TALN 2004, Fès, 19-21 avril 2004

Annoter les documents XML avec un outil d'analyse syntaxique

Claude Roux

Xerox Research Centre Europe
6, chemin de Maupertuis
38240 Meylan
claude.roux@xrce.xerox.com

Résumé – Abstract

Cet article présente l'intégration au sein d'un analyseur syntaxique (Xerox Incremental Parser) de règles spécifiques qui permettent de lier l'analyse grammaticale à la sémantique des balises XML spécifiques à un document donné. Ces règles sont basées sur la norme XPath qui offre une très grande finesse de description et permet de guider très précisément l'application de l'analyseur sur une famille de documents partageant une même DTD. Le résultat est alors être intégré directement comme annotation dans le document traité.

This article presents the embedding within a syntactic parser (Xerox Incremental Parser or XIP) of specific rules which are used to bind the grammatical analysis to the semantic of the XML mark up tags specific to a given document. The goal of these rules is to guide the application of a natural language processing tool through the use of XPath instructions to describe documents that share the same DTD. The result can then be embedded within the input document in order to annotate that document.

Mots Clés – Keywords

XML, analyse syntaxique, traitement automatique des langues, traitement de documents, Xpath,XIP.
XML, parsing, natural language processing, document processing, XPath,XIP

1 Introduction

Depuis quelques années déjà, la norme XML s'est généralisée dans la gestion de documents. Désormais de nombreux acteurs du monde des bases de données offrent des méthodes de stockage et d'indexation de documents basés sur cette norme. La plupart des traitements de texte offre aussi de leur côté des solutions pour retraduire les documents en XML. Ce format répond en effet à une demande de plus en plus forte du monde de l'industrie d'homogénéiser

les bases documentaires des entreprises. Cette migration vise toute à la fois la possibilité de disposer d'une base de texte partageant une même représentation, mais aussi et surtout d'annoter les documents de façon à renforcer l'efficacité des outils de recherche et d'extraction d'information. Cependant cette annotation lorsqu'elle concerne des millions de documents ne peut s'effectuer à la main. Or l'utilisation d'un format unique permet aujourd'hui d'offrir aux outils linguistiques, statistiques ou symboliques, une représentation standardisée des textes, ce qui permet de simplifier l'automatisation du traitement de ces documents. Cependant, pour que de tels logiciels soient efficaces, il est nécessaire de marier la représentation XML avec les outils linguistiques.

Cet article présente une méthode qui permet de placer au cœur de l'analyse linguistique le balisage des documents. En effet la sémantique des balises peut se révéler cruciale durant le traitement d'un document. Si un titre est marqué à l'aide d'une balise spécifique, il est clair que cela a un impact sur le contenu textuel associé à cette balise. De même, si dans un texte une série de mots est placée en italique ou en gras, cela aussi doit être pris en compte par la grammaire. Enfin, si l'on organise le traitement en suivant l'arborescence XML, cela simplifie aussi naturellement le réalignement du résultat avec les parties analysées.

Nous allons dans un premier temps présenter l'analyseur syntaxique XIP (Xerox Incremental Parser) développé à XRCE (Xerox Research Centre Europe). Puis, dans un deuxième temps nous présenterons le formalisme unifié qui nous permet de gérer en parallèle une grammaire de document et une grammaire de langue. Cette grammaire de document ne porte que sur la structure du document (ses balises et les attributs qui leur sont associés). Son but est de décider en fonction d'une balise donnée de l'intérêt ou non d'appliquer la grammaire de langue, et du degré de finesse d'analyse qui doit être sélectionné (simple analyse morphologique, construction de l'arbre des syntagmes noyaux[1] ou extraction des dépendances). Le résultat est alors intégré dans le document initial sous la forme de nœuds XML.

2 Xerox Incremental Parser (XIP)

XIP (Aït et al. 2000, Roux 99) est un outil de traitement linguistique dont le but est d'analyser du texte tout-venant. XIP fournit un ensemble très riche de règles pour effectuer aussi bien de la désambiguïsation de catégorie, de la construction de syntagmes noyaux et de l'extraction de dépendances syntaxiques (Tesnière 59).

