Production de documents auto-explicatifs en annotant les passages ambigus avec le sens choisi par leur auteur

Hervé Blanchon, Christian Boitet et Ali Chouman

CLIPS-GETA
BP 53
38041 Grenoble Cedex 9
{Herve.Blanchon, Christian.Boitet}@imag.fr ; Ali.Chouman@irisa.fr

Résumé

Un système de Traduction Automatisée Fondée sur le Dialogue (TAFD) permet à un auteur monolingue de traduire les documents qu'il rédige dans sa langue maternelle sans avoir à connaître les langues cibles ni les structures linguistiques manipulées par le système. Une traduction véhiculant le sens voulu par l'auteur est produite par un processus de génération à partir d'une représentation interne désambiguïsée. Cette représentation est obtenue au cours d'une session de désambiguïsation interactive lors de laquelle l'auteur répond à des questions qui permettent de lever les ambiguïtés que le système n'a pas pu résoudre automatiquement. En conservant la mémoire des questions et des réponses, on peut produire un Document Auto-Explicatif comme produit dérivé de la traduction. Dans cet article, nous rappelons brièvement nos efforts passés en TAFD et détaillons nos travaux actuels sur un meilleur environnement pour préparer et présenter des Documents Auto-Explicatifs.

Introduction

En Traduction Automatisée Fondée sur le Dialogue (voir aussi [6, 8, 11, 12, 13]), la désambiguïsation interactive permet à un auteur monolingue d'aider le système à produire des traductions de qualité. Dans notre approche, la désambiguïsation interactive est implémentée comme une succession de questions. Chacune des questions permet à l'auteur de choisir, parmi un ensemble de paraphrases exclusives, la paraphrase qui correspond à son intention. Ces choix permettent au système de conserver la bonne interprétation pour produire une traduction de qualité. Pour les besoins de traduction proprement dits, les interprétations concurrentes, les questions et les réponses ne sont pas conservées.

Cependant, conserver ces informations ouvre la voie à une possibilité nouvelle : transmettre un document avec son sens. Les paraphrases sélectionnées représentent l'intention de l'auteur pour chacun des mots, groupes syntagmatiques ou phrases ambigus. Nous proposons de les utiliser pour enrichir le document source en produisant un Document Auto-Explicatif (ce concept a été proposé dans [5]). Un DAE est un « document actif » [9] dans lequel des balises, affichées à la demande, permettent de mettre en relief les segments ambigus. Le lecteur peut sélectionner un segment pour obtenir l'« explication » associée.

Dans la première partie, nous présentons le projet LIDIA et nos deux premières maquettes LIDIA-1 et LIDIA-2. Entre ces deux maquettes, si l'architecture linguicielle que nous avons initialement proposée n'a pas été remise en cause, nous avons simplifié l'architecture logicielle. Avec la maquette LIDIA-3, l'accès aux services de TAFD et la présentation des Documents Auto-Explicatifs se fait avec le logiciel AMAYA qui permet de rédiger un document, d'accéder aux services LIDIA et de visualiser un DAE. Nous décrivons cette maquette dans la seconde partie. La troisième partie propose une démonstration de la nouvelle interface de client LIDIA. Nous proposons enfin quelques perspectives.

1. Premières expériences en TAFD et production de DAE

Notre première maquette, LIDIA-1 [4], démontre nos idées au moyen d'un document HyperCard qui présente, en contexte, des phrases ambiguës en français. Ce document peut être traduit vers l'anglais, l'allemand et le russe. Bien que cette maquette soit réduite du point de vue de sa couverture linguistique, elle montre le potentiel de l'approche. L'architecture linguicielle de cette première maquette a été conservée dans les maquettes suivantes. Nous avons, par contre, opté pour une nouvelle architecture logicielle.

Pour les maquettes LIDIA-2 [2, 3] et LIDIA-3 [7], nous avons utilisé un module de désambiguïsation interactive de l'anglais développé pour désambiguïser des énoncés oraux dans le domaine du tourisme [1]. Ce module était directement intégrable dans la nouvelle architecture logicielle. S'il ne permet pas de mettre en œuvre un analyseur et un générateur comme dans LIDIA-1, il permet quand même d'illustrer nos idées.

