Problèmes d'intelligibilité et solutions autour de XML

Jérôme Euzenat

Problèmes d'intelligibilité et solutions autour de XML

Jérôme Euzenat

INRIA Rhône-Alpes
655 avenue de l'Europe, 38330 Montbonnot, France
Jerome.Euzenat@inrialpes.fr

RÉSUMÉ DE L'INTERVENTION. Les problèmes d'intelligibilité et d'interopérabilité que pose et que résout le langage XML sont examinés en explorant progressivement les travaux destinés à les résoudre: XML en tant que langage universel, permet théoriquement l'interopérabilité. Mais XML, métalangage sans sémantique, n'offre aucune possibilité d'intelligibilité pour qui (humain ou programme) ne connaît pas le contenu. XML-Schéma n'améliore que l'interopérabilité en définissant très précisément les types de données (et parfois leurs unités). RDF, langage de description de ressources, est destiné à « ajouter de la sémantique » mais n'en dispose pas lui-même. Il sera donc très difficile (lire impossible) pour un programme de l'interpréter. Plusieurs initiatives indépendantes du W3C s'attachent à produire des langages de descriptions de contenu cette fois-ci dotés d'une sémantique rigoureuse. Ce faisant, ces langages réduisent drastiquement leurs champs d'utilisation et par conséquent les possibilités d'interopérabilité des documents les utilisant. Si le temps est suffisant, on pourra présenter brièvement (a) une proposition de langage de description de la sémantique destiné à préserver l'interopérabilité en améliorant l'intelligibilité ainsi que (b) un projet, actuellement en cours, de comparaison de plusieurs formalismes de représentation de connaissance pour la représentation du contenu.

MOTS-CLÉS : XML, RDF, Schéma, DSD, Représentation de connaissance par objets, Sémantique, Interopérabilité, Intelligibilité.

XML ("eXtensible Markup Language" [Bray& 1998a]) de par son positionnement comme métalangage d'échange est destiné à permettre l'interopérabilité entre systèmes. S'agissant de la représentation et de la conservation des données et de leur signification, on est amené à considérer son intérêt. La suite évoque les avantages et les lacunes de XML dans ce domaine et dresse un aperçu des solutions en cours de développement et envisageables pour y remédier.

Dans ce document, le mot interopérabilité décrit la faculté pour deux systèmes informatiques de s'échanger leurs données (et de pouvoir se les approprier, par exemple les restituer dans le format initial). Le mot intelligibilité décrit la faculté pour deux systèmes de « comprendre » les données qu'ils échangent. Pour des systèmes informatique, cela consistera à agir en accord avec la sémantique des données alors que pour des êtres humains cela consistera à en reconstruire le sens (quoi que cela puisse signifier).

La première section est une très rapide introduction à XML. La seconde section considère les langages de description de Schéma à commencer par les descriptions de types de documents (DTD) issus de SGML et leur rôle limité dans l'interopérabilité et la documentation des données. Pour aller plus loin on évalue l'intérêt du mécanisme d'annotation RDF ("Resource Description Framework") en tant que documentation des données exprimées en XML (§3). Mais la principale déficience, à nos yeux, des langages cités plus haut est leur absence de sémantique

formelle qui n'est pas à même de permettre une interprétation non ambiguë des données. On se penche alors sur des langages de représentation de connaissance disposant d'une sémantique et encodés en XML (§4). Mais la force de ces langages est aussi leur limite : leur précision les contraint à une expressivité étriquée qui ne peut les placer en position de langages de représentation et d'échange universels. Afin de remédier à cela on propose le développement de description de sémantique de documents (DSD) permettant d'exprimer la sémantique d'une DTD particulière (§5).

Ce résumé est un rapide survol des options présentées, on trouvera sur notre site (`http://www.inrialpes.fr/exmo`) des présentations plus précises.

1. PRÉSENTATION DE XML

XML [Bray& 1998a] est un langage de balisage de document recommandé par le "World-wide web consortium" (W3C). Il a pour vocation d'être un format d'échange de documents entre diverses applications. Il n'est pas un format de stockage — mais risque de le devenir —, ni un format de présentation. Un rapide point de départ est [Bosak& 1999] et une description plus complète en français se trouve dans [Michard 1998].

