur structurelle requise, la plupart du temps sous la forme d'un arbre. L'auteur peut aussi visualiser son document transformé selon une ou plusieurs XSL de son choix, ou, pour certains produits, selon des feuilles de styles CSS (*Cascading Style Sheets*) dont la technologie n'est pas XML et est empruntée à HTML.
Tous ces produits autorisent le travail sur des structures XML complexes, qu'elles soient orientées documents ou orientées données; on peut donc concevoir avec eux des documents qui présentent des jeux de métadonnées élaborés et sont à la fois structurés selon la présentation et selon le métier.
Quelques expériences dans la presse ont été réalisées avec ce genre d'éditeur mais elles n'ont pas rencontré de grand succès car les logiciels en question ne possèdent tout de même pas la simplicité d'un véritable traitement de texte et le rédacteur estime en général que cela rend son travail fastidieux et délicat car il doit en permanence prendre à garde à ne pas détériorer la structure de son document. Dans la presse et l'édition non technique, le déploiement d'un outil "trop explicitement XML" sur le poste auteur est toujours une entreprise malaisée ; peu ont réussi à imposer un outil de ce type à la source en raison de la prédominance de Word et des changements d'habitude imposés. Signalons cependant que Serna supporte en standard le format éditorial NITF et pourrait faire de même très prochainement avec NewsML ; il est aussi à notre connaissance le seul outil de cette classe à permettre un affichage de type traitement de texte basé sur XSL-FO, ce qui invalide l'idée répandue selon laquelle cette technologie serait inefficace pour la présentation à l'écran.

Il existe d'autres logiciels moins techniques et moins puissants qui permettent au rédacteur de rédiger directement des documents XML dans un navigateur Web. Des produits comme Ephox EditLive! for XML ou Q42 XOpus sont ainsi utilisés pour alimenter en contenus XML relativement simples des CMS (*Content Management Systems*) ainsi que des sites ou des portails Web. Là encore, le rédacteur peut à tout moment afficher son document transformé selon une XSL de son choix. Les structurations de documents que ces logiciels autorisent restent très simples comparées aux possibilités des outils de l'édition technique puisqu'il s'agit de formulaires contenant des zones contrôlées et des blocs définis acceptant la saisie de texte riche (avec choix de police, corps, gras, italique, etc.) : autrement dit, ils permettent de travailler avec des contenus essentiellement orientés données et les seuls contenus orientés documents qu'ils autorisent sont des blocs de texte riche. Néanmoins l'interface et le travail avec ces outils exigent moins d'expérience et d'attention qu'avec un véritable *authoring environment* car XML y est "masqué" ; les fonctionnalités de base que l'on est en droit d'exiger dans un véritable traitement de texte rédactionnel sont présentes dans ces outils qui, eux aussi, sont parfois utilisés dans la presse.

Les éditeurs Xsmile du système éditorial EidosMedia Methode et Dixit de MVS adoptent une approche un peu similaire ; ils n'imposent pas de DTD, mais permettent de définir et gérer

une ou plusieurs DTD selon les nécessités de chaque organisation. Dans le système Wedia Open[3] comme dans EidosMedia Methode, les articles et les pages sont affichées à l'aide de feuilles de styles CSS qui permettent d'obtenir différents aperçus précis et réels dans toutes les formes possibles et pour toutes les éditions et publications. On notera qu'aucun des systèmes éditoriaux destinés à la presse et nativement XML n'utilise XSL-FO pour l'affichage des documents sur le poste du rédacteur.
Dans Xsmile, les éléments graphiques d'une page sont décrits en SVG (*Scalable Vector Graphics* - langage défini par le W3C pour la description des graphiques en deux dimensions). On remarquera également que, à la différence de la plupart des *authoring environments* et d'Office 2007, les produits proposés par EidosMedia, Wedia et MVS utilisent les DTD pour la validation mais ne supportent pas encore les W3C Schémas ; les contraintes plus rigoureuses sur les données permises par les schémas ou le contrôle des types de données n'y sont donc pas possibles.

Le système Methode est à notre connaissance l'un des rares environnements professionnels destinés à la presse qui permette de décrire non seulement des articles avec une grande liberté de structure mais également les pages et le chemin de fer sous forme XML. De ce point de vue, il se rapproche des *authoring environments* de l'édition technique en facilitant le travail avec des structurations personnalisées qui gouvernent l'affichage visuel ; en d'autres termes, la structuration de présentation dérive de la structuration personnalisée par l'application de styles CSS. La structuration personnalisée mise en œuvre doit bien évidemment, être publiée sous forme de DTD ou W3C Schémas, mais elle peut, dans une certaine mesure, être propriétaire ou standardisée ; c'est pourquoi nous avons regroupé ces outils dans une seule cellule de la ligne 2 de notre tableau, en distinguant toutefois les trois types de caractéristiques que nous venons de décrire brièvement et qui correspondent aux marchés spécifiques auxquels ils s'adressent.

