PRÉSENTATION D’OFFICE OPEN XML

ECMA TC45 TOM NGO (NEXTPAGE), EDITEUR DE LOGICIELS

# 1 Introduction

Office Open XML (OpenXML) est une proposition de standard ouvert pour des documents de traitements de texte, des présentations et des feuilles de calcul pouvant être implémentés par de multiples applications et sur de multiples plateformes. Sa publication intéressera toutes les organisations souhaitant mettre en œuvre des applications compatibles avec ce format, les entités commerciales et gouvernementales qui utilisent ces applications, ainsi que les formateurs ou auteurs qui enseignent le format. À terme, tous les utilisateurs bénéficieront des avantages d’un standard XML pour leurs documents : stabilité, préservation, interopérabilité et évolutivité.

Les travaux de normalisation d’OpenXML ont été effectués par le comité technique 45 (TC45) d’Ecma International, constitué de représentants d’Apple, Barclays Capital, BP, la British Library, Essilor, Intel, Microsoft, NextPage, Novell, Statoil, Toshiba ainsi que la Bibliothèque du Congrès des Etats-Unis (1). Ce livre blanc résume les caractéristiques d'OpenXML. Lisez-le afin de :

* comprendre les objectifs d’OpenXML et la structure de la spécification ;
* découvrir ses propriétés : comment il adresse les problématiques de rétrocompatibilité, de préservation, d’extensibilité, de schémas personnalisés, de définition de sous-ensembles, de plateformes multiples, d’internationalisation et d’accessibilité ;
* apprendre à respecter la structure de haut niveau commune à tous les fichiers OpenXML et accéder rapidement à n’importe quelle partie de la spécification pour obtenir des précisions complémentaires.

# 2 Objectifs du standard

OpenXML a été pensé dès le départ pour pouvoir représenter de manière fiable le corpus préexistant de documents de traitement de texte, de présentations et de feuilles de calcul codés dans des formats binaires définis par Microsoft Corporation. Le processus de normalisation a consisté à reproduire en XML les capacités requises pour représenter l’ensemble des documents existants, à les élargir, à fournir une documentation détaillée et à favoriser l’interopérabilité.

À cet instant, plus de 400 millions d’utilisateurs génèrent des documents dans des formats binaires, ce qui, d’après nos estimations, représente plus de 40 milliards de documents, et des milliards d’autres sont créés chaque année.

Les formats binaires originaux de ces fichiers ont été développés à une époque où l’espace était précieux et où le temps d’analyse pénalisait le travail des utilisateurs. Ils s’appuyaient sur la sérialisation directe des structures de données en mémoire utilisées par les applications Microsoft® Office®. Les infrastructures matérielles, réseau et les standards modernes (en particulier XML) rendent possible une conception innovante et ouverte aux évolutions futures, permettant une mise en œuvre multifournisseur et multiplateforme. En parallèle à ces avancées technologiques, les marchés se sont diversifiés et englobent à présent une nouvelle gamme d’applications qui, à l’origine, n’étaient pas prises en compte dans l’univers simple des programmes d’édition documentaire. Ces nouvelles applications peuvent notamment :

* générer des documents automatiquement à partir de données métier ;
* extraire des données métier de documents et les exploiter dans des applications métier ;
* accomplir des tâches limitées sur une petite partie d’un document, sans compromettre les possibilités de modification ultérieure ;
* apporter des options d’accessibilité aux populations d’utilisateurs présentant des besoins spécifiques, par exemple aux aveugles ;
* s’exécuter sur différents matériels, y compris sur des périphériques mobiles.

Le problème le plus fondamental est sans doute celui de la préservation à long terme. Nous avons appris à créer des quantités d’informations en augmentation exponentielle. Cependant, nous avons codé ces informations à l’aide de représentations numériques si intimement liées aux programmes qui les avaient créées qu’après une ou deux décennies, il est devenu en général extrêmement difficile de les lire sans une perte d'informations significative.

Préserver l’investissement financier et intellectuel que représentent ces documents (existants ou nouveaux) est devenu une priorité majeure. L’émergence de ces quatre forces - adoption généralisée des formats binaires, avancées technologiques, forces du marché exigeant des applications diversifiées et difficulté croissante de la préservation à long terme - a rendu indispensables la définition d’un format XML ouvert et la migration des milliards de documents existants vers ce format, avec des pertes aussi limitées que possible.

En outre, la normalisation de ce format XML ouvert et sa maintenance continue dans le temps créent un environnement où les organisations peuvent s’appuyer en toute confiance sur la stabilité de la spécification, avec la certitude que ses évolutions futures seront maîtrisées dans le cadre d'un processus de normalisation. Il existe plusieurs standards et spécifications de document : HTML, XHTML, PDF et ses sous-ensembles, ODF, DocBook, DITA et RTF. À l’instar des nombreux standards de représentation des images bitmap - TIFF/IT, TIFF/EP, JPEG 2000 et PNG - chacun a été créé pour un ensemble d’objectifs particulier.

OpenXML répond à la nécessité d’un standard couvrant l'ensemble des fonctionnalités représentées dans le corpus de documents existant. À notre connaissance, c’est le seul format de document XML capable de prendre en charge toutes les fonctionnalités des formats binaires.

**3 Structure du standard**

OpenXML définit des formats pour les documents de traitement de texte, de présentation et de feuille de calcul. Chaque type de document est spécifié par un langage de marquage principal : WordprocessingML, PresentationML ou SpreadsheetML. Des mécanismes d’incorporation permettent à un document de l’un de ces trois types d’intégrer des éléments dans un autre langage de marquage principal et dans un certain nombre de langages de marquage secondaires. La spécification est composée de contenus normatifs (qui définissent OpenXML) et informatifs (qui facilitent la compréhension du lecteur mais ne sont pas indispensables). Elle est structurée en différentes parties couvrant les besoins de différents publics.

