Place de SGML parmi les nouvelles architectures documentaires

Yves Marcoux[1]
<GRDS> - EBSI
Université de Montréal

1. Introduction

La notion de document est aujourd'hui soumise à l'action de plusieurs forces qui tirent leur origine de mutations profondes dans le monde de l'informatique et de la technologie. Non seulement on assiste à la prolifération de formats " traditionnels " de documents, propriétaires et/ou spécifiques à un type d'applications, mais également, des formes radicalement nouvelles de documents voient le jour; le document devient un objet complexe, ayant sa vie et son comportement propres, capable d'interagir avec son créateur, son lecteur, son environnement.

Simultanément, SGML (Standard Generalized Markup Language; norme ISO 8879), un format normalisé de documents, connaît un essor phénoménal. Un nombre chaque jour croissant d'institutions, de projets, d'organismes, prennent le virage SGML dans le but de protéger leur capital-information des aléas de l'évolution technologique.

Quelles sont les forces en action sur la notion de document? Comment SGML se situe-t-il par rapport aux nouvelles formes de documents en émergence? Comment se compare-t-il à elles? Telles sont les questions que nous abordons dans cet article. L'article est structuré comme suit: nous rappelons d'abord les principes de base de SGML; puis, nous analysons les forces en action sur la notion de document et présentons les nouvelles formes de documents qui voient le jour en réaction à ces forces; finalement, nous discutons de la place que peut et devrait occuper SGML dans ces nouvelles formes de documents ainsi que du rôle qu'il peut et devrait jouer dans leur évolution.

2. Principes de base de SGML

SGML est un format de documents électroniques, ce qui revient à dire: un langage informatique de description de documents. La notion de documents que sous-tend SGML est extrêmement générale et recouvre potentiellement toute forme d'information électronique: documents de logiciels de traitement de texte, feuilles de calcul de tableurs, documents multimédias, hypertextes, tables de bases de données et même, logiciels. Malgré cette généralité de SGML, nous utiliserons, par souci de simplicité, un exemple très classique pour illustrer les principes de base de SGML: un mémorandum. La Figure 1 représente un tel document sur papier.

[1] Adresses Internet: yves.marcoux@umontreal.ca; http://mapageweb.umontreal.ca/marcoux/

MÉMORANDUM

De: Julia Royer
À: Jean Picard
Émilie Dugré
Sujet: Invitation

Veuillez noter que ... le 27 septembre 1996.

SVP, avisez-moi ... dans les plus brefs délais.