XML est délibérément intermédiaire entre HTML et SGML. est plus simple et moins contraignant que SGML et plus complexe et plus contraignant qu'HTML (dont l'interprétation est très tolérante puisqu'elle ignore ce qu'elle ne comprend pas). Contrairement à HTML et à l'instar de SGML, XML dispose d'un langage permettant de décrire des formats (DTD pour description de type de documents). HTML peut être, à quelques détails près, redéfini comme une DTD XML.

Les documents XML suivent une syntaxe relativement simple qui peut se résumer par :

```
<TAG att1=v1…attn=vn/>
```

ou

```
<TAG att1=v1…attn=vn>contenu</TAG>
```

où `contenu` est une suite d'expressions XML ou une chaîne de caractères, `atti` est un identificateur d'attribut et `vi` la valeur de cet attribut. Le `TAG` est appelé élément. Un document XML est vu comme un arbre dont la racine est le document et les fils d'un nœud sont les éléments de son contenu.

L'exemple ci-dessous représente une protéine (nommée « BICOID »), qui a pour longueur 422 acides aminés, qui code pour le gène « Bicoid » et qui interagit avec le gène « Hunchback ». Le voici sous la forme d'un document XML :

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<!DOCTYPE PROTEIN SYSTEM "protein.dtd">

<PROTEIN name="BICOID" length="422">
  <GENE name="Bicoid"/>
  <INTERACTION>
    <PROTEIN name="BICOID"/>
    <GENE name="Hunchback"/>
```