Les règles sont appliquées de façon incrémentale, autrement dit les règles sont ordonnées et appliquées les unes après les autres. XIP gère pour chaque phrase une séquence de nœuds lexicaux ou syntagmatiques associée à un ensemble de dépendances portant sur ces nœuds. Des traits en nombre illimité peuvent être associés à ces nœuds ou à ces dépendances. Ils proviennent soit des lexiques soit de l'application de règles antérieures.

---

[1] Le terme « syntagme noyau » correspond à la notion de chunk défini par Abney (Abney 1991).

L'application de XIP sur un texte consiste à découper d'abord le texte en mots. Ces mots sont ensuite désambiguïsés, puis réunis en syntagmes noyaux. On déduit, grâce à des règles spécifiques, des dépendances syntaxiques entre les nœuds lexicaux en se basant sur les délimitations introduites par les syntagmes noyaux.

XIP offre de nombreuses possibilités d'analyse. On peut par exemple ne calculer que la liste des mots avec leur analyse morphologique, ou bien encore s'arrêter à la seule construction de l'arbre des syntagmes noyaux. Chaque fois évidemment, le temps de calcul est d'autant réduit. XIP peut aussi renvoyer plusieurs types de résultats, un résultat en mode texte qui correspond à l'exemple ci-dessous, ou un résultat de type XML qui obéit à une DTD propriétaire épousant très précisément les résultats de XIP.

Exemple :

La dame referme la porte de la maison.

Nous extrayons la suite de dépendances suivantes ainsi que l'arbre des syntagmes noyaux pour cette phrase :

SUBJ_NOUN(referme,dame)
OBJ_NOUN(referme,porte)
0>GROUPE{SC{NP{La dame} FV{referme}} NP{la porte} PP{de NP{la maison}}}

Ces dépendances ont la signification suivante :

SUBJ_NOUN	indique une relation « sujet » entre un nom et un verbe. Une relation « sujet » entre un pronom et un verbe aurait été indiquée par SUBJ_PRONOUN.
OBJ_NOUN	indique une relation complément d'objet direct entre un nom et un verbe. Le trait « NOUN » indique que le COD est un nom.

Des grammaires XIP à large couverture sont disponibles aujourd'hui aussi bien pour l'anglais que pour le français. Des grammaires plus limitées existent aussi pour le japonais et l'allemand.

2.1 La sortie XML de XIP

Un document XML est un ensemble de nœuds dont certains sont associés à des contenus textuels. Notre objectif est d'appliquer d'abord une analyse linguistique sur ces contenus, puis d'associer au nœud XML ainsi traité le résultat de cette analyse ; ce qui impose évidemment que la sortie de l'analyseur obéisse aussi à un format XML. Or choisir le bon modèle de sortie XML n'est pas aussi trivial qu'il y paraît de prime abord. Tout d'abord, une sortie XML est par définition extrêmement verbeuse, et il faut minimiser la taille de celle-ci, sans que cela nuise à la richesse d'information que peut avoir produit le système en amont. Les travaux récents en normalisation des structures de traits (TEI/ISO 2003) montrent que le choix d'une bonne interface XML ne s'improvise pas. Si nous comparons notre choix à celui de NITE

(Carletta & al.), il est clair que nous ne fournissons pas une annotation linguistique aussi riche et aussi développée que la leur. Par exemple, notre annotation n'introduit aucune notion de synchronisation temporelle nécessaire dans le cas d'une annotation de documents audio vidéo. Notre représentation est beaucoup plus simple et beaucoup moins générale. Cependant, l'objectif reste le même, nous voulons fournir en sortie un document où les parties annotées sont clairement mises en relation avec le résultat de leur annotation.

Exemple :

L'exemple qui suit correspond à l'analyse de la phrase donnée dans l'exemple précédent :

```
<XIPRESULT>
    <LUNIT>
        <NODE num="16" tag="GROUPE" start="0" end="37">
            <NODE num="22" tag="SC" start="0" end="15">
                <FEATURE attribute="SE" value="+"/>
                <NODE num="0" tag="DET" start="0" end="2">
                    <TOKEN pos="DET" start="0" end="2">
La
                        <READING lemma="le" pos="DET"/>
                    </TOKEN>
                </NODE>
...

        </NODE>
        <DEPENDENCY name="SUBJ">
            <FEATURE attribute="NOUN" value="+"/>
            <PARAMETER ind="0" num="4" word="referme"/>
            <PARAMETER ind="1" num="2" word="dame"/>
        </DEPENDENCY>
    </LUNIT>
    <LUNIT>
    …
    </LUNIT>
</XIPRESULT>
```