Figure 1 - Exemple de mémorandum sur papier

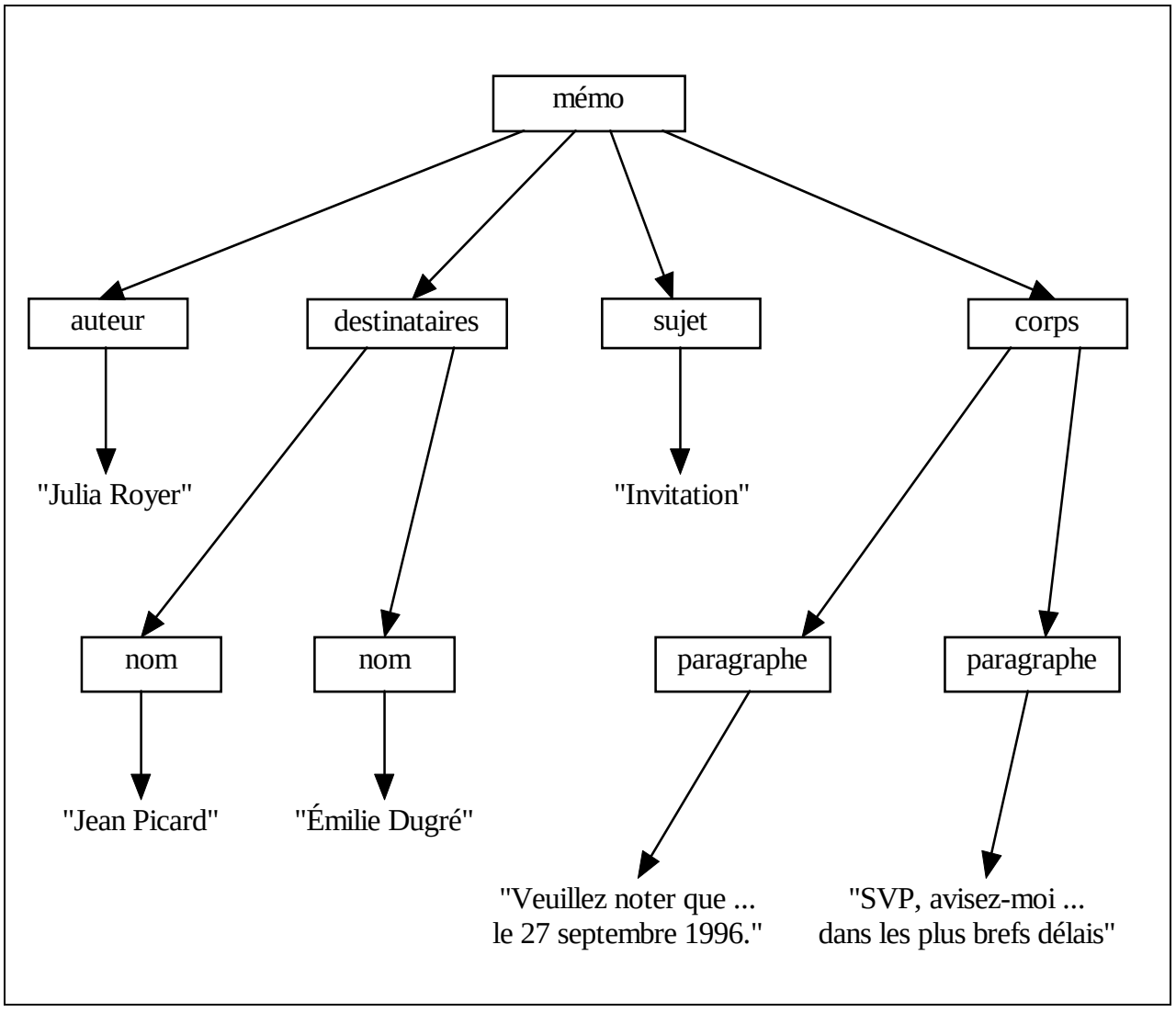

Figure 2 - Structure logique du mémo de la Figure 1

```
<mémo>
<auteur> Julia Royer </auteur>
<dest>
<nom> Jean Picard </nom>
<nom> Émilie Dugré </nom>
</dest>
<sujet> Invitation </sujet>
<corps>
<par> Veuillez noter que ... le 27 septembre 1996. </par>
<par> SVP, avisez-moi ... dans les plus brefs délais. </par>
</corps>
</mémo>
```

Figure 3 - Texte SGML pour le document de la Figure 1

SGML se préoccupe de représenter dans des documents électroniques la *structure logique* de l'information qui y est contenue, plutôt que les caractéristiques qui relèvent de la présentation sur papier, écran d'ordinateur, etc. La Figure 2 présente de façon graphique la structure logique de l'information contenue dans le mémo de la Figure 1; il s'agit d'une structure hiérarchique composée des différents éléments d'information retrouvés dans le mémo. Notez que des caractéristiques comme la présence du mot " mémorandum " centré en gras sur la première ligne de la page ne se retrouvent pas dans cette structure logique, puisqu'il ne s'agit pas d'un élément de contenu du mémo. Il s'agit plutôt d'une façon d'identifier le type de document auquel on a affaire. Ce rôle est joué dans la structure logique par la présence de l'élément " mémo " en tête de hiérarchie.

L'univers sémantique de SGML, en tant que langage informatique, est celui des documents structurés, c'est-à-dire de documents dont le contenu est vu comme une structure logique d'information du genre illustré en Figure 2. Une telle structure est le plus souvent hiérarchique (comme dans notre exemple), mais SGML permet aussi l'établissement de liens non-hiérarchiques entre les différents éléments de structure (mécanisme `ID - IDREF`).

La syntaxe de SGML est basée sur le *balisage*. Le balisage (en anglais *markup* ou *tagging*) consiste à insérer dans un document électronique de courtes chaînes de caractères, appelées *balises*, qui indiquent soit le début, soit la fin d'une partie du document. Dans la *syntaxe concrète de référence* de SGML (définie dans la norme ISO 8879), les balises d'ouverture (début) sont de la forme <*id-gén*> et celles de fermeture (fin) </*id-gén*>. Les noms que l'on retrouve à l'intérieur des balises s'appellent des *identificateurs génériques* et leur rôle est en quelque sorte d'identifier le "type" d'information que l'on retrouve entre la paire de balise. La portion de document comprise entre une balise d'ouverture et une balise de fermeture correspondante s'appelle un *élément* SGML. Dans la représentation SGML d'un document structuré, les relations hiérarchiques entre les éléments de structure logique du document sont traduites en des relations d'*imbrication* des éléments SGML. En général, une balise d'ouverture peut contenir autre chose qu'un identificateur générique, notamment, des *attributs*. Un attribut vient *qualifier*

un élément; il peut par exemple indiquer un *sous-type* d'information par rapport au type principal associé à l'identificateur générique.

SGML doit pouvoir représenter n'importe quel document. Faut-il donc que la norme prévoie tous les types d'information pouvant se retrouver dans tous les types de documents? Cela est évidemment impossible, puisqu'on pourra toujours inventer de nouveaux types de documents et d'information. La solution est qu'en fait, SGML est un "méta-langage", qui permet de *définir* les jeux de balises que l'on désire utiliser dans les documents d'un certain type, de même que les règles syntaxiques d'utilisation de ces balises. C'est cette caractéristique de SGML qui en fait un langage de "balisage généralisé".