|  |  |
| --- | --- |
| Partie, Titre et nombre de pages | Description et table des matières succintes |
| Partie 1  Principes de base  165 pages | * Définit la terminologie, les conventions de notation et les abréviations. * Décrit les trois langages de marquage principaux et les langages de marquage secondaires. * Établit les conditions de conformité et propose des directives d’interopérabilité. * Décrit les contraintes dans les conventions Open Packaging applicables à chaque type de document. |
| Partie 2  Open Packaging Conventions  125 pages | * Définit les conventions Open Packaging (OPC). Chaque fichier OpenXML se compose d’un ensemble de flux d’octets appelés parties et combinés dans un conteneur, appelé package. Le format du package est défini par les conventions OPC. * Décrit une mise en œuvre physique conseillée des conventions OPC reposant sur le format de fichier ZIP. * Déclare les schémas XML pour les conventions OPC sous la forme de XML Schema Definitions (XSD) (2), dans une annexe publiée uniquement sous forme électronique. Cette annexe comporte également des représentations non normatives des schémas à l’aide de RELAX NG (ISO/CEI 19757-2) (3). |
| Partie 3  Livre élémentaire  466 pages | * Présente les fonctionnalités de chaque langage de marquage, avec contexte, exemples et diagrammes. Cette partie est informative (non normative). * Décrit le mécanisme de stockage de données XML personnalisées (Custom XML) au sein d’un package, afin de permettre l’intégration avec les données métier. |
| Partie 4  Références du langage de marquage  5756 pages | * Définit chaque élément et attribut, la hiérarchie des relations parent/enfant des éléments et d’autres caractéristiques sémantiques le cas échéant. Cette partie a vocation à servir de référence dans tous les cas nécessitant des informations complètes sur un élément ou un attribut. * Définit le mécanisme de stockage des données XML personnalisées. * Déclare les schémas XML pour les langages de marquage sous la forme de XSD (2), dans une annexe publiée uniquement au format électronique. Cette annexe exprime également ces schémas de manière non normative à l’aide de RELAX NG (ISO/CEI 19757-2) (3). |
| Partie 5  Compatibilité et extensibilité du marquage  34 pages | * Décrit les fonctionnalités disponibles pour l’extension de documents OpenXML. * Spécifie les éléments et les attributs permettant l’interopérabilité avec des applications aux extensions différentes. * Exprime les règles d’extensibilité à l’aide de NVDL (ISO/CEI 19757-4) (4). |

Pour faciliter la lecture et la consultation de ces documents, les versions électroniques comportent de nombreux liens internes actifs. La partie 4, notamment, intègre systématiquement des liens vers des éléments parents et enfants.

**4 Propriétés du standard**

Cette section prépare à l’étude d’OpenXML en décrivant quelques-unes de ses principales propriétés. Chaque paragraphe présente une de ces propriétés et fait référence à des fonctionnalités spécifiques d’OpenXML.

* Le paragraphe « Interopérabilité » décrit l’indépendance d’OpenXML par rapport aux formats, fonctionnalités et environnements d’exécution propriétaires, garantissant aux développeurs un choix particulièrement étendu.
* Le paragraphe « Internationalisation » énumère quelques méthodes représentatives de la prise en charge des principaux groupes de langues par OpenXML.
* Les paragraphes « Adoption facilitée pour les développeurs », « Compacité » et « Modularité » décrivent des approches spécifiques d’OpenXML permettant d'éviter ou d'éliminer des freins pratiques à la mise en œuvre par différents intervenants : courbe d’apprentissage, jeu de fonctionnalités minimum et performances.
* Le paragraphe « Migration haute fidélité » explique comment OpenXML atteint l’objectif hautement essentiel de préservation des informations, y compris les intentions originelles de l’auteur, dans des documents nouveaux et existants.
* Le paragraphe « Intégration avec les données métier » décrit comment OpenXML incorpore les informations métier dans des schémas personnalisés pour favoriser l’intégration et la réutilisation des données entre applications de productivité et systèmes d’information.
* Le paragraphe « Un potentiel d’innovation » montre comment OpenXML prépare l’avenir en définissant des mécanismes d’extensibilité complémentaires et en favorisant l’interopérabilité avec des applications présentant des jeux de fonctionnalités différents.

Le reste de ce document, cette section comprise, constitue un guide général sur OpenXML. Les références à la spécification se font toujours sous la forme §Partie:section.paragraphe. Par exemple : §1:2.5 correspond à la partie 1, section 2.5 de la spécification. Les références aux autres sections de ce document se font par nom.

**4.1 Interopérabilité**

Les développeurs peuvent écrire des applications qui exploitent et produisent de l'OpenXML sur de nombreuses plateformes.

L’interopérabilité d’OpenXML résulte avant tout d’un effort considérable de contribution, de modification et de révision de la spécification par les membres du comité technique TC45 d’Ecma (1), aux cursus et aux intérêts professionnels variés. Les membres de ce comité comprenaient :

* des fournisseurs (Apple, Intel, Microsoft, NextPage, Novell et Toshiba), avec de nombreux systèmes d’exploitation (Linux, MacOS et Windows) et de multiples utilisations d’OpenXML en prévision ;
* des entreprises (BP, Barclays Capital, Essilor, Statoil) ayant d’importants investissements dans du contenu existant, dont des systèmes de transaction critiques ;
* la British Library et la Bibliothèque du Congrès des Etats-Unis, toutes deux particulièrement intéressées par la problématique de la préservation.

Au cours de la préparation, les membres du comité ont soulevé et résolu des centaines de problèmes en matière de règles, de clarté, de sémantique et de possible dépendance par rapport à l’environnement. Catégories représentatives des problèmes et autres activités gérés :

* fonctionnalités permettant d’assurer l’indépendance, par rapport à la plateforme, de mécanismes qui étaient propriétaires dans les formats binaires d’origine ;
* conditions de conformité ;
* contenu des schémas ;
* représentations alternatives des schémas et des mécanismes d’extensibilité à l’aide de RELAX NG (ISO/CEI 19757-2) et de NVDL (ISO/CEI 19757-4) (4) ;
* développement d’outils automatisant l’analyse et la visualisation des schémas ;
* internationalisation ;
* exhaustivité, justesse et clarté des descriptions dans toute la spécification, généralement à la suite d’essais d’implémentation de parties de la spécification.

Le reste de ce paragraphe souligne des domaines spécifiques dans lesquels OpenXML se démarque des formats binaires d’origine à des fins d’interopérabilité. L’une des exigences essentielles en matière d’interopérabilité est l’indépendance par rapport à tout type de contenu source.

* OpenXML ne prévoit aucune restriction sur les types de fichier image, audio ou vidéo. Par exemple, les images peuvent être au format GIF, PNG, TIFF, PICT, JPEG ou tout autre type d’image (§1:14.2.12).
* Les contrôles incorporés peuvent être de tous types, par exemple Java ou ActiveX (§1:15.2.8).
* Les spécifications de police dans WordprocessingML peuvent comprendre des informations métriques et PANOSE facilitant le choix d’une police de substitution lorsque la police d’origine n’est pas disponible (§3:2.10.5).

En outre, OpenXML évite toute dépendance par rapport à l'environnement d'exécution de l’application qui a produit un document.