Il s'agit d'un simple extrait de la structure véritablement générée par XIP. Mais elle illustre la façon dont l'information est renvoyée par l'analyseur. Les liens en particulier qui s'établissent entre les dépendances et les nœuds de l'arbre sont basés sur l'attribut « num », l'identificateur unique de chaque nœud pour une interprétation donnée. Chaque « LUNIT (*Linguistic Unit*) » correspond à une séquence de mots isolée par la grammaire formant une unité linguistique, dans la majorité des cas il s'agit d'une phrase. Lorsqu'un texte est analysé, il est découpé en autant de « LUNIT » que de phrases détectées. Le nœud XML « XIPRESULT » correspond à la racine du document XML ainsi généré.

3 Grammaire de document XML

La manipulation des documents XML est aujourd'hui le centre d'une intense réflexion sur la définition des méthodes et des outils les plus souples pour aborder cette tâche (Chidlovskii 2003, Vion-Dury 2003, Chen et al. 2003). Or l'analyse syntaxique n'est pas un but en soi mais le maillon dans une chaîne plus large de traitements qui visent à enrichir à chaque nouvelle étape un document. Aussi, nous proposons de considérer le traitement linguistique comme une autre forme de transformation de document à l'instar de XSLT par exemple. Pour cette raison, le langage de manipulation que nous allons définir va comme XSLT ou XQuery reposer sur XPath (W3C 99). XPath permet la description des nœuds XML d'un document sous la forme de chemins qui définissent des contraintes portant tout à la fois sur les noms de balise, la comparaison d'attributs ou encore la position relative des nœuds XML les uns par rapport aux autres. Enfin, dernière contrainte, nous voulons pouvoir varier le traitement du contenu textuel de ces balises en fonction de leur sémantique particulière.

La solution que nous proposons consiste à développer une véritable grammaire de document qui va guider l'application de l'analyseur linguistique pas à pas. Cette grammaire très simple doit permettre de décider, en fonction de la nature de la balise, du type de traitement le plus approprié. Nous avons identifié quatre situations parmi les plus courantes :

1. Le contenu de cette balise ne doit pas être analysé.

2. Le contenu de cette balise doit être directement analysé

3. Cette balise est la racine d'un sous-arbre XML dont les contenus textuels doivent être fusionnés avant d'appliquer l'analyseur.

4. Cette balise est par exemple une balise typographique. Les mots ainsi balisés doivent être marqués par un trait spécifique. Cette balise ne doit pas interférer avec l'analyse du contenu plus large où elle se trouve.

3.1 XPath

Comme nous l'avons dit en introduction, chacune des règles de cette grammaire de document repose sur une description de nœuds XML sous la forme d'un XPath. On peut en effet considérer un document XML comme un arbre dont chaque nœud est identifiable par son chemin depuis la racine et par la valeur de ses attributs. XPath permet de décrire un tel chemin en utilisant une syntaxe proche des chemins dansUNIX.

Exemple :

/XIPResult/LUNIT *ce XPath correspond à tous les nœuds LUNIT placés directement sous un nœud XIPResult.*

//Node[@num=10] *ce XPath relatif correspond à tous les nœuds Node dont l'attribut* num *porte la valeur* 10.

XPath dispose de plus de toute une série d'opérateurs pour rejoindre des nœuds parents ou frères à partir d'un nœud donné.

Grâce à XPath, l'exécution n'est pas uniquement guidée par le seul nom de la balise. On peut aussi bien intégrer la position de ce nœud dans l'arborescence du document que la valeur de ses attributs.

3.2 Les instructions

Le formalisme que nous allons décrire ici correspond à celui que nous avons implanté dans XIP. Comme nous l'avons mentionné plus haut, cette grammaire doit offrir plusieurs types d'analyse. Les instructions dont nous disposons sont les suivantes :

- Ignore *cette instruction permet d'ignorer le contenu textuel d'une balise*
- Analyse(*type*) *cette instruction provoque l'analyse du contenu de la balise*
- Fusionne(*type*) *cette instruction effectue la fusion des contenus textuels placé sous le nœud courant.*
- Marque(*trait*) *cette instruction marque le contenu textuel avec le trait* « trait ».