```
  </INTERACTION>
</PROTEIN>
```

L'intérêt de considérer XML ici est qu'un format d'échange aussi simple et, on le verra à la section suivante, aussi malléable devrait être à même d'assurer l'interopérabilité des applications informatiques par la communication des données. Ce format est appréhensible sans avoir une définition préalable des données disponibles. Il encourage donc ouvertement la réappropriation de données disponibles.

D'autre part, XML, porté par la vague du Web, devrait être largement adopté d'autant qu'il s'agit d'une proposition ouverte, de qualité et pour laquelle sont d'ores et déjà disponibles de nombreux logiciels. L'"Object Management Group" ne s'y est pas trompé qui a choisi XMI ("XML Metadata Interchange" [XMI 1999]) comme langage de sérialisation de son langage de spécification UML.

2. LES SCHÉMAS

Les schémas permettent de décrire l'organisation des données au sein d'un document XML. Ils précisent les éléments autorisés, leurs attributs et leur contenu. Ces derniers peuvent être décrit de manière plus ou moins précise suivant le langage. Même s'il n'est pas obligatoire de référencer le type dans les documents XML et même si ces types sont de nature limitée, connaître le type des documents est le début de l'intelligibilité. C'est pourquoi on évoque ici les schémas, permettant de décrire les types de documents et leur intérêt. On présentera aussi les limitations actuelles de ces schémas et leur limitation immédiate : ils ne représentent pas la sémantique des documents mais leur syntaxe.

On présente d'abord les DTD, partie intégrante de la recommandation XML (§2.1), les premières extensions proposées hors du W3C (§2.2) et les travaux du groupe de travail Schéma du W3C (§2.3).

2.1 Description de type de documents

Les documents XML sont destinés à être manipulés indépendamment de leur DTD, c'est pourquoi il est justifié de présenter le document sans sa DTD. Une DTD a pour but de définir chaque élément en précisant :

— son contenu comme une expression régulière introduisant la séquence et l'alternative fonction d'un nombre variable (un, zéro ou un, plus d'un, nombre quelconque) d'autres éléments ;

— ses attributs (optionnel) en précisant le type de valeur prise et un ensemble d'autres paramètres (requis ou optionnels, valeurs par défaut…).

En XML un document est dit :

— Bien Formé (“well-formed”) s’il correspond à la syntaxe XML (c’est-à-dire si l’imbrication des chevrons et guillemets forme une expression bien parenthèsée et les éléments ouverts sont bien fermés sans entrelacement).
— Valide (“valid”) s’il satisfait les contraintes exprimées dans sa DTD (ou DOCTYPE). Ci-dessous un tel document sera qualifié d’XML-valide.
— lide (“Schema-valid”, non normatif) s’il satisfait les contraintes exprimées dans son Schéma (voir §3.2).

XML est assez peu précis en ce qui concerne la description des types de documents. Nombreux sont ceux qui voudraient y ajouter les caractéristiques suivantes :

— La relation de spécialisation (base de l’héritage) qui est absente (tout comme l’héritage) des définitions d’éléments.
— Les types de données qui sont restreints (chaînes et énumération). Le typage est à renforcer avec des types plus diversifiés (externes) et la possibilité de raffiner les valeurs possibles (un entier positif non nul par exemple).
— Les collecteurs : s’il est possible de spécifier qu’un contenu peut être composé d’un (ou plus) élément(s), il n’est ni possible de préciser sa cardinalité, ni de spécifier qu’il s’agit d’un ensemble ou d’une liste.
— La notion de référence entre objets (et non entre parties de documents comme le fait ID/IDREF).

Toutes ces caractéristiques se trouvent dans les langages de représentation de connaissance. Quelques propositions dont le but est d’inclure ces constructeurs dans XML sont présentées ci-dessous. Elles permettent ainsi de définir une structure plus précise et d’en vérifier le typage.

2.2 Premières extensions

Il existe plusieurs initiatives, liées au W3C, contribuant à donner un aspect objet à XML. SOX (“Schema for Object-Oriented XML”) est un schéma minimal et bien pensé pour XML. Il reprend les principaux traits des DTD (exprimées en XML) et organise les types d’éléments en hiérarchie. Il semble que cette proposition ait été utilisée dans plusieurs systèmes si bien que ses auteurs la tiennent à jour par rapport à XML-schéma. DCD (“Document content description” [Bray& 1998b]) est une proposition qui vise à étendre la définition des DTD avec la spécialisation, un contenu structuré sous la forme de divers types de groupes (ensembles, multi-ensembles, listes…), des contraintes de cardinalités sur ce contenu, des attributs structurés (types, contraintes d’intervalles, valeurs par défaut), l’introduction de types externes et les valeurs nulles. Cette proposition est assez bien pensée et extensible. Ces deux propositions servent de base aux travaux du groupe XML-Schéma.