* L’exemple classique est celui d’un contrôle ou d’une application externe qui génère une image pour une portion de la surface d’affichage. Pour se prémunir contre le risque que le contrôle ou l’application soit indisponible ou impossible à exécuter dans un environnement spécifique, le fichier de document peut intégrer une représentation d’image. Ce mécanisme était également employé dans les anciens formats binaires.
* OpenXML offre un mécanisme plus général, dénommé bloc de contenu alternatif (§3:2.18.4), qui peut s’adapter à différentes situations où une application consommatrice est incapable d’interpréter ce qu’une application productrice a écrit. Il est couramment employé dans le contexte de l’extensibilité. Ce mécanisme est détaillé dans le paragraphe « Un potentiel d’innovation ».
* OpenXML évite toute dépendance par rapport à des paramètres qui ont une valeur dans l’environnement de production du document, mais pas nécessairement dans l’environnement d’utilisation. Par exemple, le paramètre CT\_SYSCOLOR est un index de table de couleurs dans l’environnement producteur. Pour assurer la portabilité vers un environnement d’utilisation différent, PresentationML permet au producteur de mettre en cache la couleur système utilisée au moment de la création du document.

Enfin, et plus fondamentalement, Office OpenXML est conforme aux standards W3C ouverts tels que XML (5) et XML Namespaces (6). À elle seule, cette caractéristique garantit une interopérabilité de base entre toutes les plateformes et les systèmes d’exploitation qui adhèrent à ces standards ouverts.

**4.2 Internationalisation**

OpenXML prend en charge les fonctionnalités d’internationalisation qu’exigent certaines langues comme l’arabe, le chinois (trois variantes), l’hébreu, le hindi, le japonais, le coréen, le russe et le turc. OpenXML gère l'unicode en natif, puisque c'est du XML. De plus, OpenXML offre une grande richesse de fonctionnalités d’internationalisation, affinées au fil des années. La liste suivante est représentative de ces options.

*Sens d’écriture*: OpenXML prend en charge les langues qui s’écrivent de gauche à droite et de droite à gauche. Il gère également les langues d’écriture bidirectionnelle telles que l’arabe, le farsi, l’ourdou, l’hébreu et le yiddish. Ces langues s’écrivent de droite à gauche mais peuvent comporter des segments écrits de gauche à droite. En WordprocessingML, le sens d’écriture peut être géré au niveau d’un paragraphe (§4:2.3.1.6) et au niveau d'un "run" (zone d'éléments aux propriétés identiques) à l'intérieur d'un paragraphe (§4:2.3.2.28). De même, dans un texte en DrawingML, il peut être contrôlé au niveau du corps du texte (§4:5.1.5.1.1), au niveau d’un paragraphe (§4:5.1.5.2.2) et dans les listes à puces numérotées (§4:5.1.5.4).

*Flux de texte*: en WordprocessingML, le sens du flux de texte peut être géré au niveau d’une section ou d’un tableau (§4:2.3.1.41) ou au niveau d’un paragraphe (§4:2.3.2.28). Aux niveaux de la section et du tableau, le flux de texte peut être contrôlé verticalement et horizontalement. OpenXML gère ainsi toutes les mises en page possibles (ex. : lignes verticales orientées de haut en bas et de gauche à droite, pour le mongol). Cette capacité se répercute sur la mise en page des listes, tableaux et autres éléments de présentation. DrawingML utilise également les paramètres kumimoji au niveau des paragraphes et des "run" pour orienter le texte horizontalement et les nombres, verticalement (§4:5.1.5.2.3, §4:5.1.5.3.9). En WordprocessingML (§4:2.3.1.16) et PresentationML (§4:4.3.1.15), le flux de caractères peut également être spécifié à l’aide des paramètres kinsoku, pour préciser les caractères autorisés en tête ou en fin de ligne de texte.

*Représentation des nombres*: pour le formatage des champs en WordprocessingML (§4:2.16.4.3), la numérotation des paragraphes / des listes à puces en WordprocessingML (§4:2.9) et la numérotation en DrawingML (§4:5.1.5.4, §4 :5.1.12.61), les nombres peuvent utiliser plusieurs dizaines de formats, dont hiragana, arabe, abjad, thaï, texte cardinal (ex. : « cent vingt-trois »), chinois, coréen (chosung ou ganada), hébreu, hindi, japonais, roman ou vietnamien. Ces mécanismes prennent également en charge les valeurs à séparation fractionnaire arbitraire (ex. : « 1.00 » / « 1,00 ») et les séparateurs de liste. L’internationalisation du formatage des nombres est particulièrement robuste en SpreadsheetML, qui gère toutes ces fonctionnalités dans les formats de cellule (§4:3.8.30) et dans les références aux données externes (4:3.13.12).

*Représentation des données*: en WordprocessingML (§4:2.18.7) et SpreadsheetML (§4:3.18.5), les dates calendaires peuvent être écrites dans les formats grégorien (trois variantes), hébreu, hijri, japonais (Emperor Era), coréen (Tangun Era), saka, taïwanais et thaï.

*Formules*: la spécification des formules en SpreadsheetML propose plusieurs fonctions de conversion liées à l’internationalisation, telles que BAHTTEXT (§4 :3.17.7.22), JIS (§4:3.17.7.185) ou ASC (§4:3.17.7.11).

*Identificateurs de langue*: en WordprocessingML (§4:2.3.2.18) et DrawingML (4:5.1.5.3), chaque paragraphe et chaque "run" peut être marqué par un identificateur de langue qui permet à l’application de sélectionner les outils de vérification et autres fonctionnalités spécifiques à la langue. Outre un identificateur pour chaque langue, OpenXML permet la désignation d’un jeu de caractères, d’une famille de police et d’une valeur PANOSE pour aider l’application à choisir un jeu de caractères de substitution approprié lorsque le jeu d’origine n’est pas reconnu localement.

**4.3 Adoption facilitee pour les développeurs**

Un développeur expérimenté peut commencer à écrire des applications simples conformes à OpenXML à peine quelques heures après avoir commencé la lecture de la spécification. Bien que cette spécification décrive un nombre de fonctionnalités important, une application conforme OpenXML n’a pas besoin de toutes les prendre en charge. La déclaration de conformité (§1:2) nécessite simplement qu’un programme utilisateur « ne rejette aucun document conforme du type [attendu] » et qu’un programme producteur « soit capable de produire des documents conformes » (§1:2.5). Elle fournit également des directives d’interopérabilité qui définissent le rôle de la sémantique des éléments (§1:2.6).

Une application conforme peut présenter des fonctionnalités extrêmement ciblées. Par exemple, il peut s’agir d’un programme de traitement par lots qui actualise les avis de copyright dans une série de documents issus d’un traitement de texte, ou d’un logiciel de synthèse vocale qui comprend suffisamment un diaporama pour produire le texte en version sonore lorsque l’utilisateur passe d’une diapositive à l’autre. La structure du format de fichier permet d’écrire de tels programmes sans avoir une connaissance très poussée d’OpenXML. En particulier :

* Le format de fichier respecte des standards établis (XML, ZIP) pour lesquels il existe déjà des outils matures.
* Le format de fichier s’appuie sur les conventions Open Packaging Conventions, qui combinent XML et ZIP à des mécanismes standard pour décrire les relations au sein d’un fichier. De ce fait, il est souvent possible de parcourir le contenu d’un fichier sans connaître la sémantique des balises des différents langages de marquage principaux ou secondaires d’OpenXML.
* Les éléments les plus bas dans l’arborescence XML sont accessibles et peuvent être modifiés sans perturber le reste de la structure.