La définition d'un jeu de balises et des règles syntaxiques associées s'appelle une *DTD*, pour *Document Type Definition*.[2] Le langage dans lequel on exprime les DTD fait partie intégrante de SGML et il utilise lui aussi le balisage (mais d'une forme particulière). Tout document SGML possède un prologue (possiblement implicite, mais néanmoins existant) dans lequel la DTD à laquelle il se conforme est identifiée. La DTD peut être directement incluse dans le prologue ou elle peut simplement y être nommée. (Notez que le prologue du document n'est pas inclus dans la Figure 3.) À titre d'illustration, la Figure 4 présente un prologue incluant une DTD qui pourrait être utilisée pour des documents de type mémo et à laquelle le document de la Figure 3 est conforme.

```
<!DOCTYPE mémo [
<!ELEMENT mémo - -
     ((auteur & (date?) & sujet & dest & (cc?)), corps)>
<!ELEMENT (dest | cc) - - (nom+)>
<!ELEMENT corps - - (par*)>
<!ELEMENT (auteur | date | sujet | nom | par) - - (#PCDATA)>
]>
```

Figure 4 - Un prologue incluant une DTD pour les mémos

Le fait de pouvoir spécifier des règles d'utilisation des balises dans une DTD (quel élément peut être imbriqué dans quel autre élément, etc.) est un aspect très important de SGML. En effet, cela permet d'imposer une uniformité aux documents d'un même type, ce qui rend plus facile l'exploitation de l'information contenue dans les documents.

SGML ne s'intéresse pas du tout à la façon dont les différentes balises contenues dans un document doivent être interprétées. Il n'y a aucun moyen de spécifier cette information en SGML. Pour pouvoir effectuer un traitement sur des documents SGML (par exemple, imprimer notre mémo de la Figure 3), il faut définir une *application de traitement*. C'est l'application de traitement seule qui détermine comment sont interprétées les différentes balises. Cette

[2] Rigoureusement parlant, une DTD est une notion plus générale. Cependant, la définition que nous utilisons est de loin la plus répandue.

séparation entre contenu et traitement est un des avantages cruciaux de SGML, puisque, comme nous verrons plus loin, elle permet la *réutilisation* de l'information contenue dans les documents.

SGML permet la représentation de la structure logique des documents par balisage, mais il ne peut pas l'imposer, puisque ce sont les applications de traitement qui déterminent la signification des balises. Ainsi, rien ne peut empêcher l'utilisation de balises pour représenter des opérations de formatage; c'est d'ailleurs largement le cas dans HTML (Hypertext Markup Language), le principal langage du World Wide Web (WWW). Lorsque effectivement les balises SGML sont utilisées pour représenter la structure logique de l'information contenue dans des documents, on parle de balisage *descriptif* ou *logique*. Ce type de balisage s'oppose au balisage *procédural*, qui caractérise la plupart des formats de documents électroniques courants, et dans lequel les balises sont associées à des opérations spécifiques à effectuer sur l'information, par exemple, des directives de formatage (gras, italique, centré, etc.)

Le balisage descriptif, combiné à la séparation contenu-application inhérente à SGML, permet la *réutilisation* de l'information contenue dans les documents. Lorsqu'on veut développer une nouvelle application pour traiter des documents existants, la présence des balises descriptives permet de retrouver rapidement et facilement l'information pertinente à la nouvelle application; de plus, les documents ne sont pas " encombrés " par une foule de balises correspondant à des opérations qui n'ont aucun intérêt pour la nouvelle application.

3. Les forces en action sur la notion de document

L'évolution du document est aujourd'hui influencée par trois grands changements de paradigme qui affectent toute l'industrie des technologies de l'information. Ces trois grands changements sont: (1) le passage de la notion de communication globale à celle de *coopération* globale, (2) la transformation du rôle de l'interface-utilisateur qui, du moyen de communication entre une application et un utilisateur qu'elle était, devient graduellement un outil de contrôle et d'intégration des applications par l'utilisateur, (3) le passage de la notion d'information comme bien de consommation, ou denrée périssable, à celle d'information en tant qu'actif organisationnel et social.

3.1 De la communication globale à la coopération globale

Avec l'augmentation de la puissance des ordinateurs et des moyens de communication, une part de plus en plus importante des activités jusqu'à présent réalisées ou séquencées par l'humain est maintenant prise en charge par des ordinateurs et/ou des réseaux de télécommunication. Déjà, une part importante des activités d'échange d'information est assurée par des réseaux informatiques globaux comme l'Internet: le World Wide Web, l'EDI (échange de documents informatisés) et le commerce électronique en général illustrent bien cette tendance. Les futures " inforoutes " devraient encore augmenter les possibilités de soutien technologique aux communications (vidéo sur demande, etc.)

Cette augmentation de la puissance des moyens technologiques disponibles pour soutenir l'activité humaine fait en sorte que la complexité des activités prises en charge va naturellement en augmentant. Jusqu'à présent, les applications informatiques distribuées assuraient surtout le transfert d'information relativement *statique* entre humains, alors que l'information très

dynamique, essentielle à la coopération entre humains, circulait surtout par des moyens extérieurs aux applications: téléphone, télécopie, etc. Aujourd'hui, les utilisateurs demandent des applications distribuées qui intègrent non seulement des moyens de partager l'information statique, mais également des mécanismes sophistiqués de coopération. Ainsi, on voudrait pouvoir recevoir de l'information, l'analyser rapidement avec des logiciels spécialisés, effectuer des modélisations ou des simulations en temps réél, demander l'avis d'experts ou de collaborateurs éloignés, prendre des décisions et finalement, rédiger de façon collaborative un rapport et une note de service sur ces activités, en y incluant les résultats des analyses ou des simulations réalisées, le tout sur un même poste de travail et le plus facilement possible.