2.3 XML-Schéma

XML-Schéma est un groupe de travail du W3C qui s'occupe des extensions de XML concernant les types de données. Les résultats de ses travaux ont été développés dans deux propositions préliminaires :

- Schema-Structure [Thomson& 1999] donne la définition d'un schéma permettant d'associer aux éléments des DTD des structures plus complexes dont le type des attributs est plus précis;
- Schema-Data [Biron& 1999] permet de définir ses propres types de données à inclure dans les schémas.

Plusieurs avantages sont espérés de ces extensions dont la simplification des DTD grâce à l'utilisation d'attributs XML typés et plus généralement une plus grande expressivité des DTD permettant aux analyseurs XML un contrôle de type plus précis sur des types plus étendus que ceux de XML. Actuellement, les schémas permettent d'exprimer trois sortes d'entités : types de données (ou types de base : chaînes, entiers, dates…), types d'éléments et éléments. Les types de base peuvent être raffinés (entiers positifs pairs non nuls) mais pas étendus (impossible d'ajouter des séquences d'amino-acides par exemple). Les types d'éléments sont organisés par une relation de sous-typage et les problèmes posés par la spécialisation de contenu ont été codifiés. Il est par ailleurs possible de poser des contraintes de cardinalité sur les séquences d'éléments. Les éléments, curieusement, ne peuvent se spécialiser les uns les autres, cette faculté étant réservée à leurs types.

Un document Schéma-valide, à la différence d'un document simplement valide, n'est plus seulement bien structuré mais il est aussi bien typé. Par contre il manque encore beaucoup à XML-Schéma pour approcher une représentation de connaissance par objets : L'héritage est ici de la réutilisation dans les types (mais il n'y a pas de spécialisation entre éléments), les collecteurs (ensemble, liste) sont toujours absents, et il y a clairement dans XML-Schéma la possibilité de raffiner les types existants en utilisant des propriétés abstraites de longueur ou d'ordre (et même la cardinalité des ensembles supports), mais il n'est pas possible d'ajouter de nouveaux types comme cela l'est dans un langage comme TROEPS dispose pas de types primitifs. Pour cela il faudrait pouvoir définir un prédicat d'appartenance ou d'ordre par exemple [Capponi 1995]. Ceci suppose de pouvoir donner une définition exécutable des types de données. Les concepteurs d'XML-Schéma ont évité cela (de manière à ne pas faire allégeance à langage particulier). C'est d'autant plus dommage que la « sérialisation » telle que promue par Java est un excellent point de départ pour l'intégration de types externes à XML.

3. DESCRIPTION DE RESSOURCES

Le typage introduit par les schémas est d'abord un élément de génie logiciel ou de génie documentaire. Il permet à la fois une expression plus complète des éléments et une factorisation

des définitions (qui contribue à une meilleure maintenabilité). Si ces éléments ne permettent pas directement une plus grande intelligibilité, au moins ils y contribuent par une meilleure lisibilité des spécifications.

Connaître le type n'est pas connaître le contenu ou le contexte. Afin de fournir ceux-ci, de gros efforts ont porté sur les méta-données. Le Dublin-core ("Dublin core metadata initiative") définit un ensemble de 15 éléments (titre, auteur, date…) pour annoter les documents et en particulier les documents HTML. Mais ces annotations, si elle sont très utiles, restent intrinsèquement liées à la documentation.

De son côté, le W3C développe RDF ("Resource description framework" [Lassila& 1999, Champin 2000]) en tant que cadre général d'annotation des documents XML. RDF permet de décrire les relations binaires entre ressources (qui peuvent être des documents ou parties de documents ou de chaînes de caractères). Les ressources sont identifiées par des URI et les propriétés par un nom. Le fait qu'une ressource puisse être la valeur d'une relation fait que les annotations RDF constituent un immense graphe à l'échelle du monde entier (grâce aux URI). RDF possède en plus la faculté de pouvoir grouper ses valeurs dans des ensembles, listes, multi-ensembles (ce qui aurait dû être spécifié par XML-Schéma). Par ailleurs, les traits de spécialisation et de typage décrits dans les schémas RDF [Brickley& 1999] sont plus simples que ceux permis par les schémas XML mais permettent à un « analyseur RDF » de produire un certaines inférences.