Dans tous les formats de fichiers, de petits détails parfois inexistants dans formats binaires facilitent la conception d'applications aux fonctionnalités minimales, grâce à la mise en cache de valeurs. Par exemple :

* Sans mettre en œuvre un programme de pagination, une application de type logiciel de lecture pour aveugles peut permettre de naviguer entre les pages en se basant sur les derniers sauts de page calculés (§4:2.3.3.13).
* Sans mettre en œuvre de formules et sans intégrer de source de données externe, un tableur peut exploiter des calculs (§3:3.2.9) et des données externes (§4:3.14 et §4:3.10.1.76) mis en cache.

Un document conforme minimal est d’une grande simplicité. Voir à ce sujet le paragraphe « Document WordprocessingML minimal ».

**4.4 Compacité**

Le format de fichier OpenXML prend en charge la création d’applications hautes performances. Dans ce paragraphe, nous décrivons certains points de conception qui permettent une grande compacité de fichier, accélérant ainsi la manipulation et l’analyse.

Dans les paragraphes suivants, nous décrivons comment une structure de fichier modulaire permet à une application d’accomplir de nombreuses tâches en n’analysant ou en ne modifiant qu’un petit sous-ensemble d’un document. Un fichier OpenXML est traditionnellement stocké dans une archive ZIP pour les besoins de la mise en package et de la compression, conformément aux recommandations de mise en œuvre des conventions OPC.

Cela peut sembler surprenant mais les fichiers OpenXML sont en moyenne 25 % - et parfois jusqu’à 75 % - moins volumineux que leurs homologues binaires. Ainsi, la taille de ce livre blanc serait supérieure de 85 % en format binaire ! Deuxième facteur simple de compacité, notamment lorsqu’une représentation non compressée est requise, la longueur des identificateurs dans le XML. Les noms de balise les plus fréquents sont courts. Les développeurs sont également encouragés à choisir des préfixes courts pour les espaces de nommage. Par exemple, par convention, le préfixe de l’espace de nommage de WordprocessingML est « w ».

La taille du fichier est également réduite en évitant les répétitions dans le format. Une classe d’exemples évite le stockage redondant d’objets volumineux.

* Dans SpreadsheetML, les chaînes de caractères répétées sont conservées dans une table au sein du classeur, et référencées par un index (§3:3.3).
* Dans SpreadsheetML, une formule propagée ou copiée dans plusieurs cellules est enregistrée comme formule « maîtresse » unique dans la cellule supérieure gauche. Les autres cellules de la plage se réfèrent à cette formule par un index de regroupement (§3:3.2.9.2).
* Dans DrawingML, les noms de forme (§4:5.1.12.56), les caractéristiques géométriques de texte (§4:5.1.12.76) et d’autres éléments prédéfinis (§3:5.8, §3:5.9 et §4:5.1.12) sont représentés par nom ou par numéro et non explicitement. Dans ces cas en particulier, les significations des noms et des numéros sont indiquées dans la spécification et non dans le fichier. La représentation choisie est ici le fruit d’un compromis explicite au cours du processus de normalisation. Compacte, elle peut être modifiée au niveau d’abstraction approprié : par exemple, un rectangle peut être remplacé par un ovale en changeant un attribut (§4:5.1.11.18).

Dans une autre classe d’exemples, la hiérarchie est utilisée pour fournir une sémantique d’héritage. Conséquence appréciable, la taille des fichiers s’en trouve réduite, ce qui améliore les performances.

* En WordprocessingML, les styles sont hiérarchiques (§3:2.8.9).
* En DrawingML, les formes sont regroupées de manière hiérarchique (§4:5.1.2.1.20).
* En PresentationML, une hiérarchie par défaut relie les masques de diapositive, les dispositions de diapositive et les diapositives elles-mêmes (§3:4.2).

D’autres aspects d’OpenXML sont également conçus pour favoriser une mise en œuvre efficace. En SpreadsheetML, par exemple, la table des cellules conserve uniquement les cellules non-vides et peut représenter des cellules fusionnées sous forme d’une unité. Les économies réalisées grâce à cette technique sont considérables pour les feuilles de calcul contenant peu de données.

**4.5 Modularité**

Une application peut accomplir de nombreuses tâches en analysant ou en modifiant un petit sous-ensemble d’un document. Trois fonctionnalités du format OpenXML se combinent pour offrir cette modularité :

* Un document n’est pas monolithique. Il est constitué de nombreuses parties.
* Les relations entre ces parties sont elles-mêmes enregistrées dans des parties.
* Le format d’archive ZIP généralement utilisé pour gérer les documents OpenXML permet l'accès indépendant à chaque partie.

Par exemple :

* Une application peut déplacer intégralement une diapositive d'une présentation à une autre, avec les ressources correspondantes (images et dispositions) sans analyser le contenu de la diapositive (§3:13.3.8). Cet exemple utilise des informations dénommées « relations explicites » pour repérer la diapositive et ses ressources. Les relations explicites sont définies par les conventions OPC, et peuvent être analysées sans aucune connaissance de la sémantique des balises de PresentationML (§1:9.2, §2:8.3).
* Une application peut extraire tous les commentaires d’un document WordprocessingML sans analyser son contenu (§1:11.3.2). Cet exemple utilise des informations dénommées « relations implicites » pour repérer les commentaires. Les relations implicites sont spécifiques à OpenXML et nécessitent donc une certaine connaissance du langage de marquage correspondant (§1:9.2).

4.6 MIGRATION HAUTE FIDELITE

OpenXML est conçu pour prendre en charge toutes les fonctionnalités des formats binaires de Microsoft Office 97-2003. On ne saurait trop souligner la difficulté d'un tel objectif et la réussite d'OpenXML à cet égard n'en est que plus frappante. Certains formats, tels que le format PDF, sont conçus pour fournir un "facsimile" visuel d'un document fini à un utilisateur final. Par contraste, OpenXML a pour but de permettre l'édition ou la manipulation future avec le même niveau d'abstraction que celui dont disposait le créateur initial ; ainsi réduire un graphique vectoriel à un bitmap ne satisferait pas cette exigence, de même que transformer une hiérarchie de styles en styles indépendants.

De plus, un document peut contenir une sémantique informatique que le créateur initial souhaiterait préserver, comme par exemple la logique d'une formule dépendant de résultats de calculs intermédiaires, y compris des codes d'erreur ou des règles d'animation qui produisent un comportement dynamique. Ces références à la spécification témoignent de la capacité d'OpenXML à représenter des aspects subtils des formats binaires.