Ces nouvelles préoccupations se réflètent par exemple dans l'intégration grandissante des moyens de communication " classiques " (téléphone, télécopie, vidéo) à l'informatique, dans l'essor que connaissent actuellement les logiciels de groupe comme Notes, de même que dans les activités de recherche et de développement sur les moyens de soutenir technologiquement la coopération, plus particulièrement dans le domaine du travail collaboratif assité par ordinateur (Computer Supported Cooperative Work; CSCW).

Cette vision des applications distribuées exploite l'augmentation de l'intelligence côté client (client-side intelligence) à laquelle on assiste depuis l'avènement des architectures client-serveur. Les données reçues par un utilisateur doivent pouvoir être traitées par lui, en partie selon des paramètres contenus dans les données elles-mêmes, mais aussi en fonction de nouvelle information fournie par l'utilisateur. Les données doivent donc pouvoir interagir avec l'utilisateur. C'est un premier facteur qui contribue à l'émergence de la notion de " document actif ".

Voici un exemple qui illustre l'intérêt que peut présenter le document actif dans le contexte de collaboration à distance. Supposons que deux collaborateurs s'échangent régulièrement des données pour commentaires et que la plupart des échanges suivent le scénario suivant: A envoie à B un texte, B en extrait quelques paragraphes, qu'il modifie et renvoie à A accompagnés de quelques commentaires. Plutôt que de devoir basculer constamment d'une application à une autre, ou encore de faire développer une toute nouvelle application *ad hoc*, il serait beaucoup plus simple que A et B puissent créer des documents comportant, par exemple, certains boutons permettant de copier automatiquement certains paragraphes dans un message électronique en ébauche, de passer directement d'un paragraphe du texte à la section du message électronique qui le concerne et, finalement, d'envoyer le message une fois terminé.

Une illustration de l'avènement du paradigme du document actif est l'utilisation de plus en plus répandue du concept de " formulaire " dans le développement d'applications selon l'approche par composantes (components). L'utilisation de ce terme, à connotation très documentaire, illustre bien, selon nous, le rapprochement des concepts de document et de programme.

3.2 L'interface-utilisateur comme outil de contrôle et d'intégration

L'arrivée de systèmes de fenêtrage comme X-Windows et Windows nous ont fait réaliser que l'interface-utilisateur peut devenir plus qu'un simple moyen de communication entre une application et un utilisateur. Elle est en fait en train de devenir un outil de contrôle et d'intégration des applications *par l'utilisateur*. Au départ présentés comme des interfaces-utilisateur graphiques (GUIs), les systèmes de fenêtrage se sont vite révélés des outils aux

possibilités beaucoup plus grandes que celle de servir d'interface uniforme à de nombreuses applications.

Voici quelques exemples de la tendance des interfaces-utilisateur à devenir des outils de contrôle des applications par l'utilisateur:

- La possibilité de démarrer les applications à partir du système de fenêtrage, d'en faire exécuter plusieurs simultanément et de basculer de l'une à l'autre.

- La possibilité pour l'utilisateur de regrouper les applications dans différents groupes, selon ses propres critères.

- La possibilité de créer des macros ou des scripts au niveau même du système de fenêtrage.

- La possibilité d'associer certaines applications à certains types de documents et de les faire démarrer simplement en " activant " un document (e.g., double-clic dans Windows).

- La possiblité de programmer le déclenchement de certaines actions en fonction de l'heure de la journée ou en réponse à certains événements. Ainsi, par exemple, une application d'agenda électronique peut prendre le contrôle à une certaine heure pour rappeler un rendez-vous important dans une fenêtre qui se superpose aux fenêtres actives à ce moment. Une application de courrier électronique peut alerter l'utilisateur de la réception de nouveaux messages au moment où ils arrivent.

Voici maintenant quelques exemples de l'utilisation d'interfaces-utilisateur comme outils d'intégration des applications par l'utilisateur:

- La norme Z39.50 est un protocole d'interrogation à distance de bases de données. Cette norme a d'abord été introduite comme interface uniformisée de recherche à distance dans des bases de données de type catalogue de bibliothèque. Elle a donc résulté en la possibilité d'utiliser un même logiciel client (et donc, interface) pour interroger une foule de bases de données en mode client-serveur. Étant donnée cette possibilité, les producteurs de ces logiciels y ont naturellement ajouté la capacité de configurer l'interface pour pouvoir effectuer des recherches en parallèle sur plusieurs bases de données. L'interface permet donc d'activer ou désactiver certaines bases de données, de fusionner les résultats des recherches, bref, d'intégrer en un tout commode les différentes applications serveurs qui tournent sur les différentes bases de données séparées.

- Les fonctionnalités copier-coller que l'on retrouve dans la plupart des systèmes de fenêtrage sont aussi un exemple d'intégration de différentes applications rendue possible par l'interface elle-même. Grâce à ces fonctionnalités, l'usager peut déclencher lui-même le transfert d'information entre des applications qui n'ont pas été conçues explicitement pour travailler l'une avec l'autre.