Les structures de documents utilisées avec les logiciels de l'édition technique s'affranchissent de plus en plus souvent des contraintes liées à la présentation éditoriale classique car ce secteur privilégie dorénavant d'autres canaux de diffusion que le papier. À preuve, le succès grandissant de DITA (*Darwin Information Typing Architecture*), une technologie XML développée initialement par IBM, maintenant confiée au consortium OASIS, et devenue de ce fait une technique ouverte [11 et 12]. DITA est destiné à la rédaction, à la production et à la livraison de documents techniques sous diverses formes. Son extension est rapide, surtout aux États-Unis où il tend d'ailleurs à sortir du cadre de la réalisation des documents techniques pour gagner d'autres secteurs du *publishing*. DITA est supporté par un nombre croissant d'éditeurs et d'outils. Le monde de l'édition technique connaît bien le standard DocBook permettant la description des ouvrages selon leur mise en forme (chapitres, pages, etc.). DITA au contraire ne structure pas les documents selon leur mise en forme, mais a été conçu pour décrire des unités d'information (*topics*) en vue de réduire la redondance (copie de contenu d'un endroit à un autre du document) et de faciliter la réutilisation des contenus ; un *topic* possède un *type* et constitue une unité d'information réutilisable ; les *topics* sont ensuite regroupés sous la forme d'ensembles. DITA ouvre aussi de nouvelles possibilités par rapport au papier en permettant par exemple la découverte progressive (*progressive disclosure*) des contenus rédigés selon divers niveaux (résumé, description courte, description longue) ou divers publics (débutant, expert). Avec le développement de nouveaux médias indépendants des contraintes de présentation sur papier, qui agrègent de nombreuses sources en plus de leur production propre et répondent au besoin d'une information rapidement lue et mise à jour, et où, enfin, la catégorisation de l'information en nombreux sujets est déterminante (Wikio par exemple), la presse devrait probablement être attentive à ces nouvelles façons de concevoir le travail du rédacteur.

2.3. Les traitements de texte et les logiciels de mise en page généralistes offrent une liberté et une souplesse inégalables pour structurer un document selon sa présentation ; cette flexibilité ne peut être atteinte avec les logiciels décrits précédemment qui travaillent uniquement selon les contraintes d'une structuration prédéterminée dont dérive la présentation. Mais depuis plusieurs années d'autres besoins sont apparus ; en ce qui concerne la presse, par exemple : la gestion des objets multimédias et l'insertion de liens hypertextes dans le document ; la publication différenciée selon le canal (papier, Web, etc.) ou selon l'édition ayant chacun des formats et des chronologies spécifiques de distribution de l'information - l'article doit donc posséder des sections distinctes selon la cible ; la création de listes d'index en exploitant le balisage direct des entités nommées (sociétés, noms de personnes, etc.) sur le poste auteur ; permettre, depuis un tableau publié, l'accès direct et actualisé aux données chiffrées (au format Excel par exemple) ; proposer un ensemble riche de métadonnées contrôlées ; consolider les métadonnées d'une mise en forme à partir de celles de ses composants (articles, images), etc. Ces besoins, qui ne sont d'ailleurs pas tous implémentés loin s'en faut dans les systèmes professionnels actuels, impliquent la possibilité d'ajouter au document une structuration personnalisée conforme aux besoins métiers et non nécessairement liée à la présentation. On notera néanmoins que l'éditeur Dixit de MVS permet d'effectuer un *mapping* automatique d'une structure de présentation paramétrable vers une autre structure sans exiger l'écriture d'une transformation XSL et possède une fonctionnalité autorisant le balisage direct et individuel des entités nommées ; cette dernière caractéristique constitue une première possibilité d'appliquer une structuration sémantique personnalisée en plus de la structuration de présentation.

La première des recommandations du comité TAC publiées à la suite du rapport Valoris traduit sur le plan des formats de documents retenus ces nouvelles exigences concernant la structuration : il est demandé au comité technique d'OASIS d'étudier la possibilité pour le format ODF de supporter des schémas personnalisés (*custom-defined schemas*). Cette demande qui correspond à la ligne n° 3 de notre tableau est directement issue d'une fonctionnalité déjà présente dans Office 2003 et mise en avant par le rapport Valoris.