* La description de SpreadsheetML inclut la spécification complète des formules (§4:3.17.7).
* La spécification de WordprocessingML documente les règles au moyen desquelles les propriétés des paragraphes, caractères, numérotations ou tableaux sont composées avec un formatage direct (§3:2.8, particulièrement §3:2.8.10).
* La spécification de PresentationML documente les fonctionnalités d'animation (§3:4.4).

OpenXML permet à de multiples implémentations de se conformer au standard sans avoir à prendre en compte tous les détails. C'est particulièrement important pour les applications impliquant des calculs numériques, tels que la mise en page, le rendu d’effets graphique ou l'évaluation d'une formule. Exiger plus de cohérence qu'il n'est pratiquement réalisable créerait pour les développeurs des obstacles inutiles sur la voie de la conformité. A cet égard, le comité a par exemple pris les décisions suivantes :

* OpenXML définit des effets tels que des rendus de surface (§5.1.12.50) sans contraindre un développeur à se conformer à ces effets au pixel près.
* OpenXML définit des paramètres tels que les marges de page (§4:2.6.11), la police (§4:2.8) et la justification (§4:2.3.1.13). Il permet aux développeurs d'implémenter des algorithmes de rendu différents sous réserve de respecter ces paramètres.
* La spécification des formules de SpreadsheetML (§4:3.17.7) ne tente pas de supprimer les variations apparaissant dans les calculs en virgule flottante car, en général, cela exigerait que les applications conformes implémentent une émulation lente au lieu de pouvoir exploiter du matériel natif. Par contre, elle spécifie le nombre minimum de bits de précision pour les calculs numériques (§4:3.17.5).
* La spécification des formules de SpreadsheetML laisse aussi à la charge de l’implémentation certaines décisions conditionnelles, afin de permettre des innovations futures. Par exemple, elle ne limite pas le nombre d'itérations d'un calcul comme NORMINV (§4:3.17.7.227). (La fonction NORMINV effectue l'inverse de la distribution normale en exécutant une recherche itérative.)

Plusieurs fonctionnalités plus anciennes, telles que VML (§3:6), sont incluses principalement à des fins de rétrocompatibilité. L'utilisation de standards plus récents déjà intégrés dans OpenXML, tels que DrawingML (§3:5), est encouragée lors de l'écriture de nouveaux documents.

**4.7 INTEGRATION AVEC LES DONNEES METIER**

Grâce à l'utilisation de schémas personnalisés au sein des documents OpenXML, les organisations peuvent intégrer des applications de productivité avec les systèmes d'informations qui gèrent les processus métier. Une organisation pourrait adopter cette approche afin de réutiliser et d'automatiser le traitement des informations métier qui, dans le cas contraire, resteraient cachées de manière opaque au sein des documents, inaccessibles aux applications métier. Les applications peuvent être par exemple :

* *Recherche* : un utilisateur pourrait rechercher dans un ensemble de feuilles de calcul toutes les sociétés présentant des marges de profit supérieures à 20 %.
* *Marquage au moyen de métadonnées* : Un cabinet juridique pourrait marquer des présentations approuvées d'un point de vue réglementaire.
* *Assemblage de document* : Un groupe de proposition pourrait rationaliser la génération de proposition en automatisant la préparation des données sous-jacentes.
* *Réutilisation des données* : Un ingénieur commercial pourrait générer un rapport de tous les contrats de ventes passés au cours d'une période donnée, en précisant les clients, les montants des contrats, ainsi que toutes les modifications éventuelles des termes et conditions.
* *Applications sectorielles* : Les professionnels d'un secteur d'activité particulier pourraient préparer des livrables dans un environnement d’édition familier, tout en voyant les produits de leur travail automatiquement injectés dans les systèmes métier.

Pour parvenir à ces objectifs, il convient de définir la structure et le type de données qu'une classe de documents peut contenir, et d'autoriser l'accès à l'information partout où elle se produit naturellement dans le flux de chaque document. Considérez l'exemple simple d'un curriculum vitae. Nous pourrions définir une structure de données incluant des champs appelés nom, numéro de téléphone, adresse, objectifs de carrière et qualifications. Il faudrait ensuite faire en sorte que ces champs apparaissent là où des auteurs humains les mettraient dans un document. Dans un autre contexte métier, tel qu'un groupe financier ou un centre médical, la structure et les champs de données seraient différents. OpenXML permet à ce processus de se produire de manière standardisée.

Premièrement, la structure des données métier est d'abord exprimée à l'aide d'un schéma XML personnalisé. Cela permet à une organisation d'exprimer des données avec des balises significatives d'un point de vue métier. Une organisation peut créer ses propres schémas, ou utiliser des schémas standard de l'industrie tels XBRL pour le reporting financier (7) ou HL7 pour les informations médicales (8). Des schémas sont créés dans le secteur public, dans les entreprises, et en tant que standards industriels, pour gérer aussi bien des certificats de naissance que des informations d'assurance. Tout schéma personnalisé peut être utilisé, à condition qu'il soit exprimé au format XSD (2).

Deuxièmement, les données personnalisées sont encapsulées dans tout document OpenXML au sein d’une partie Custom XML (§3.7.3) et peuvent être décrites à l'aide d'une partie Custom XML Data Properties (Propriétés de données XML personnalisées) (§4:7.5). En séparant ces données personnalisées de la présentation, OpenXML permet d’intégrer proprement des données, tout en rendant possible la présentation et la manipulation par l'utilisateur final dans une large variété de contextes, y compris documents, formulaires, diapositives et feuilles de calcul. L'interopérabilité est ainsi assurée à un niveau plus fondamental et sémantiquement plus précis.

**4.8 Un potentiel d'innovation**

OpenXML est conçu pour encourager les développeurs à créer de nouvelles applications, des applications que personne n'avait envisagé lorsque les formats binaires ont été définis, ni même quand OpenXML a été défini. Tout d'abord, nous discuterons des mécanismes d'extensibilité dont la synergie assure l'interopérabilité entre des applications offrant pourtant des ensembles de fonctionnalités différents. Considérez une application de *niveau supérieur* (qui contient une nouvelle fonctionnalité non documentée dans OpenXML) et une application de *niveau inférieur* (qui ne comprend pas cette fonctionnalité). Les trois objectifs essentiels de l'extensibilité sont les suivants :

* *Fidélité visuelle* : la capacité pour l'application de niveau inférieur à afficher ce que l'application de niveau supérieur afficherait. Cela requiert de manière inhérente qu'un fichier stocke de multiples représentations des mêmes données.
* *Capacité de modification* : la possibilité d'éditer une ou plusieurs des représentations.
* *Confidentialité* : la possibilité de garantir qu'il n'existe plus d'anciennes versions d'une représentation après l'édition d'une autre représentation, laissant de manière inattendue des informations qu'un utilisateur croit supprimées ou modifiées. Une application peut satisfaire cet objectif en éliminant ou en synchronisant les représentations.
* Un développeur souhaitant étendre le jeu de fonctions d'OpenXML dispose principalement de deux options :
* les blocs de contenu alternatif : Un bloc de contenu alternatif (§3:2.18.4 et §5:9.2) stocke de multiples représentations du même contenu, chacun à l'intérieur de son propre bloc de choix. Une application de niveau inférieur lit un bloc de choix qu'elle est capable de lire. Lors d'une édition, elle écrit autant de blocs de choix qu'elle est capable d'en écrire.
* les listes d'extension : Une liste d'extension (§3:2.6) stocke du XML personnalisé arbitraire sans représentation visuelle.