- La possibilité pour certaines applications de se substituer au pilote d'impression du système, de façon à récupérer automatiquement des données à traiter. Par exemple, beaucoup d'applications de télécopie fonctionnent de cette façon; l'opération d'expédition d'une télécopie est alors aussi facile que l'impression et ce, à partir de n'importe quelle application capable d'imprimer.

- La possiblité de créer des documents multimédias (par exemple, des documents OLE ou OpenDoc) incluant des composantes qui requièrent chacune sa propre application de restitution. La navigation à l'intérieur du document peut être vue comme une application originale développée par l'utilisateur et intégrant les applications de restitution des différents contenus selon les " directives " consignées dans le document multimédia.

Cette tendance à faire des interfaces-utilisateur des outils de contrôle et d'intégration résulte en un besoin de langages de configuration et de scriptage puissants, mais aussi accessibles directement à l'utilisateur (averti, soit), plutôt qu'à un programmeur chevronné. Elle renforce également le besoin de documents actifs, qui peuvent déclencher différentes actions dans leur environnement et être maniés par l'utilisateur de façon très souple.

3.3 L'information comme actif organisationnel

La considération de l'information comme un actif pour la société est chose courante depuis très longtemps dans le monde de la bibliothéconomie et de l'archivistique. Le phénomène est un peu plus récent dans les grandes organisations comme les gouvernements, les grandes sociétés d'état, mais on commence nettement à l'observer, par exemple en voyant les importants travaux d'analyse et d'organisation de l'information qui sont en cours dans différents organismes gouvernementaux un peu partout dans le monde. Le phénomène est carrément nouveau dans le milieu des petites et moyennes organisations. Cependant, des facteurs comme les lois d'accès à l'information, la législation sur les archives et la prise de conscience de la valeur de l'information font en sorte qu'il prend de plus en plus d'importance.

Peu importe le milieu dans lequel il se manifeste, le phénomène de l'" information-actif " doit actuellement composer avec l'avènement de l'information sous forme électronique. Cette nouvelle donnée représente d'une part un défi additionnel, puisque les mécanismes de gestion de l'information mis en place doivent être capables de traiter l'information électronique, et d'autre part, des possibilités nouvelles, puisque plusieurs opérations automatiques sont plus faciles à réaliser avec de l'information électronique qu'avec de l'information sur supports traditionnels.

Il n'est pas faux de dire que, jusqu'à présent, l'information électronique était considérée d'abord et avant tout comme une forme transitoire d'information, que l'on pouvait simplement jeter après usage (la forme définitive, si elle était requise, était sur un support traditionnel, comme le papier ou le microfilm). Le passage au paradigme de l'information-actif implique donc un changement d'approche. Les principes et méthodes élaborés en bibliothéconomie et en archivistique, bien qu'encore valides dans le contexte de l'information électronique, ne sont pas directement applicables. On est donc actuellement en phase de réflexion et d'expérimentation concernant les méthodes pour traiter l'information électronique comme un réel " actif " dans une organisation ou dans la société.

Les effets de cette situation sont:

- Le besoin de recourir à des formats qui offrent une stabilité certaine pour l'information électronique. L'utilisation de formats normalisés est donc une nécessité. Notons d'ailleurs que plusieurs projets pilotes de gestion d'archives électroniques utilisent SGML.

- L'importance accordée à la notion d'" entrepôt " documentaire (document repository). L'entrepôt documentaire agit en quelque sorte comme le service d'archive électronique d'une organisation; c'est un endroit où sont consignés l'ensemble des documents produits (et même éventuellement, reçus) par l'organisation, sous une forme recherchable et réutilisable. Les différentes applications de l'organisation peuvent tirer parti de l'entrepôt, en venant y puiser de l'information grâce à différents mécanismes de recherche. La tendance est d'ailleurs en ce moment de retirer ces mécanismes de recherche des applications spécifiques et de les intégrer dans les systèmes d'exploitation de réseaux.

- Le besoin de stocker l'information électronique sous une forme qui favorise sa réutilisation. En effet, un des grands attraits de préserver l'information électronique comme un actif est la possibilité de la réutiliser pour produire de la nouvelle information (par exemple, après avoir effectué des analyses sur les anciennes données, ou encore après y avoir apporté certaines modifications).

4. Nouvelles architectures de documents

L'ensemble des tendances relatées à la section précédente amène un besoin de documents actifs, paramétrables autant par des informaticiens que par les utilisateurs, partageables, réutilisables, et qui soient représentés dans des formats normalisés et durables. Ces multiples aspects font qu'il y aura certainement plusieurs modes d'interaction avec les " nouveaux documents ". On peut déjà prévoir au moins les niveaux suivants: programmation, scriptage (par des architectes de documents ou des utilisateurs avertis) et création de contenu. Toute architecture de " nouveaux documents " devra offrir ces différents modes d'interaction.

Toutes les architectures documentaires existantes ne s'adressent évidemment pas à tous ces besoins à la fois. Plusieurs sont issues de propositions spécifiques de l'industrie pour répondre à un besoin unique. Nous essaierons dans cette section de les regrouper par grande famille et d'identifier à quels besoins chaque famille s'adresse. Bien sûr, certaines architectures sont situées à la frontière entre deux familles.