Le client LIDIA-2, implémenté en JAVA, manipule un document XML. Ce document contient le texte rédigé par l'auteur et l'historique de la désambiguïsation interactive. Il est filtré pour produire un DAE.

1.1. *Architecture linguicielle*

Chaque phrase du texte source est d'abord analysée (Figure 1) pour produire une structure *mmc-source* (multisolution[1], multiniveau, concrète[2]). La structure (Figure 2) contient trois niveaux d'interprétation linguistique : le niveau des classes syntaxiques (classes terminales) et syntagmatiques (classes non terminales qui donnent le parenthésage de la phrase en groupes syntagmatiques), le niveau des fonctions syntaxiques (rôle syntaxique de chaque nœud dans le groupe), et le niveau des relations logiques et sémantiques (places des arguments attachés aux unités lexicales prédicatives et interprétation sémantique des compléments et circonstants).

Cette structure *mmc* est alors utilisée pour construire un arbre des questions qui seront posées à l'auteur. À l'issue de l'étape de désambiguïsation interactive, le système obtient la structure *umc-source* (unisolution, multiniveau, concrète) non ambiguë choisie par l'auteur. Cette structure *umc* est ensuite transformée en une structure abstraite *uma-source* (unisolution, multiniveau, abstraite[2]).

Un composant de transfert lexical et structural produit ensuite une structure *gma-cible* (génératrice, multiniveau, abstraite). Une première étape de génération produit une structure *uma-cible* qui est homogène à la structure qui serait produite en analysant et en désambiguïsant interactivement le texte cible qui va être généré. Le processus de traduction se termine avec les générations syntaxique et morphologique.

Au cours de la traduction, ou après l'analyse suivie de désambiguïsation interactive uniquement, les informations nécessaires à la construction d'un DAE sont conservées.

1.2. *Architecture logicielle*

Dans la maquette LIDIA-1, les composants distribués (environnements de rédaction, d'analyse, de désambiguïsation et de traduction) communiquent de façon complexe en utilisant plusieurs protocoles de communication (AppleEvent, SMTP, émulation de terminal). Avec LIDIA-2, nous avons mis en œuvre une architecture plus simple et robuste (Figure 3) : tous les composants communiquent de manière homogène via un serveur de communication qui distribue tous les messages en utilisant le protocole Telnet.