Les développeurs ont également toute latitude pour innover en dehors de ces mécanismes d'extensibilité.

* *Paradigmes d'interaction alternatifs*. OpenXML va au-delà de la simple définition syntaxique d'un document, mais sans aller jusqu'à définir le comportement d'une application. Comme le décrit la déclaration de conformité, il met l'accent sur la sémantique (§1:2.2, §1.2.3). Par conséquent, une application dite conforme est libre de communiquer avec un utilisateur final par une grande diversité de moyens, ou de ne pas communiquer du tout - à condition de respecter la sémantique spécifiée.
* *Nouveaux environnements informatiques*. La déclaration de conformité admet des applications à faible capacité, qui peuvent donc s'exécuter sur de petits terminaux, et des applications qui n'implémentent qu'un sous-ensemble d'OpenXML (§1:2.6). Le mécanisme des caractéristiques additionnelles permet à une application productrice de communiquer ses limites de capacité (§3:8.1).

Comme nous l'avons vu dans le paragraphe précédent, certaines des opportunités les plus importantes en matière d'innovation ne concernent pas le rendu de documents à des fins d'interaction directe avec l'utilisateur. Au contraire, elles impliquent un traitement de machine à machine en utilisant des formats de message XML, par exemple via des services Web XML(9). Bien que de telles applications n'aient pas de comportement visible par l'utilisateur autre que leurs opérations sur les données contenues au sein des documents OpenXML, elles sont soumises à la conformité de niveau document (§1:2.4) et à la conformité de niveau application (§1:2.5), qui sont purement syntaxiques, et aux principes d'interopérabilité (§1:2.6), qui incorporent la sémantique.

S'il est impossible d'énumérer tous les cas d'utilisation possibles de traitement de données XML personnalisées, on peut envisager des services traitant des documents OpenXML à des fins d'extraction et d'insertion automatique de données personnalisées, des services de sécurité personnalisés tels que la signature numérique XML (*XML Digital Signature*, 10) ou le chiffrement XML (*XML Encryption*, 11) ou même des transformations XSLT arbitraires (12) effectuant des conversions vers et à partir d'autres formats XML. OpenXML ne fixe aucune limitation ou interdiction sur de tels traitements.

**5 STRUCTURE D'UN DOCUMENT OFFICE OPEN XML**

L'un des principaux objectifs de ce livre blanc est de permettre au lecteur de suivre la structure de haut niveau de tout fichier OpenXML. Pour parvenir à cette fin, nous ne fournissons qu'un niveau limité de détails concernant les conventions OPC, et moins encore concernant les différents langages de marquage.

**5.1 CONVENTIONS OPEN PACKAGING**

Les conventions OPC offrent un mécanisme pour stocker de multiples types de contenus (par ex. XML, images et métadonnées) dans un conteneur, tel qu'une archive ZIP, afin de représenter un document dans sa totalité. Elles décrivent un modèle logique pour représenter les imbrications et les relations. La mise en oeuvre recommandée pour les OPC utilise le format d'archive ZIP.

Il est possible d'inspecter la structure de tout fichier OpenXML en utilisant n'importe quel visualisateur ZIP. Il peut être intéressant d'inspecter le contenu d'un petit fichier OpenXML de cette manière pendant la lecture de cette description. Sous le système d'exploitation Windows, il suffit simplement d'ajouter une extension “.zip” au nom de fichier et de double-cliquer. Sur le plan logique, un document OpenXML est un *package* OPC (§5:8). Un package est un ensemble de *parties* (§5:8.1).

Chaque partie a un *nom de partie* insensible à la casse qui se compose d'une séquence de noms de segments délimités par des barres obliques, par exemple “/pres/slides/slide1.xml” (§5:8.1.1). Chaque partie a également un *type de contenu* (§5:8.1.2). Physiquement, l'archive ZIP est un package, chaque entrée ZIP dans l'archive est une partie, et les noms des chemins dans l'archive ZIP correspondent directement aux noms des parties.

Dans l’implémentation ZIP, “/[Content\_Types].xml” permet à un consommateur de déterminer le type de contenu de chaque partie du package (§2:9.2.6). La syntaxe et la définition des types de médias suivent la section 3.7 de la RFC 2616 (13). Les packages et parties peuvent contenir des *relations explicites* (§1:9.2) avec d'autres parties au sein du package, ainsi qu'avec des ressources externes.

Chaque relation explicite comporte un ID de relation, qui permet au contenu d'une partie d'y faire référence ; et d'un type, qui permet à une application de décider comment la traiter. Les types de relations sont nommés avec des URI, permettant à des parties non coordinatrices de créer de nouveaux types de façon sûre et sans conflit. L'ensemble des relations explicites pour un package source ou une partie source donnée est stocké dans une *partie relations*. La partie relations d'un package dans son ensemble est appelée “/\_rels/.rels” ; la partie relations d'une partie nommée “/a/b/c.xml” est appelée “/a/b/\_rels/c.xml.rels”. La partie relations (et, dans l'implémentation ZIP, la partie type de contenu, ou *content type*) sont les seules parties spécifiquement nommées dans un package.

Pour ouvrir un package, une application doit analyser la partie relations du package et suivre les relations du type approprié. Toutes les autres parties d'un document OpenXML contiennent de l'OpenXML, du XML personnalisé ou du contenu de type arbitraire comme des objets multimédia. La possibilité pour une partie de contenir du XML personnalisé est un mécanisme particulièrement puissant pour intégrer des données métier et des métadonnées.