Cependant, la sémantique des méta-données, si elles sont claires bien qu'informelles dans le Dublin core, ne le sont pas dans RDF. Ainsi, si on considère une URL comme identifiant une ressource (et référençant une page HTML), cette ressource est-elle le document référencé, ce que représente le document (son contenu) ou quelque chose qu'il représente en lui-même ? Ces questions typiquement du ressort de la sémantique d'un formalisme n'ont pas été éclaircies dans RDF. Ce premier travail reste à faire après quoi il sera possible de définir des jeux de méta-données adaptés à une tâche particulière telle le Dublin core.

4. UN XML SÉMANTIQUEMENT DÉFINI

Les schémas, les annotations et les méta-données ne fournissent pas une sémantique. Pour cela, des langages de représentation de connaissance, dotés de sémantique formelle, ont été développés depuis des années. De nombreux efforts portent donc vers l'encodage de ces langages dans XML (représentations de connaissance par objets [Heflin& 1998, Euzenat 2000], logiques de description [Bechofer& 1999], graphes conceptuels [Kent 1999], logiques de traits [Erdman& 1999]). Ces DTD, plus spécifiques et hors du W3C, permettent donc d'exprimer des notions relativement générales dotées d'une sémantique formelle.

Il est commun d'entendre la réflexion «XML, c'est des objets». À l'exception de la différentiation entre contenu et attributs, XML fait naturellement penser à un langage objet. En tant que métalangage sans sémantique, XML peut être qualifié de déclaratif : il décrit une structure qui est interprétée de l'extérieur. Il est donc possible de chercher à utiliser XML comme un langage de représentation de connaissance par objets.

La vision selon laquelle un document XML est une base d'objets peut être plaquée directement sur une situation où la DTD correspond à la description de classes et les documents XML sont des instances. Cette solution a pour principal défaut de ne pouvoir accepter de modifications dynamiques des classes ce qui est couramment fait en représentation de connaissance. Par contre, il faut remarquer que toute la vérification de type est déléguée à l'analyseur XML : un document XML-valide est aussi RC-valide (un document XML sera dit RC-valide, s'il est valide et si son contenu est considéré comme consistant par le système de représentation de connaissance, en particulier, il sera bien typé).

Dans une seconde solution, les documents contiennent classes et instances et les attributs font partie du contenu des éléments. Cette seconde approche peut être qualifiée de réification de l'« ontologie ». Ici, le schéma de la base se trouve décrit dans le document XML et non dans la DTD (qui est commune à l'ensemble des bases possibles). Ceci conduit à des documents qui seront plus volumineux. Cependant, l'expérience montre que la taille d'un schéma est négligeable par rapport à la taille des données.

Il n'y a pas de typage (au niveau d'XML) dans cette proposition. Mais les éléments de typage présents dans la représentation de connaissance sont aussi présents dans le document si bien qu'un post-processeur connaissant leur sémantique pourra vérifier la RC-validité du document.

La seconde solution permet de récupérer les objets (dans un système de représentation de connaissance). Ce n'est pas le cas de la première car les balises utilisées lui seront totalement inconnues et dépendantes de la base considérée. En effet, il n'y a pas dans la première solution de classes et par conséquent d'instances de ces classes : cela correspond à une base de connaissance compilée. Par contre, la seconde solution implique qu'il ne sera pas possible de déléguer l'édition, ou même la vérification de types, à un simple éditeur XML validant.

Le clivage entre les deux solutions rejoint le clivage général entre bases de données et bases de connaissance : dans les premières les données sont massivement manipulées alors que dans les secondes c'est plutôt le schéma conceptuel qui est l'objet de manipulations.