Il est en effet possible d'enrichir un document Word 2003 avec une structuration sémantique conforme au souhait de l'utilisateur. Cette structuration personnalisée est techniquement contrôlée par un ou plusieurs W3C Schémas associé(s) au document. Elle permet aux utilisateurs d'annoter le document avec des éléments issus des schémas personnalisés qui lui sont attachés. Cela rend possible le fait d'injecter un balisage orienté métier au document de telle façon qu'il puisse être traité dans certaines applications selon ce balisage métier plutôt que selon le balisage plus générique de Word. Un document peut donc être structuré à la fois :

- selon les schémas XML de référence définis dans Open XML ; ces schémas sont fondamentalement orientés présentation (attributs de caractères et paragraphes, styles, etc.) et permettent l'archivage et l'interopérabilité[6] ;
- selon des schémas personnalisés pour représenter l'information métier présente dans le document ; ces schémas sont orientés données et permettent une décomposition du document en éléments signifiants conformes au besoin de l'utilisateur. Ils offrent la possibilité d'un modèle documentaire orienté données, et non pas orienté présentation.

[6] - Avec Word 2007, il est aussi possible de personnaliser l'interface utilisateur en insérant du code XML spécifique (non Open XML bien sûr) dans un document.

Les balises actives (*smarttags*) de Word 2003 permettent également d'ajouter au document des marquages personnalisés du contenu, en reconnaissant automatiquement des listes statiques de termes et en y associant éventuellement des actions en relation avec le Web.

Le déploiement de schémas personnalisés introduit cependant une technicité certaine sur le poste de travail : affichage de balises, arbre des éléments dans le document, validation selon les schémas transforment le traitement de texte généraliste en éditeur XML similaire aux outils de l'édition technique que nous avons évoqué précédemment. Le travail de rédaction est plus délicat car il est malheureusement facile d'endommager la structure du document. Par ailleurs, on retrouve dans le codage XML du document des balises définies dans les schémas personnalisés mêlées aux balises du schéma de référence ; l'exploitation ultérieure de la structuration personnalisée du document nécessite ainsi de transformer celui-ci, la plupart du temps à l'aide d'une XSLT. Au total donc, ces schémas personnalisés entraînent à la fois une plus grande difficulté pour l'auteur et un travail technique supplémentaire (et pas toujours simple) de transformation et exploitation.

La version 2007 de Word permettra d'envisager des solutions plus simples en donnant la possibilité de définir dans un document des contrôles de contenus personnalisés (*content controls*). Ces contrôles permettent d'ajouter au document une structuration personnalisée contenant du texte simple ou riche, des listes déroulantes, des images, des dates contrôlées par calendrier, etc. L'utilisateur est alors guidé dans sa saisie par ces contrôles de contenus qui se comportent un peu à la manière de sections de formulaires. À l'exception du contrôle permettant la saisie de texte riche, cette technologie permet donc d'agrémenter le document de contenus uniquement orientés données. Aucun concept XML (élément, attribut, structure, validation) n'est visible à l'écran ; le travail sur le document devient plus sûr en étant à l'abri des modifications accidentelles qui peuvent intervenir quand la structure XML est apparente. L'exploitation est elle aussi largement simplifiée ; les contenus saisis à l'aide de ces contrôles peuvent être isolés dans un fichier XML très simple et personnalisé maintenu en permanence par le traitement de texte dans le conteneur *zip* qu'est le document ; il est donc possible d'exploiter ces contenus structurés sans avoir à transformer un document XML compliqué avec une XSLT.

3. Conclusion

À l'instar de la progression vers des formats XML natifs, ouverts et non propriétaires, le support de schémas personnalisés orientés métiers en complément de schémas de référence essentiellement tournés vers la présentation constitue une évolution importante. Ce double mouvement vers l'ouverture et l'interopérabilité ainsi que les possibilités de structurations personnalisées doit à notre avis être poursuivi et s'étendre également aux logiciels de mise en page généralistes ainsi qu'aux systèmes spécialisés ; les formats de documents et les processus métiers constituent actuellement des enjeux fondamentaux pour les applications bureautiques classiques et, à court terme et par extension inéluctable, pour les outils de mise en forme généraliste et les systèmes éditoriaux.
Nous proposons donc pour terminer cette étude quelques remarques (qui n'engagent que l'auteur…) concernant ce que nous souhaiterions voir apparaître à l'avenir :

- Selon le point de vue du consortium OASIS, le support de schémas personnalisés avec ODF n'est pas en soi un problème puisqu'il s'agit juste d'inclure dans un document des balises provenant d'un vocabulaire tiers, ce qui est précisément l'objet des espaces de noms XML. Il ne semble pas cependant que son implémentation figure dans les

développements annoncés d'OpenOffice, ce qui était pourtant explicitement souhaité dans les recommandations du comité TAC de la Commission Européenne ; cette demande nous paraît toujours actuelle et importante.