**5.2 WORDPROCESSINGML**

Un document WordprocessingML est composé d'une collection d'articles (*stories)* (§3:2.1). Chaque article peut être : le document principal (§3:2.2), le document glossaire (§3:2.13), un sous-document (§3:2.18.2), un en-tête (§3:2.11.1), un pied de page (§3:2.11.2), un commentaire (§3:2.14.5), un cadre, une zone de texte (§3:2.18.1), une note de bas de page (§3:2.12.1), ou une note de fin (§3:2.12.2). Le seul article obligatoire est le document principal. Il s'agit de la cible de la relation de package dont le type est : <http://schemas.openxmlformats.org/officeDocument/2006/relationships/officeDocument>

Un chemin type de la racine à la feuille dans l'arborescence XML comprendrait les éléments XML suivants (§3:2.2) :

* document – l'élément racine du document principal (§3:2.3).
* body – le corps (§3:2.7.1). Peut contenir de multiples paragraphes. Peut également contenir des propriétés de section spécifiées dans un élément sectPr.
* p – paragraphe (§3:2.4.1). Peut contenir un ou plusieurs "run". Peut également contenir des propriétés de paragraphe spécifiées dans un élément pPr, pouvant lui-même contenir des propriétés de run (également appelées propriétés de caractère) par défaut spécifiées dans un élément rPr (§3:2.4.4).
* r – run (§3:2.4.1). Peut contenir de multiples types de contenu de run, principalement des plages de texte. Peut également contenir des propriétés de run (rPr). Le run est un concept fondamental dans OpenXML. Un run est une zone contiguë de texte présentant des propriétés identiques ; un run ne contient pas de balise de texte supplémentaire. Par exemple, la phrase contenant les mots “Voici effectivement **trois** run” serait représentée par au moins trois run : “Voici effectivement ”, “**trois**” et “ run”. A cet égard, OpenXML diffère de manière significative des formats autorisant une imbrication arbitraire des propriétés, tels que le HTML.
* t – plage de texte (§3:2.4.3.1). Contient une quantité arbitraire de texte sans formatage, sauts de ligne, tableaux, graphiques ou autres éléments non textuels. Le formatage utilisé pour le texte est hérité des propriétés du run et des propriétés du paragraphe. Cet élément utilise souvent l'attribut xml:space=“preserve”.

Dans ce paragraphe, nous avons évoqué la notion de formatage direct du texte par spécification de propriétés de paragraphe et de run. Le formatage direct est pris en compte à la fin d'une séquence d'application qui comprend également les styles de caractère, de paragraphe, de numérotation et de tableau, ainsi que des valeurs de document par défaut (§3:2.8.10). Ces styles sont eux-mêmes organisés en hiérarchies d'héritage (§3:2.8.9). Le paragraphe "Document WordprocessingML minimal" ci-dessous présente un document WordprocessingML dans son intégralité.

**5.3 PRESENTATIONML**

Un document PresentationML est décrit par une partie présentation. La partie présentation est la cible de la relation de package dont le type est : <http://schemas.openxmlformats.org/officeDocument/2006/relationships/officeDocument>

La présentation fait référence aux constructions primaires (§3:4.2) que nous énumérons de manière descendante dans la hiérarchie par défaut :

* des masques de diapositive (*slide masters*), masques de pages de commentaires (*notes masters*) et masques de documents (*handout masters*) (§3:4.2.2), qui héritent tous des propriétés de la présentation ;
* des dispositions de diapositive (*slide layouts*, §3:4.2.5), qui héritent des propriétés du masque de diapositive ; et
* des diapositives (§3:4.2.3) et pages de commentaires (§3:4.2.4), qui héritent respectivement des propriétés des dispositions de diapositive et des masques des pages de commentaires.

Chaque masque, disposition et diapositive est stocké dans sa propre partie. Le nom de chaque partie est spécifié dans la partie relation pour la partie présentation.

Chacune des six parties en plus de la présentation est structurée essentiellement de la même manière. Un chemin type de la racine à la feuille dans l'arborescence XML comprendrait les éléments XML suivants (§3:2.2) :

* sld, sldLayout, sldMaster, notes, notesMaster ou handoutMaster – l'élément racine.
* cSld – diapositive (§4:4.4.1.15). Peut contenir des éléments DrawingML (ainsi que décrit dans les deux puces qui suivent) et d'autres éléments structurels (ainsi que décrit ci-dessous)
* spTree – arborescence de forme (§4:4.4.1.42). Peut contenir des propriétés de formes groupées dans un élément grpSpPr (§4:4.4.1.20) et des propriétés de formes groupées non visuelles dans un élément nvGrpSpPr (§4:4.4.1.28). Ce noeud et ses descendants sont tous des éléments DrawingML. Nous citons quelques éléments DrawingML ici du fait de leur rôle essentiel dans PresentationML.
* sp – forme (§4:4.4.1.40). Peut contenir des propriétés de forme dans un élément spPr (§4:4.4.1.41) et des propriétés de forme non visuelle dans un élément nvSpPr (§4:4.4.1.31).

En plus du contenu de forme DrawingML, un cSld peut contenir d'autres éléments structurels, selon l'élément racine dans lequel il réside, ainsi que le résume le tableau suivant :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Diapositive | Disposition de diapositive | Masque de diapositive | Masque de document | Masque des commentaires | Page de commentaires |
| Common Data | X | X | X | X | X | X |
| Transition | X | X | X |  |  |  |
| Timing | X | X | X |  |  |  |
| Headers and Footers |  | X | X | X | X |  |
| Matching Name |  | X |  |  |  |  |
| Layout Type |  | X |  |  |  |  |
| Preserve |  | X | X |  |  |  |
| Layout List |  |  | X |  |  |  |
| Text Style |  |  | X |  |  |  |

Les propriétés spécifiées par des objets à un niveau inférieur dans la hiérarchie par défaut (masque de diapositive, disposition de diapositive, diapositive) remplacent les propriétés correspondantes spécifiées par les objets situés à un niveau supérieur dans la hiérarchie. Par exemple, si aucune transition n'est spécifiée pour une diapositive, c’est celle de la disposition de la diapositive qui est utilisée, et en l'absence de transition au niveau de la disposition, c’est celle du masque de la diapositive qui est utilisée.

**5.4 SPREADSHEETML**

Un document SpreadsheetML est décrit au niveau supérieur par une partie classeur (workbook). La partie classeur est la cible de la relation de paquet dont le type est : <http://schemas.openxmlformats.org/officeDocument/2006/relationships/officeDocument>

La partie classeur stocke des informations sur le classeur et sa structure, comme la version de fichier, l'application auteur, ainsi que le mot de passe de modification. Du point de vue logique, le classeur contient une ou plusieurs feuilles (§3:3.2) ; physiquement, chaque feuille est stockée dans sa propre partie et est référencée de manière habituelle à partir de la partie classeur. Chaque feuille peut être une feuille de calcul, une feuille de graphique ou une feuille de dialogue. Nous ne discuterons ici que de la feuille de calcul, qui est le type le plus courant. Dans un objet feuille de calcul, un chemin type de la racine à la feuille dans l'arborescence XML comprend les éléments XML suivants :

* worksheet – l'élément racine dans une feuille de calcul (§3:3.2).
* sheetData – la table des cellules, qui représente toutes les cellules non vides dans la feuille de calcul (§3:3.2.4).
* row – une ligne de cellules dans la table des cellules (§3:2.8).
* c – une cellule (§3:3.2.9). L'attribut r indique l'emplacement de la cellule par des coordonnées de type A1. La cellule peut également avoir un identificateur de style (attribut s) et un type de données (attribut t).
* v et f – la valeur (§3:3.2.9.1) et le cas échéant la formule (§3:3.2.9.2) de la cellule. Si une cellule comporte une formule, alors la valeur est le résultat du calcul le plus récent.