4.1 Documents du WWW

Une première famille est celle des documents du WWW. Mentionnons notamment HTML, qui confère aux documents des capacités de liens hypertextuels et les formulaires (déjà une certaine forme d'" activité "), de même que son successeur, HTML 3, et les extensions de Netscape. L'introduction d'un véritable langage de programmation comme Java (de Sun), et de langages de scriptage comme JavaScript, le VRML (Virtual Reality Modeling Langage[3]) et le SMSL (Standard Multimedia Scripting Language; une proposition de norme présentement à l'étude par ISO/IEC) constitue une percée franche dans la direction du document actif.

[3] Ce langage est aussi appelé " Virtual Reality *Markup* Language " (e.g., Byte, mars 1996). Cette confusion de nomenclature reflète bien l'apparition du nouveau paradigme de document actif. On est vraiment à la limite entre un langage de description de documents (Markup) et un langage de type programmation (Modeling).

Cette première famille vise bien sûr les besoins des documents actifs. Par la nature multi-plateforme de l'Internet, l'aspect d'échange des documents est également assuré. Le principal manque est actuellement du côté normalisation, surtout au niveau des langages de programmation, mais également au niveau du scriptage et du contenu (balisage) des documents. Java est un langage sous license, VRML et HTML existent en une myriade de saveurs et SMSL n'est qu'une proposition de norme encore loin du statut de norme ISO. Pour ce qui est de la réutilisation, elle est aujourd'hui assurée de façon précaire au niveau programmation par la nature orientée-objet de Java, mais pas au niveau des contenus: les balises HTML sont trop souvent utilisées de façon orientée-présentation, ce qui empêche en grande partie la réutilisation des contenus.

4.2 Documents composés

Une deuxième grande famille provient du génie logiciel, plus particulièrement de techniques mises au point pour augmenter la réutilisation du logiciel. Il s'agit des documents " composés " (compound documents). On retrouve ici principalement OLE (Object Linking and Embedding, de Microsoft) et OpenDoc (de Component Integration Laboratories), basé sur CORBA (Common Object Request Broker Architecture), de OMG (Object Management Group). Ces architectures suivent l'approche orientée-objet. Cette approche s'inscrit dans la tradition des méthodologies de développement de logiciels. Son objectif principal est la réutilisation du logiciel.

La notion de document joue un rôle central dans ces architectures. L'application type de ces architectures en est une de création de documents, avec laquelle l'utilisateur peut façonner un document en y intégrant différentes composantes, entre autres multimédias. Cette forme de document est donc compatible avec l'idée d'interface comme outil de contrôle et d'intégration. Grâce au souci d'orientation-objet, la réutilisation, du point de vue programmation, est assurée. Ce qui est moins sûr, c'est la réutilisation des contenus: ces architectures admettent n'importe quelle représentation de l'information et les développeurs devront donc de façon très délibérée adopter une structuration logique de l'information. Il semble aussi y avoir une faiblesse du point de vue langage de scriptage: les applications OLE et OpenDoc semblent devoir être développées entièrement par des informaticiens. CORBA 2.0 et Network OLE amélioreront les possiblités d'échange géographique des documents grâce à la notion d'objets distribués.

4.3 Formats propriétaires

Une troisième famille est issue de l'évolution de formats propriétaires. Certains essaient de s'imposer comme normes *de facto* (surtout sur le WWW) ou deviennent plus sophistiqués, en offrant de plus en plus des fonctionnalités de documents actifs. Par exemple, Word avec son langage WordBasic, le format Windows Help avec les fonctionnalités hypertextuelles, PDF (Portable Document Format, d'Adobe), inspiré du PostScript, auquel on a ajouté des fonctionnalités hypertextuelles et navigationnelles et que l'on a rendu plus facile à transporter sur réseaux en y intégrant la compression du texte. Le problème majeur avec cette famille est l'absence quasi-totale de réutilisation, principalement parce que l'information est prisonnière de formats propriétaires, mais aussi parce que la plupart de ces formats sont orientés présentation.

4.4 Formats normalisés de documents structurés

Une quatrième famille est celle des formats normalisés de documents structurés, c'est-à-dire capables de représenter la structure logique de l'information contenue dans les documents. Les formats normalisés de documents structurés reconnus par ISO sont SGML et ODA (Open Document Architecture; ISO 8613). ODA permet une structuration logique de l'information similaire à celle de SGML, mais inclut en plus une description physique de l'information telle que restituée (sur papier, par exemple). Bien qu'intéressant en principe, ODA a, pour diverses raisons, eu du mal à se faire des adeptes et est encore actuellement très peu utilisé.

On inclut également différentes couches sémantiques particulières qui peuvent s'ajouter à SGML. On compte ici certaines DTD normalisées spécifiques, comme la DTD ISO 12083 (anciennement connue comme la DTD de l'AAP; Association of American Publishers) ou celle de la TEI (Text Encoding Initiative, vaste entreprise d'encodage en SGML de textes littéraires). On compte également les DTD HTML 2 et 3, résultats d'un effort de normalisation du langage du WWW, mais qui sont en pratique peu utilisées. On inclut ici aussi HyTime (Hypermedia/Time-based Document Structuring Language; ISO 10744), une extension compatible de SGML pour la représentation de documents hypermédias.