La variable *type* associé à *Fusionne* et à *Analyse* permet de définir le type d'analyse associé à ce XPath. Les valeurs sont les suivantes :

a) token *le texte est simplement segmenté en mots analysés morphologiquement.*

b) noyau *le texte est analysé en syntagmes noyaux.*

c) dépendance *les dépendances sont extraites.*

De cette façon, on peut donc indiquer la finesse d'analyse désirée pour chacun des types de balises.

3.3 Exemple

Supposons que nous voulions analyser des documents contenant des dépêches ayant la forme suivante :

<*NOUVELLES*>
<*DEPECHE* ID="1">
<*DATE*>26-FEB-1987 15:01:01.79</*DATE*>
<*LIEUX*><*D*>Brésil</*D*><*D*>Bahia</*D*></LIEUX>
<*TEXTE*>
<TITRE>Production de cacao à Bahia</TITRE>

<*CORPS*>Des averses ont bouleversé la région de <*GRAS*>**Bahia**</*GRAS*> pendant tout la semaine perturbant la récolte de cacao.</*CORPS*>

</*TEXTE*>

</*DEPECHE*>

</*NOUVELLES*>

Ce document présente un mélange de champs textuels dont le contenu comprend aussi bien des dates, des titres, des mots clefs que du texte libre. Pour l'analyse de ce document, nous ne voulons pas appliquer de façon aveugle l'extraction des dépendances au champ *DATE* par exemple, ou encore au champ *LIEUX,* puisque ces champs se résument à une simple liste de mots clefs. En revanche, nous voulons appliquer ce type de traitement au contenu du champ *CORPS*. Remarquons au passage, que ce champ comprend lui-même du texte balisé en *GRAS*, ce qui impose un traitement particulier avant l'application de l'analyseur syntaxique.

a) Par défaut, les contenus textuels des balises ne sont pas analysés ce que nous traduisons par la ligne suivante :

#defaut->ignore() ;

b) Les balises typographiques doivent être retirées pour éviter de perturber l'analyse du texte, mais comme ce balisage peut revêtir une certaine importance, nous devons le remplacer par une structure de traits, ce que nous traduisons par :

//GRAS->Marque(focus) ;

Le trait « focus » sera automatiquement associé à tous les mots balisés par *GRAS*.

c) Enfin, le contenu de la balise *CORPS* est un contenu complexe, puisqu'il peut contenir d'autres balises. Par conséquent, nous allons devoir effectuer une phase de fusion de tous les « sous-textes » présents avant d'appliquer notre analyse. La description de la balise sera donc la suivante :

//CORPS->fusionne(dépendance) ;

Le moteur fonctionne en traitant les documents en profondeur d'abord, ce qui nous assure un traitement de la balise *GRAS* avant celui de la balise *CORPS*. De cette façon, nous nous assurons que la transformation de cette balise typographique sera effectuée avant celle de la balise *CORPS*.

Ainsi, si l'on applique cette grammaire à notre document, l'analyseur syntaxique recevra la seule phrase suivante :

Des averses ont bouleversé la région de Bahia[focus] pendant tout la semaine perturbant la récolte de cacao.

3.4 Le document résultat

Nous avons présenté cette grammaire de document comme une nouvelle forme de transformation de documents. De fait, l'application de cette grammaire génère un résultat qui doit être associé à chacune des balises qui ont été analysées. Pour ce faire, nous allons exploiter la possibilité qu'offre XIP de fournir un résultat au format XML. A chaque étape, le contenu des balises décrites par la grammaire de document va être analysé, le résultat de ces analyses sera renvoyé en XML et directement intégré sous le nœud même qui aura été analysé. Ainsi, le document résultat comprendra aussi bien le texte initial que le résultat de l'analyse de ce texte, le tout conforme de bout en bout à la norme XML. De cette façon, l'alignement des textes analysés avec le résultat de l'analyse se fera automatiquement.

Exemple :

Si nous utilisons la grammaire de document donné en exemple, le document final en sortie sera le suivant:

…

<TEXTE>

<TITRE>Production de cacao à Bahia</TITRE>

<*CORPS*>Des averses ont bouleversé la région de <*GRAS*>**Bahia**</*GRAS*> pendant tout la semaine perturbant la récolte de cacao.</*CORPS*>

<XIPRESULT>…

</XIPRESULT>

</TEXTE>

Le résultat XML de XIP est simplement intégré au document final sous la forme d'un sous-arbre XML *XIPRESULT*.