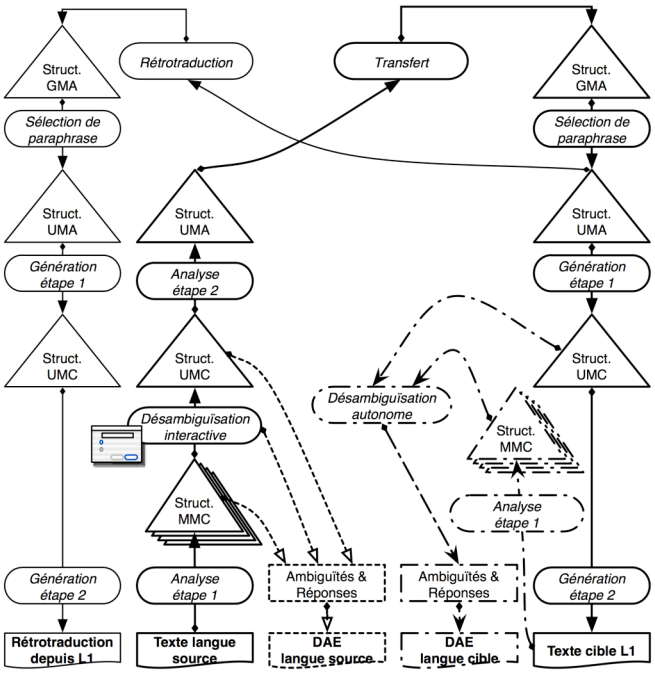

Figure 1. Architecture linguicielle LIDIA

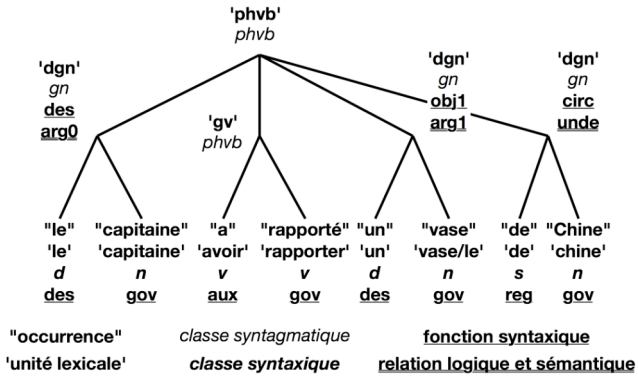

Figure 2 : Niveaux de représentation linguistiques

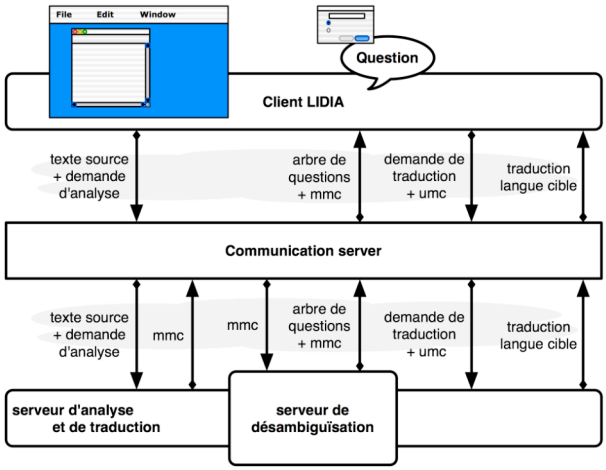

Figure 3. Architecture logicielle LIDIA-2

---

[1] La structure produite est dite multisolution car, pour chaque phrase, l'analyseur produit toutes les analyses vérifiant le modèle syntagmatique, syntaxique et logico-sémantique des grammaires utilisées.

[2] Une représentation d'un texte est dite « concrète » si l'on retrouve directement le texte représenté par un parcours simple de la structure (mot des feuilles pour une structure syntagmatique, parcours infixé pour une structure de dépendances). Sinon, la structure est dite « abstraite ».

2. LIDIA-3, un environnement auteur intégré

Les limitations de LIDIA-2 nous ont conduit à choisir AMAYA[3] pour LIDIA-3 [7]. AMAYA est à la fois un éditeur de documents pour le Web qui offre des services XML évolués [10] et un navigateur. Il permet de créer des documents XHTML conformes à une DTD et inclut une application d'annotation collaborative basés sur RDF, XLink, et XPointer.

2.1. Scénario implémenté

Au document source (*document édité*) est associé un *document compagnon* enrichi avec les données produites lors de la traduction interactive (arbre de questions, réponses, et traductions). Le *document édité* est aussi annoté afin de permettre l'accès aux arbres de questions ou aux explications. Le *document édité* et le *document compagnon* doivent être aussi synchronisés.