Cette famille d'architectures est la seule qui se préoccupe de pérennité de l'information, élément assuré par la normalisation. SGML, lorsque utilisé pour le balisage logique, permet d'atteindre la réutilisation de l'information contenue dans les documents. Certaines DTD, comme celles de HTML ou RAINBOW (de Electronic Book Technologies), plus orientées vers la présentation, n'offrent pas les mêmes possiblités de réutilisation. Certaines autres, élaborées de façon très soigneuse et réfléchie, exploitent vraiment cette possiblité; c'est le cas notamment de la DTD TEI.

Pour l'aspect " actif " des documents, on peut maintenant compter sur DSSSL (Document Style Semantics and Specification Language; future norme ISO 10179) pour représenter de façon normalisée les traitements à effectuer sur (entre autes) les documents SGML. DSSSL est un langage descriptif basé sur Scheme (variante orientée-objet de LISP). On peut donc espérer qu'il sera un substrat adéquat pour conférer aux documents SGML la qualité de documents actifs. Ce qui semble manquer actuellement est un langage de scriptage. Pour l'échange géographique des documents (et des programmes DSSSL), l'approche la plus naturelle est de s'appuyer sur le protocole HTTP (Hypertext Transfer Protocol), sous-jacent au WWW; il pourrait cependant être avantageux d'adopter (ou de développer) un jour une architecture distribuée comme celles de Network OLE et CORBA 2.0.

Le Tableau 1 situe quelques architectures documentaires par rapport aux quatre familles présentées ci-dessus. Chaque famille est subdivisée en trois catégories, selon le niveau d'interaction prévue: création de contenus, scriptage et programmation. À l'intérieur d'une catégorie, la direction verticale vers le bas correspond à une progression du plus générique au plus spécifique, ou encore d'une représentation plus abstraite de l'information à une représentation plus concrète. Les flèches représentent des liens de parenté.