- Il serait souhaitable que soient développées des API libres et multiplateformes permettant de travailler avec les différents formats de documents XML natifs. Ces formats sont en effet complexes à manipuler sans une couche d'abstraction logicielle qui les rende accessibles à un grand nombre de développeurs ; ceci est particulièrement crucial pour Open XML car Microsoft impose pratiquement l'utilisation de ses outils .Net pour travailler commodément avec ses formats.
- La distinction entre mise en page et traitement de texte devenant de moins en moins significative, les éditeurs de logiciels de mise en page comme Quark et Adobe devraient enfin proposer des formats XML natifs et ouverts. Le langage SVG fournit un bon cadre pour la description de la géométrie des pages et il a d'ailleurs été adopté comme nous l'avons vu par des fournisseurs de systèmes professionnels ; il pourrait l'être aussi par les éditeurs d'outils généralistes et, pourquoi pas, rendre leurs logiciels facilement interopérables (on peut rêver). Ceci permettrait par exemple d'envisager des solutions intégrant au mieux les documents issus de sources diverses en autorisant leur édition *in situ* lors de la mise en page tout en conservant leur format originel.
- Les éditeurs de systèmes éditoriaux doivent également prendre en compte XML nativement dans leurs systèmes et ne plus se satisfaire de solutions d'import/export plus ou moins paramétrables en début et fin de production des publications. Comme le reconnaissent les éditeurs de solutions déjà engagés dans le "tout XML", c'est l'essor de l'édition selon plusieurs canaux qui a enfin donné un sens concret à leurs efforts. La description d'une parution complexe et en général de n'importe quel objet manipulé par ces systèmes est en effet mieux adaptée à un format arborescent comme XML qu'au relationnel.
- Les schémas personnalisés, les balises actives et les contrôles de contenus devraient être adaptés par d'autres éditeurs de logiciels afin de proposer des outils rédactionnels très simples. Ils devraient aussi faire leur apparition sur les outils de mise en page quand leurs formats de document seront nativement XML.
- Des contrôles de contenus plus évolués devraient être développés pour faciliter l'ajout de contenus orientés données plus complexes : structures arborescentes plutôt que simples formulaires linéaires, spécification des contenus à l'aide d'attributs, etc.
- Des technologies plus avancées que les schémas personnalisés et les contrôles de contenus devraient permettre d'enrichir un document avec de véritables contenus orientés documents (au delà donc de la saisie de type formulaire) tout en "masquant" la technicité XML que sont les balises et arbres structurels.
- Il pourrait être intéressant d'adapter les concepts à l'œuvre dans DITA au domaine de la presse quand la catégorisation par sujets est forte et la structuration éditoriale ou narrative n'est pas prédominante ; la conception modulaire du futur NewsML 2, qui prend en charge la différenciation des contenus selon leur canal de diffusion par le biais des "vues" (*renditions*), pourrait peut-être permettre de concevoir une architecture rédactionnelle de ce genre, une sorte de "DITA-news". Plus généralement, nous estimons que l'exploitation approfondie de XML dans l'édition technique devrait davantage inspirer la presse.
- La modélisation des documents à structures multiples est encore un sujet de recherche (voir par exemple [13]) mais devrait permettre à terme de mettre au point des techniques destinées à faciliter réellement le travail avec des structures orientées métiers associées à des structures orientées présentation.