La première solution a été retenue par DTDMaker [Erdman& 1999], qui transforme une « ontologie » (en F-Logic) en une DTD, et XMI [XMI 1998], qui transforme une description en "Meta Object Facility" (dont le méta-modèle UML) en une DTD. Beaucoup d'autres tentatives adoptent plutôt la seconde solution.

SHOE ("Simple HTML Ontology Extensions" [Heflin& 1998]) bien que lié à HTML (comme son nom l'indique) est immédiatement transposable en XML. Il permet de définir des hiérarchies

de classes munies d'attributs typés et des règles de déduction de type clauses de Horn. Son but est d'intégrer aux pages HTML une représentation formelle du contenu accessible à des agents. L'expressivité du langage est relativement réduite et la sémantique reste informelle.

OML ("Ontology Markup Language" [Kent 1999]) a pour but d'introduire les graphes conceptuels en XML. En fait, c'est aussi un encodage du calcul des prédicats du premier ordre. Il manipule les notions de classes, d'objets, de relations et de fonctions, mais aussi de quantificateurs et de connecteurs booléens. Son expressivité est donc très importante. CKML ("Conceptual Knowledge Markup Language" [Kent 1999]) est un sur-ensemble d'OML permettant la représentation des contextes, des treillis de concepts et de séquents. C'est une option maximaliste dont le but principal est d'échanger de l'information entre applications (en communiquant le contexte), de faire de la construction de taxonomies à base de treillis de Galois et de la recherche d'information. Il existe évidement d'autres initiatives telles que la DTD de TROEPS [Euzenat 2000] ou celle de FaCT [Bechofer& 1999].

5. DESCRIPTION DE LA SÉMANTIQUE DES DOCUMENTS

Il existe donc différentes initiatives pour utiliser XML de manière à préciser la sémantique des éléments manipulés. Mais, leur principal problème est de focaliser d'emblée sur un type de langage et de réduire l'ouverture inhérente à XML. En contrepartie, ils peuvent définir de manière très précise la sémantique des éléments.

Ces efforts contribuent à l'expression dans les DTD du maximum d'information et par conséquent à la délégation à l'analyseur XML de plus en plus de tâches. Il en résulte que, progressivement, les tâches d'un système de représentation de connaissance y sont intégrées. Cette évolution est présentée ainsi :

	XML	XML-Schéma	XML-RC
XML	Échange Analyse syntaxique	Échange Analyse syntaxique	Échange Analyse syntaxique
RC	Typage Inférence	Typage Inférence	Typage Inférence

Tableau 1 : Répartition des tâches entre XML et représentation de connaissance dans les extensions présentées ci-dessus.

Ainsi, sur les aspects syntaxiques, l'évolution de XML rapproche d'un langage de représentation de connaissance. Mais la dernière colonne du tableau ci-dessus n'offre plus réellement un métalangage permettant d'échanger entre systèmes différents, elle exige de connaître la sémantique des objets manipulés.

Le résultat obtenu par la colonne du milieu, même s'il peut être critiqué n'est pas à négliger. Il permet effectivement de définir des DTD qui comprennent un sous-ensemble important de la

syntaxe des langages à objets. Ce n'est pas le cas de la sémantique. C'est principalement pourquoi XML n'est pas un langage de représentation de connaissance.

Mais pouvoir donner une sémantique à un document XML est d'autant plus important que l'avènement d'un « web sémantique » [Berners-Lee 1998] est annoncé (et que chacune des extensions précitées promettent de mettre « plus de sémantique » dans XML).

Il existe donc une tension entre XML et représentation de connaissance : celle de l'ouverture face à la transparence sémantique. Cette tension semble irréconciliable. Cependant, il est imaginable de donner aux DTD XML une sémantique par le biais d'un langage de définition de celle-ci. En se restreignant à la théorie des modèles dans un domaine ensembliste, il semble possible de produire une description du type de ce qui suit :

$$I(\texttt{CLASS}) \subseteq I(D) \cap I(\texttt{@specialises}) \cap \bigcap_{f \in \texttt{FIELD}} \left\{ i \in D; I(f_{\texttt{@name}})(i) \in I(f.\texttt{DESCRIPTOR}) \right\}$$

Cette description signifie que chaque classe est interprétée comme un sous-ensemble de l'intersection du domaine D avec l'interprétation de sa super-classe et de l'ensemble des individus pour lesquels l'application de l'interprétation de chacun des attributs retourne un élément satisfaisant l'interprétation des descripteurs de l'attribut dans la classe.