Les chaînes de texte et les formules sont stockées dans des tables partagées (§3:3.3 et §3:3.2.9.2.1) afin d'éviter un stockage redondant et accélérer les chargements et les enregistrements.

**5.5 Langages de marquage secondaires**

Plusieurs langages de marquage secondaires peuvent également être utilisés pour décrire le contenu d'un document OpenXML.

* DrawingML (§3:5) – utilisé pour représenter des formes et d'autres objets rendus de manière graphique au sein d'un document.
* VML (§3:6) – un format pour les graphiques vectoriels inclus à des fins de rétrocompatibilité, destiné à terme à disparaître au profit de DrawingML.
* des langages de marquage partagés : Math (§3:7.1), Metadata (§3:7.2), Custom XML (§3:7.3) et Bibliography (§3:7.4).

**5.6 DOCUMENT WORDPROCESSINGML MINIMAL**

Ce paragraphe contient un document WordprocessingML minimal composé de trois parties.

La partie type de contenu “/[Content\_Types].xml” décrit les types de contenus des deux autres parties requises.

<Types xmlns="http://schemas.openxmlformats.org/package/2006/content-types"> <Default Extension="rels" ContentType="application/vnd.openxmlformats-package.relationships+xml"/> <Default Extension="xml" ContentType="application/vnd.openxmlformats-officedocument.wordprocessingml.document.main+xml"/> </Types>

La partie relation de package “/\_rels/.rels” décrit la relation entre le package et la partie document principal.

<Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships"> <Relationship Id="rId1"

Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/officeDocument" Target="document.xml"/> </Relationships>

La partie document, dans ce cas “/document.xml”, contient le contenu du document. <w:document xmlns:w="http://schemas.openxmlformats.org/wordprocessingml/2006/main"> <w:body> <w:p> <w:r> <w:t>Hello, world.</w:t> </w:r> </w:p> </w:body> </w:document>

La spécification fournit les documents minimum nécessaires et des informations supplémentaires sur WordprocessingML (§1:11.2), PresentationML (§1:13.2) et SpreadsheetML (§1:12.2).

**6 RESUME**

OpenXML est l'aboutissement d'un effort conséquent mis en œuvre par des représentants de nombreuses industries et institutions publiques avec des contextes et des intérêts organisationnels très divers.

Il couvre l'ensemble du spectre des fonctionnalités utilisées dans le corpus documentaire existant, ainsi que les besoins d'internationalisation inhérents à tous les groupes de langues majeurs dans le monde.

Conséquence du travail de normalisation effectué par l'Ecma TC45 (1) et des contributions par commentaires publics, OpenXML permet un niveau élevé d'interopérabilité et d'indépendance de plateforme. Sa documentation est aujourd'hui à la fois complète (grâce à des documents de référence exhaustifs) et accessible (via des descriptions non normatives). Elle inclut également suffisamment d'informations pour permettre à des technologies d’accessibilité de traiter correctement des documents.

Une implémentation OpenXML peut être très compacte et fournir des fonctionnalités très spécialisées, ou englober tout le spectre de fonctionnalités. Les mécanismes d'extensibilité intégrés au format garantissent les possibilités d'innovation. Le fait de normaliser la spécification du format et de la maintenir au fil du temps est un gage de pérennité pour de nombreux utilisateurs, qui savent aussi pouvoir compter sur des évolutions futures maîtrisées dans le cadre d'un processus de normalisation ouvert.

La nécessité d'un standard de format de document ouvert, capable de préserver les milliards de documents créés dans les formats binaires préexistants et les milliards d'autres qui continuent à être créés chaque année s'impose de plus en plus chaque jour. Grâce aux progrès technologiques aux niveaux matériel et réseau, ainsi qu'à une infrastructure logicielle fondée sur les standards, c'est désormais possible. Avec la diversification explosive de la demande et les investissements existants importants en systèmes de gestion stratégiques, cela devient essentiel.

**7 CITATIONS**

1. **Ecma international.** TC45 - Office Open XML Formats. *Ecma international.* [En ligne] <http://www.ecma-international.org/memento/TC45.htm>.

2. **W3C.** XML Schema. *World Wide Web Consortium.* [En ligne] <http://www.w3.org/XML/Schema>.

3. **ISO.** ISO/IEC 19757-2:2003. *International Organization for Standardization.* [En ligne] <http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=37605&ICS1=35&ICS2=240&ICS3=30>.

4. **ISO.** ISO/IEC 19757-4:2006. *International Organization for Standardization.* [En ligne] <http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=38615&ICS1=35&ICS2=240&ICS3=30>.

5. **W3C.** Extensible Markup Language (XML) 1.0 (Quatrième édition). *World Wide Web Consortium.* [En ligne] 2006. <http://www.w3.org/TR/2006/REC-xml-20060816/>.

6. **W3C.** Namespaces in XML 1.0 (Deuxième édition). *World Wide Web Consortium.* [En ligne] 2006. <http://www.w3.org/TR/2006/REC-xml-names-20060816/>.

7. **XBRL International.** XBRL Specifications. *Extensible Business Reporting Language.* [En ligne] <http://www.xbrl.org/Specifications/>.

8. **Health Level Seven.** HL7 ANSI-Approved Standards. *Health Level Seven.* [En ligne] <http://www.hl7.org/about/directories.cfm?framepage=/documentcenter/public/faq/ansi_approved.htm>.

9. **W3C.** W3C Web Services Architecture. *World Wide Web Consortium.* [En ligne] 2002. <http://www.w3.org/2002/ws/>.

10. **W3C.** W3C XML Signature. *World Wide Web Consortium.* [En ligne] <http://www.w3.org/Signature/>.

11. **W3C.** W3C XML Encryption. *World Wide Web Consortium.* [En ligne] 2001. <http://www.w3.org/Encryption/2001/>.

12. **W3C.** XSL and XSLT. *World Wide Web Consortium.* [En ligne] <http://www.w3.org/Style/XSL/>.

13. **W3C.** Hypertext Transfer Protocol -- HTTP/1.1. *World Wide Web Consortium.* [En ligne] http://www.w3.org/Protocols/rfc2616/rfc2616.html.