4 Conclusion

Nous avons présenté une méthode qui permet l'application de traitements linguistiques qui respectent le format des documents XML. Cette méthode basée sur XPath associe des nœuds XML avec des types de traitement particulier tel que l'extraction de dépendance ou l'extraction des syntagmes noyaux. Le résultat de l'analyse est alors fourni sous la forme d'un document reproduisant le document initial mais dont les parties traitées sont enrichies avec le résultat issu d'un traitement linguistique contrôlé. Cette méthode est, de plus, très générale puisqu'elle n'implique aucun balisage a priori du document et offre une mise en route très rapide, la description de cette grammaire de document se réduisant le plus souvent à quelques règles. L'utilisation de XPath permet par la même occasion un traitement très souple de ces mêmes documents, on peut par exemple décider de ne traiter que les balises DEPECHE dont l'attribut *ID* est inférieur à 10.

Example :

//DEPECHE[@ID<=10]->fusionne(dépendance)

Nous n'avons pas décrit dans le détail le fonctionnement complet du système. Ainsi, par exemple nous avons utilisé *libxml* comme bibliothèque de base pour le traitement des documents XML. l*ibxml* fournit déjà un traitement efficace des expressions XPath, ce qui permet de nous affranchir de la tâche hasardeuse d'écrire notre propre version. Il est aussi possible de spécialiser la grammaire de langue sur la base de XPath. Par exemple, on peut décider de suspendre certaines parties de la grammaire lors de l'analyse d'une balise « titre », en omettant l'extraction des dépendances. Enfin, la plate-forme que nous avons développée offre la possibilité de traiter de gros documents en les découpant en sous-arbres XML qui sont analysés indépendamment.

La fusion d'une grammaire propre au document avec la grammaire d'analyse syntaxique simplifie grandement le traitement des gros fichiers XML. Elle permet de considérer les balises non plus comme un simple bornage de champs textes mais comme une information sémantique supplémentaire pour guider l'analyse du document.

5 Références

ABNEY S., (1991), *Parsing by chunks*, in *Principled-Based Parsing*, R. Berwick, S. Abney, and C. Tenny, editors. Kluwer Academic Publishers, Dordrecht, 1991.

AÏT-MOKHTAR S., CHANOD J.-P., ROUX C., (2002) *Robustness beyond shallowness: incremental dependency parsing*, in a special issue of the NLE Journal, 2002.

AÏT-MOKHTAR S., LUX V., BÁNIK. É., (2003) *Linguistic parsing of lists in structured documents*, in Language Technology and the Semantic Web: 3rd Workshop on NLP and XML (NLPXML-2003), EACL, Budapest, 2003.

CARLETTA, J., KILGOUR, J., O'DONNELL, T., EVERT, S., AND VOORMANN, H. (2003) *The NITE Object Model Library for Handling Structured Linguistic Annotation on Multimodal Data Sets,* in proceedings of the EACL Workshop on Language Technology and the Semantic Web (3rd Workshop on NLP and XML, NLPXML-2003).

CHEN H., TOMPA F.W., *Set-at-a-time Access to XML through DOM,* in DocEng 2003 (Meylan).

CHIDLOVSKII B. (2003), *A Structural Adviser for the XML Document Authoring,* in DocEng 2003 (Meylan).

ROUX C., (1999)**,** *Phrase-Driven Parser*, VEXTAL 99 (Venice).

TESNIERE L., (1959), *Eléments de syntaxe structurale,* Klincksiek (Paris).

VION-DURY J.Y. (2003), *XPath on Left and Right Sides of Rules : Toward Compact XML Tree Rewriting through Node Patterns,* in DocEng 2003 (Meylan).

VOUTILAINEN A., HEIKKILA J., (1994), *An English constraint grammar (EngCG): a surface syntactic parser of English*, Fries, Tottie and Schneider (eds.), Creating and using English language corpora. Rodopi.

TEI/ISO Joint Activity on Feature Structures: Réunions tenues les 4 et 5 Novembre 2003 à l'ATILF, Université de Nancy 2.

W3C (1999), *XML Path Language (XPath) 1.0*, W3C recommendation, November 16th 1999 (http://www.w3.org/TR/xpath)

W3C (1999), *XSL Transformations XSLT Version 1.0,* W3C recommendation, November 16th 1999.