Tableau. Positionnement de traitements de texte et outils de mise en page courants selon deux critères

Degré d'ouverture XML et possibilités de structuration XML

Ouverture XML ⇒ **Structuration XML** ⇓	**1**. Export XML à l'aide d'un format tiers (conversion)	**2**. Export XML direct à l'aide d'une extension	**3**. Import/Export XML direct à l'aide d'une extension	**4**. Import/Export direct de formats XML spécifiques	**5**. Import/Export direct et paramétrable de formats XML	**6**. Import/Export XML + Schéma DOM ≈ API XML normalisée	**7**. Format XML natif = Sérialisation DOM	**8**. Format XML natif propriétaire et publié	**9**. Format XML natif publié et standardisé
1. Structuration selon la présentation (styles, attributs typographiques, etc.)	Sauvegarde de documents Word en RTF et convertisseur : - Tétrasys Mosca - Logictran R2Net - etc.	Word et extension: - YAWC Pro - etc. XPress et extension: - avenue.quark - Easypress Atomik Xport - CDROM-SNI - Eurocortex - Rosebud e-Gate - Trias - etc.	Word et extension : - Media Entities XML ToolWorks for Word XPress et extension: - avenue.quark et XML Import - Easypress Atomik Roundtrip - Trias XMLImport/XML Export - etc.	Protec Milenium (Insert)	InDesign CS, CS2 FrameMaker 2.0+ Unisys Hermès (MediaApi)	XPress 6.5 et extension XML Plus XPress 7 (QXML)	InDesign Interchange Format (INX)	OpenOffice.org 1.0 Scribus 1.2+ (logiciel libre de mise en page) Wedia Open[3] Éditeurs rédactionnels pour correspondants	OpenOffice.org 2.0+, Abiword 2.4+, KWord 1.4+, etc. Open Document Format (ODF, standard OASIS)
2. Structuration personnalisée (DTD ou W3C Schémas personnalisés)								PTC Arbortext Editor (Epic) XMetal Author Syntext Serna XMLmind XML Editor (XXE) Ephod EditLive! for XML Q42 XOpus EidosMedia Xsmile	
3. Structuration selon la présentation et structuration personnalisée								MVS Dixit Word 11 (Word 2003) Format WordprocessingML	Futur Office 2007 Format Open XML (standard ECMA)

Références

[1] David Miller - *Adobe's InDesign and XML*
http://www.xml.com/pub/a/2004/08/04/indesign.html

[2] Blog de Eric van der Vlist - *Quark's desperate attempt to keep XML under control* (Juillet 2005)
voir aussi *InDesign and XML : no better than its competitor* (Août 2005)
http://eric.van-der-vlist.com/blog/

[3] Adobe – *Working with INX file format* (document *ww-inx-file-format.pdf*, Novembre 2005), in Adobe InDesign CS2 Products SDK
http://partners.adobe.com/public/developer/indesign/sdk/index.html

[4] European Commission - *Comparative assessment of Open Documents Formats Market Overview (Valoris Report, 2003)* ; voir également les réponses de Microsoft et de Sun
http://europa.eu.int/idabc/en/document/3439

[5] Alex Hudson, J. David Eisenberg, Bruce D'Arcus and Daniel Carrera - *Format comparison between ODF and MS XML*
http://www.groklaw.net/article.php?story=20051125144611543
voir aussi Wikipedia - *Comparison of OpenDocument and Microsoft XML formats*
http://en.wikipedia.org/wiki/Comparison_of_OpenDocument_with_Microsoft_XML_formats

[6] *ODF Add-in for Word 2007*
http://odf-converter.sourceforge.net/

[7] Ronald Bourret – *XML and databases*
http://www.rpbourret.com/xml/XMLAndDatabases.htm
traduit par Patrick Peccatte sur le site du CampuxXML
http://www.campusxml.org/news/fullstory.php/aid/614/XML_et_les_bases_de_donn%E9es.html

[8] *Open Document Alliance*
http://www.odfalliance.org/
voir aussi *The OpenDocument Foundation, Inc.*
http://www.opendocumentfoundation.us
et http://opendocumentfellowship.org

[9] Blog de Brian Jones (Microsoft) – *Open XML formats*
http://blogs.msdn.com/brian_jones/default.aspx
voir également le site http://openxmldeveloper.org

[10] Evan Lenz, Mary MacRae, Simon St. Laurent - *Office 2003 XML*. (O'Reilly)
Chapitre 2 en libre téléchargement - *The WordprocessingML vocabulary* :
http://www.oreilly.com/catalog/officexml/chapter/index.html?CMP=ILL-4GV796923290

[11] Don Day et al. - *An XML Architecture for Technical Documentation: The Darwin Information Typing Architecture*
http://www.winwriters.com/articles/DITA/index.html

[12] *OASIS Darwin Information Typing Architecture TC Approves DITA Version 1.0.*
http://xml.coverpages.org/ni2005-02-17-a.html

[13] Abascal et al. – *Modéliser la structuration multiple des documents, 2005*
http://liris.cnrs.fr/~yprie/download/h2ptm03-ms.pdf