De même qu'une DTD est une définition syntaxique et que la sémantique est souvent définie de manière compositionnelle, il ne semble pas y avoir de problème particulier au développement d'une telle description. Il est possible d'avoir plusieurs DSD pour la même DTD (et donc plusieurs manières d'interpréter le même langage). L'assemblage de spécifications décrites dans différents langages ne partageant ni leur syntaxe, ni leur sémantique a déjà été exploré dans le formalismes des institutions [Goguen& 1992].

Nous avons défini un langage de Description de Sémantique de Documents (en XML) qui permet d'exprimer ceci. Ce langage est principalement fondé sur le langage MathML permettant d'exprimer les mathématiques et un appareil d'expression de la sémantique très proche du langage de transformation de XML : XSLT. Il est actuellement appliqué à la représentation par objets TROEPS, aux logiques de descriptions et à la syllogistique. Son utilisation se restreint pour l'instant à la validation manuelle de transformation entre langages mais est destinée à s'étendre.

L'échange des données, non seulement avec leur format, mais encore avec leur sémantique permettrait, pour l'ordinateur, de manipuler les données suivant la sémantique qui leur a été donnée par leur auteur. Par exemple, des transformations de langage à langage peuvent être appliquées en prenant en compte la sémantique (ne serait-ce qu'en vérifiant les transformations définies manuellement). Il est concevable d'importer dans une représentation par objets des définitions de concepts de logiques terminologiques sachant qu'elles ne seront interprétées que comme des descriptions et inversement, d'importer dans un système terminologique des descriptions d'objets sachant qu'elles doivent être interprétées comme primitives. Nous avons développé de telles traductions entre langage des syllogismes et logiques de descriptions.

L'intérêt du dispositif présenté est que l'exportation se fait dans un format qui préserve la structure des données (et non seulement la sémantique). Ainsi, les expressions exportées pourront être réimportées sans perte d'information. Parallèlement, l'interopérabilité est facilitée par l'exportation de la sémantique qui permet de mettre en correspondance deux langages. Bien entendu, une telle interopérabilité est au prix du développement d'un langage de description de sémantique complexe (dont la manipulation de manière autonome nécessiterait pour le moins un démonstrateur en théorie des ensembles).

6. CONCLUSION

Le compromis entre expressivité et complexité est particulièrement développé dans le domaine de la représentation de connaissance. Il semble que la solution soit dans la modularité des formalismes permettant de supporter uniquement la complexité nécessaire aux expressions manipulées. XML se veut un métalangage d'échange. À ce titre, il est d'autant mieux armé pour jouer ce rôle entre les différents formalismes de représentation de connaissance que la modularité (et la possibilité d'importer) et l'adressage (évitant les conflits de noms) des DTD ont été bien pensés.

Il est donc important que XML reste un langage extensible dynamiquement de manière à ne pas le brider (en empêchant son extensibilité) ni le plomber (en imposant des contraintes inutiles aux implémentations). Mais XML, comparé aux représentations de connaissance, possède les défauts de ses qualités. En particulier, le typage, l'agrégation et la spécialisation en sont absents. De nombreuses initiatives visent à combler ces lacunes. Ainsi, sur l'aspect syntaxique, seul l'accès à des types de données externes n'est pas complètement couvert afin de préserver la portabilité du langage.

Il n'en va pas de même de l'aspect sémantique : l'ouverture de XML lui interdit de définir une sémantique intrinsèque aux structures manipulées. Les initiatives pour associer une sémantique précise au langage se cantonnent à un langage particulier et nuisent à l'interopérabilité (car il y a peu de chances que cette sémantique devienne standard à l'échelle du World-wide web). Cela a conduit à ébaucher une solution permettant de définir non seulement la syntaxe des documents XML mais aussi leur sémantique de la même manière que cela est fait en représentation de connaissance.