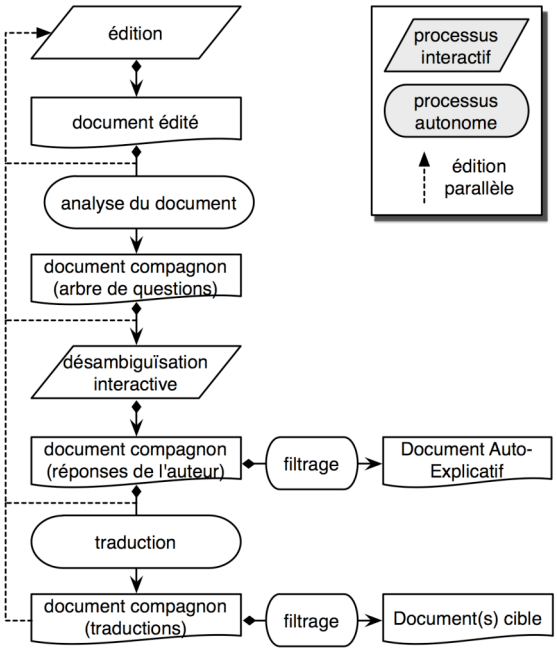

Figure 4. Diagramme fonctionnel de LIDIA-3

2.2. Document compagnon

Le document compagnon (Figure 5), défini par un schéma XML[4], comporte des `paragraphes` composés de `phrases`. Une `phrase` comporte un élément `désambiguïsation` qui est vide si la phrase n'a pas encore été analysée. Dans le cas contraire, l'élément `désambiguïsation` est un ensemble de `questions` portant chacune sur un segment de la phrase défini par son caractère de début et son caractère de fin. Une `question` se compose d'au moins deux `reformulations` (paraphrases) associées au(x) numéro(s) d'analyse(s) concernée(s). Si la phrase n'est pas ambiguë, l'élément `désambiguïsation` est constitué du résultat d'analyse sous forme de *structure-umc*. Une reformulation peut contenir une ou plusieurs autres questions. Après analyse, l'élément désambiguïsation est enrichi comme le montre la Figure 6.

```
<paragraph id="a1">
  <sentence stamp="1" status="nonDesamb">
     Goog morning conference center.
  </sentence>
  <sentence stamp="2" status="nonDesamb">
    <original sourceLang="En">
      Let me pull up my maps to help you.
    </original>
    <translation/><disambiguation/>
  </sentence>
  …
</paragraph>
```

Figure 5. *Document compagnon* avant analyse

```
<disambiguation>
  <question chBegin="16" chEnd="35" questionLang="En" questionType="G">
    <reformulation>
      <text> let me pull up (my maps to help you) </text>
      <refAnalyse>2<refAnalyse>
      <disambiguation> <solution id="2"> (umc) </solution> </disambiguation>
    </reformulation>
    <reformulation>
      <text> to help you, let me pull up my maps </text>
      <refAnalyse>1<refAnalyse>
      <disambiguation> <solution id="1"> (umc) </solution> </disambiguation>
    </reformulation>
  </question>
</disambiguation>
```

Figure 6. *Document compagnon* après analyse, une question est prête

2.3. Interactions entre le document édité et le document compagnon

Chaque modification du document édité doit être reproduite dans le document compagnon. L'analyse et la désambiguïsation doivent être exécutées de nouveau seulement sur les parties du document qui ont été modifiées. Le document compagnon est synchronisé avec le document édité au niveau des éléments XHTML de type `<p>`,`<h1>`, `<h2>`, … en appliquant une transformation (XSLT) sur le document édité.

Lorsqu'un arbre de questions est retourné à AMAYA, le document compagnon est mis à jour. Des balises `<span>` sont ajoutées dans le document édité pour mettre en relief le support des ambiguïtés. Lorsque les questions sont en attente de réponse, le support est en rouge. Il devient bleu dès que l'auteur a répondu. Le document édité est annoté, au moyen de Xpointers, pour permettre de demander l'affichage des questions ou des explications. Ces annotations sont présentées au moyen d'un crayon rouge (Figure 8) pour une question en suspens, et une coche verte (Figure 11) pour une explication.

---

[3] `http://www.w3.org/Amaya/`

[4] `http://wam.inrialpes.fr/people/roisin/lidia/CDoc.xsd`

2.4. *Préparation des questions et des explications*

Pour chaque question, un fichier XHTML est créé. La question peut être une question terminale ou non. Dans le second cas, les questions suivantes sont représentées comme des annotations dans le fichier XHTML qui représente la question courante.

Lorsque l'auteur répond à une question, le document compagnon est mis à jour pour prendre en compte le choix courant. Lorsqu'il a répondu à toutes les questions correspondant à un support d'ambiguïté, l'auteur, et plus tard le lecteur, peut accéder aux explications en cliquant sur la coche verte. À cet instant, un cadre est préparé et affiché à la volée. Il contient les paraphrases sélectionnées par l'auteur lors de l'étape de désambiguïsation interactive.

3. Démonstration de LIDIA-3

Figure 7, l'auteur rédige son document. Il demande ensuite l'analyse et un document compagnon est créé L'analyseur traite les trois phrases et produit une analyse multiple pour la seconde et la troisième. Ces analyses sont prises en charge par le module qui va préparer, pour chacune d'entre elles, un arbre de questions. Ces arbres de questions sont retournés à AMAYA qui met à jour le document compagnon et produit les annotations pour le document édité (Figure 8).

Lorsque l'auteur clique sur l'annotation signalant un arbre de questions pour la troisième phrase, une première question est proposée (Figure 9). La sélection de la bonne paraphrase se fait au moyen du crayon ce qui signifie que les deux paraphrases proposées vont permettre de sélectionner chacune plusieurs analyses. Lorsque l'auteur choisit l'une des paraphrases, la question suivante est présentée (Figure 10). Comme cette question est la dernière, le choix se fait via un bouton de sélection exclusif.