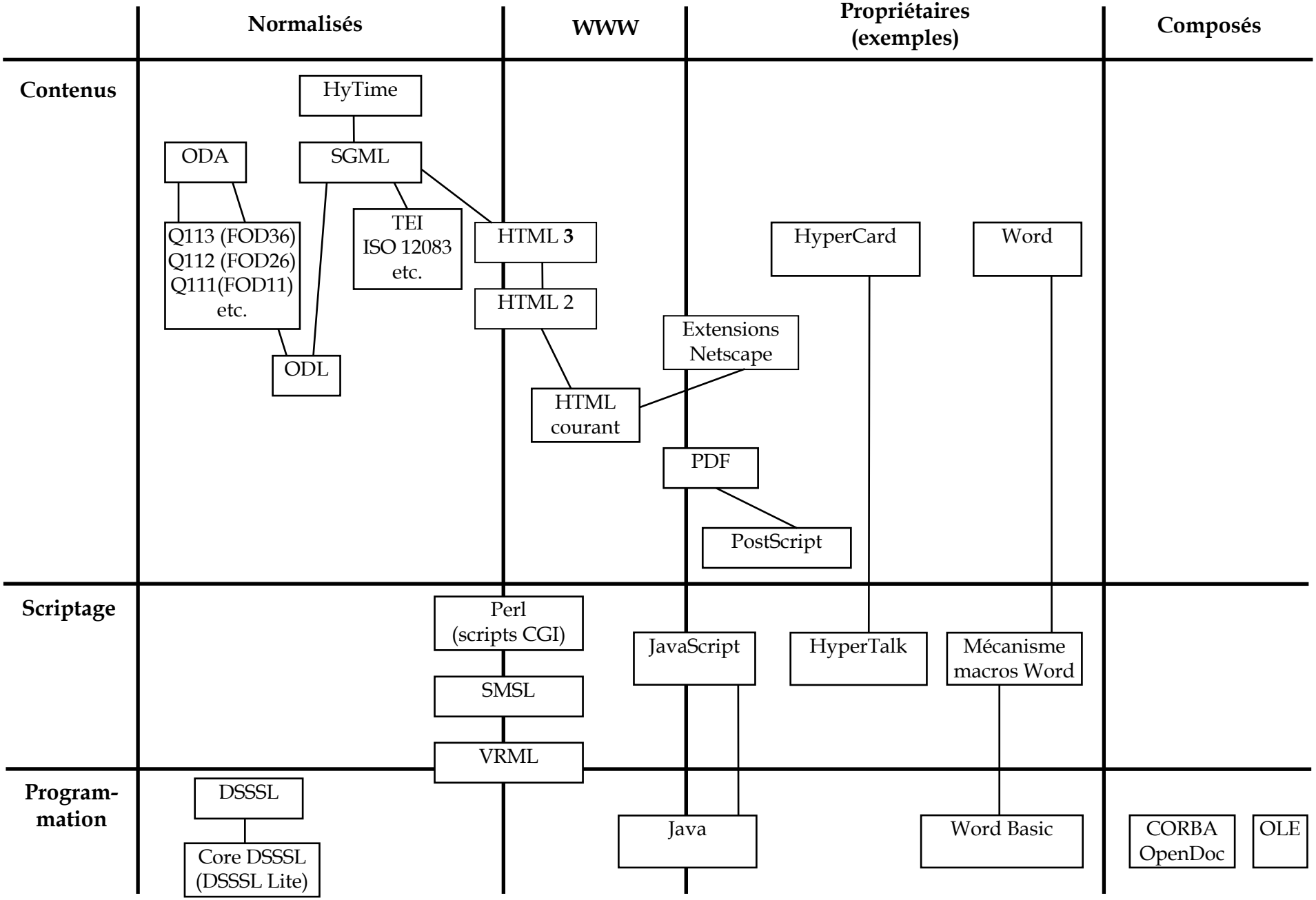

Tableau 1 - Quelques architectures documentaires

5. Rôle de SGML

Le principe de la séparation contenu-traitement en SGML est très voisin de l'approche orientée-objet; en fait, il correspond à une forme d'*encapsulation*, un des principes fondamentaux de l'orientation-objet. Il n'est pas surprenant que SGML, tout comme l'orientation-objet, permette une forme de réutilisation de l'information: dans le cas de SGML, c'est la réutilisation des contenus; pour l'orientation-objet, c'est la réutilisation des programmes. Dans la perspective du " nouveau document ", SGML et l'orientation-objet apparaissent donc comme des compléments mutuels.

SGML est un substrat syntaxique qui a prouvé déjà son utilité pour la normalisation. Ce qui le rend intéressant par rapport à d'autres normes possibles, c'est son haut degré d'utilisation dans le monde et sa flexibilité très grande, qui en fait un substrat syntaxique adéquat pour à peu près tous les types de sémantiques imaginables. Ainsi, SGML pourrait être utilisé pour des langages de scriptage et même des langages de programmation. En fait, plusieurs propositions de telles utilisations de SGML circulent régulièrement sur l'Internet.

Même s'il y a des avantages à utiliser SGML comme unificateur syntaxique (e.g., la même série d'outils peut être utilisée), il faut faire attention: D'abord, l'utilisation de la syntaxe SGML n'est pas *essentielle*; après tout, DSSSL a été développé d'abord pour traiter des documents SGML et c'est une syntaxe " classique " qui a été retenue. Ensuite, les caractéristiques les plus importantes pour un langage de traitement des documents sont qu'il permette vraiment de rendre les documents " actifs " et aussi qu'il permette la réutilisation du " code " (programmes ou scripts). L'orientation-objet semble donc très appropriée. Ceci dit, notre impression est que la syntaxe SGML pourrait être particulièrement intéressante pour les langages de scriptage.

SGML peut aussi jouer un autre rôle dans l'évolution des architectures documentaires. Alors que plusieurs architectures documentaires se généralisent pour essayer d'englober plusieurs caractéristiques du nouveau document, cette généralisation devrait se faire dans le respect des principes de réutilisation des contenus et de normalisation. SGML doit ici servir de phare: encourager le balisage descriptif, la représentation de la structure logique de l'information et la normalisation. Ce sont des éléments essentiels à la pleine exploitation de la valeur de l'information.

6. Conclusion

Dans cet article, nous avons rappelé les principes de base de SGML; puis, nous avons analysé les forces en action sur la notion de document et présenté les principales architectures qui occupent aujourd'hui le terrain de l'informatique documentaire. Finalement, nous avons discuté du rôle que peut jouer SGML dans l'évolution des nouvelles architectures documentaires.

Il n'existe pas à notre connaissance d'architecture de documents unique qui réponde pleinement à toutes les exigences du " nouveau document ", ni existante, ni même en développement. Cependant, quelques approches adoptées par des groupes de recherche nous semblent dignes

de mention, car elles pointent dans la bonne direction. Incidemment, toutes ces approches sont basées sur SGML.

Le projet Metamedia, dirigé par Michael D. McCool, de l'Université de Waterloo, vise, via les concepts d'objets distribués et de langages multi-plateformes, à faire des applications distribuées de véritables outils d'apprentissage et de coopération.[4] Matthew Fuchs, de la West Virginia University, considère pour sa part une interface-utilisateur simplement comme un document, dont l'interprétation est contrôlée par un interprète, lui-même spécifié dans un document SGML.[5] Patricia François, de l'Aérospatiale, en France, propose un modèle d'entrepôt de documents SGML et HyTime.[6]

Devant ces exemples d'efforts de recherche et de développement, nous croyons réellement que des architectures normalisées de documents actifs, réutilisables et échangables verront éventuellement le jour. Alors, la société sera un peu plus à même de toucher les bénéfices qu'elle est en droit d'attendre de l'informatique. Nous sommes convaincus que SGML a un rôle important à jouer dans l'élaboration de ces solutions.

[4] Site WWW du projet Metamedia: `http://www.meta.cgl.uwaterloo.ca/`

[5] FUCHS, Matthew. "The user interface as document: SGML and distributed applications." *Computer Standards and Interfaces*, Vol. 18, 1996, pp. 79-92.

[6] FRANÇOIS, Patricia. "Generalized SGML repositories: Requirements and modelling." *Computer Standards and Interfaces*, Vol. 18, 1996, pp. 11-24.