Cependant, même si la sémantique était préservée avec les données, nous ne serions pas au bout de nos peines. La sémantique garanti en effet l'appréhension correcte des données par les machines. Elle ne permet pas, par contre, l'intelligibilité des données par des êtres humains. Par exemple, le commentaire « passable » retourné à un auteur sur un article soumis, pas plus que la note 3/7, n'a sans doute le sens du choix du rapporteur parmi « 1. Excellent, 2. Good, 3. Acceptable, 4. Average, 5. Bad, 6. Very bad, 7. Dreadful ». Pourtant, l'ordinateur qui effectue la transformation ne perd aucune information : il est capable de retourner à la situation initiale.

Pour être intelligible, le signe potentiel contenu dans les données doit être convoyé avec la manière de l'interpréter (ou son contexte d'énonciation). Si une sémiologie calculatoire était capable de nous la donner, nous aurions donc besoin d'un langage de description de l'interprétation (et des interprétations de ce langage).

7. REMERCIEMENTS

Ce travail a été partiellement réalisé dans le cadre du programme GENIE II soutenu par le MENRT et la DGA/SPAé et a bénéficié du soutien de l'action de recherche coopérative ESCRIRE de l'INRIA/France Telecom.

8. RÉFÉRENCES

[Bechofer& 1999] Sean Bechhofer, Ian Horrocks, Peter Patel-Schneider, Sergio Tessaris, A proposal for a description logic interface, Actes int. workshop on description logics, Linköping (SE), CEUR-WS-22, 1999. http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-22/bechhofer.ps

[Berners-Lee 1998] Tim Berners-Lee, Semantic web roadmap, 1998. http://www.w3.org/DesignIssues/Semantic.html

[Biron& 1999] Paul Biron, Ashok Malhotra (éds.), XML Schema part 2: datatypes, Working draft, W3C, novembre 1999 http://www.w3.org/TR/XMLschema-2/

[Bosak& 1999] Jon Bosak, Tim Bray, Le XML, Pour la science 261, 1999. http://www.pourlascience.com/numeros/pls-261/art-4.htm

[Bray& 1998a] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen (éds.), Extensible Markup Language (XML) 1.0, Recommendation, W3C, février 1998 http://www.w3.org/TR/REC-XML

[Bray& 1998b] Tim Bray, Charles Frankston, Ashok Malhotra (éds.), Document Content Description for XML, Submission, W3C, juillet 1998 http://www.w3.org/TR/NOTE-dcd

[Brickley& 1999] Dan Brickley, R. Guha (éds.), Resource description framework schema specification, Proposed recommandation, W3C, mars 1999 http://www.w3.org/TR/PR-rdf-schema

[Capponi 1995] Cécile Capponi, Identification et exploitation des types dans un modèle de représentation des connaissances par objets, Thèse d'informatique, université Joseph-Fourier, Grenoble (FR), 1995

[Champin 2000] Pierre-Antoine Champin, RDF tutorial, 2000 http://www710.univ-lyon1.fr/~champin/rdf-tutorial/

[Davidson& 1999] Andrew Davidson, Matthew Fuchs, Mette Hedin, Mudita Jain, Jari Koistinen, Chris Lloyd, Murray Maloney, Kelly Schwarzhof, Schema for object-oriented XML, Note, W3C, juillet 1999 http://www.w3.org/1999/07/NOTE-SOX-19990730

[Erdmann& 1999] Michael Erdmann, Rudi Studer, Ontologies as conceptual models for XML documents, Actes 12th KAW, Banff (CA), 1999 http://sem.ucalgary.ca/KSI/KAW/KAW99/papers/Erdmann1/erdmann.pdf

[Euzenat 2000] Jérôme Euzenat, XML est-il le langage de représentation de connaissance de l'an 2000?, Actes 6e journées langages et modèles à objets, Mont-Saint-Hilaire (CA), pp59-74, 2000

[Genesereth& 1992] Michael Genesereth, Richard Fikes, Knowledge interchange format, version 3.0 - reference manual, Rapport de recherche Logic-92-1, Stanford university, Stanford (CA US), 1992 http://logic.stanford.edu/papers/kif.ps

[Goguen& 1992] Joseph Goguen, Rod Burstall, Institutions : abstract model theory for specification and programming, *Journal of the ACM* 39(1):95-146, 1992

[Heflin& 1998] Jeff Heflin, James Hendler, Sean Luke, Qin Zhendong SHOE: a knowledge representation language for internet applications, submitted to *Artificial intelligence*, 1998 http://www.cs.umd.edu/projects/plus/SHOE/aij-shoe.ps

[Kent 1999] Robert Kent, Conceptual Knowledge Markup Language, Actes 12th KAW, Banff (CA), 1999 http://sem.ucalgary.ca/KSI/KAW/KAW99/papers/Kent1/CKML.pdf

[Lassila& 1999] Ora Lassila, Ralph Swick (éds.), Resource Description Framework (RDF) Model and syntax specification, Recommendation, W3C, février 1999 http://www.w3.org/TR/REC-rdf-syntax

[Michard 1998] Alain Michard, XML: langage et applications, Eyrolles, Paris (FR), 1998

[Thompson& 1999] Henry Thompson, David Beech, Murray Maloney, Noah Mendelsohn, (éds.), XML Schema part 1: structures, Working draft, W3C, novembre 1999 http://www.w3.org/TR/XMLschema-1/

[XMI 1998] XMI Consortium, XML Metadata Interchange, submitted to OMG OA&DTF R Stream-based model interchage format, 1998 ftp://ftp.omg.org/pub/docs/ad/98-10-05.ps