La troisième phrase est maintenant complètement désambiguïsée, le document compagnon est mis à jour pour refléter les choix de l'auteur. De plus, dans le document édité, le crayon est remplacé par une coche verte (Figure 11). En cliquant sur la coche verte (Figure 12), l'auteur, ou plus tard le lecteur, fait apparaître les explications en reproduisant les paraphrases sélectionnées par l'auteur lors de l'étape de désambiguïsation interactive.

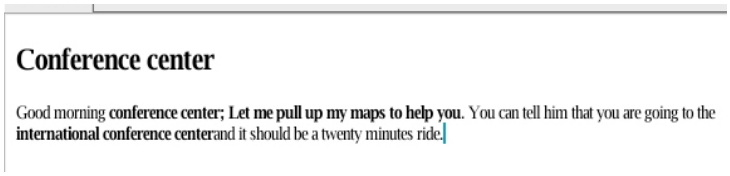

Figure 7. LIDIA-3, *document édité*

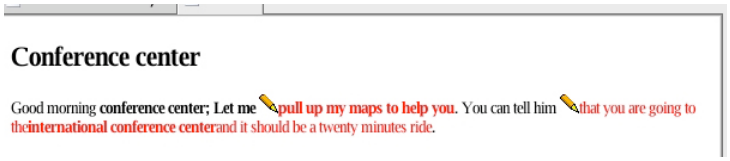

Figure 8. LIDIA-3, *document édité* annoté

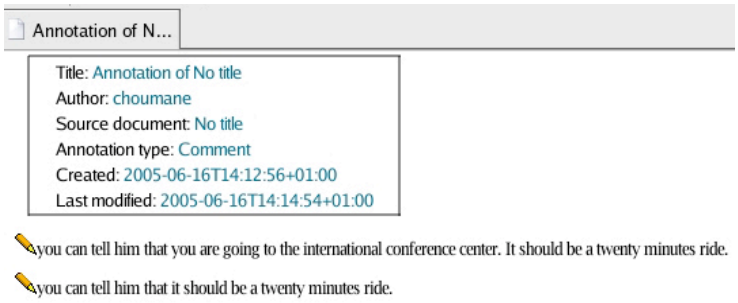

Figure 9. LIDIA-3, question intermédiaire

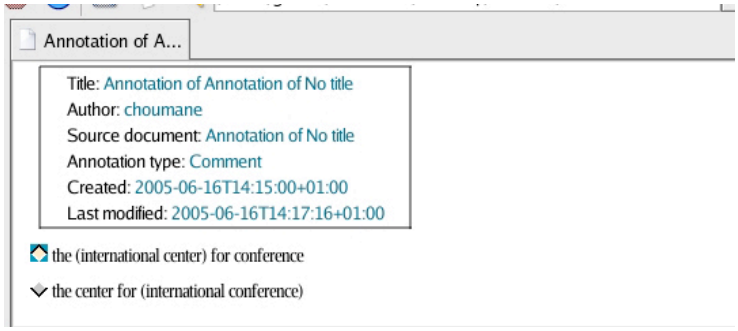

Figure 10. LIDIA-3, question terminale

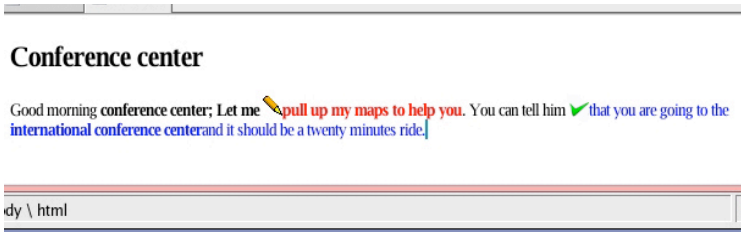

Figure 11. LIDIA-3, *document édité* mis à jour

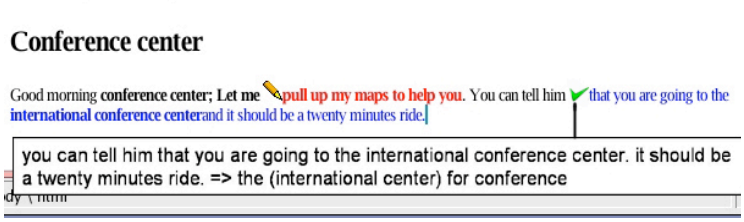

Figure 12. LIDIA-3, présentation des explications

Conclusion et perspectives

Le concept de DAE est apparu dans le contexte d'une recherche en Traduction Automatisée Fondée sur le Dialogue. Ceci explique peut-être pourquoi, bien qu'il ouvre des perspectives fascinantes dans l'exploitation des documents numériques et ait des liens forts avec le Web sémantique, ce n'est pas encore un thème de recherche très actif même au sein de la communauté du Traitement Automatique des Langues Naturelles.

Partant de notre premier prototype de DAE basé sur l'architecture LIDIA-2 et d'un visualiseur primitif, nous avons proposé une bien meilleure implémentation du point de vue de la modélisation des documents (*document édité* et *document compagnon*) et de l'interface utilisateur. L'implémentation actuelle réalisée avec AMAYA permet à l'auteur de produire des documents XHTML. Cependant, le modèle de document compagnon, l'accès aux services d'analyse, de désambiguïsation et de traduction est complètement indépendant de l'environnement de rédaction.

Dans l'architecture linguicielle proposée, un DAE traduit dans une langue cible peut être utilisé pour produire un DAE en langue cible sans intervention humaine. Pour ce faire, il faut disposer d'un analyseur multiple et d'un désambiguïseur pour la langue cible. Il suffit de faire une analyse multiple de chaque phrase et de préparer un arbre de questions. Les réponses aux questions de désambiguïsation sont celles qui permettent de conserver l'analyse qui correspond à la structure *umc-cible* produite lors de l'étape de génération vers la langue cible comme le montre la Figure 1.

Dans le futur nous envisageons de réaliser une implémentation plus réaliste dans un cadre multilingue qui permette la production de DAE en langue cible. Nous prévoyons d'utiliser des analyseurs de l'anglais, du russe et du chinois produisant des structures UNL[5] multiples et de développer les modules de préparation de désambiguïsation interactive en mettant en œuvre la méthodologie que nous avons proposée et implémentée pour le français et l'anglais.

Bibliographie

[1] Blanchon, H. (1995) *An Interactive Disambiguation Module for English Natural Language Utterances*. Proc. NLPRS'95. Seoul, Korea. Dec 4-7, 1995. vol. 2/2: pp. 550-555.

[2] Blanchon, H. and Boitet, C. (2004) *Deux premières étapes vers les documents auto-explicatifs.* Proc. TALN 2004. Fès, Maroc. 19-21 avril 2004. vol. **1/1**: pp. 61-70.

[3] Blanchon, H. and Boitet, C. (2006) Annotating Documents by Their Intended Meaning to Make Them Self Explaining: An Essential Progress for the Semantic Web. Proc. FQAS 2006, LNAI 4027. Milan, Italy. 7-10 June, 2006. vol. **1/1**: pp. 601-612.

[4] Boitet, C. (1990) *Towards Personal MT: general design, dialogue structure, potential role of speech.* Proc. COLING-90. Helsinki, Finland. August 20-25, 1990. vol. **3/3**: pp. 30-35.

[5] Boitet, C. (1994) *Dialogue-Based MT and self explaining documents as an alternative to MAHT and MT of controlled language.* Proc. Machine Translation Ten Years On. Cranfield, England. Oct. 12-14, 1994. 7p.

[6] Brown, R. D. and Nirenburg, S. (1990) *Human-Computer Interaction for Semantic Disambiguation.* Proc. COLING-90. Helsinki. August 20-25, 1990. vol. **3/3**: pp. 42-47.

[7] Choumane, A., Blanchon, H. and Roisin, C. (2005) *Integrating translation services within a structured editor.* Proc. DocEng 2005 (ACM Symposium on Document Engineering). Bristol, United Kingdom. November 02-04, 2005. vol. **1/1**: pp. 165-167.

[8] Maruyama, H., Watanabe, H. and Ogino, S. (1990) *An Interactive Japanese Parser for Machine Translation.* Proc. COLING-90. Helsinki. August 20-25, 1990. vol. **2/3**: pp. 257-262.

[9] Quint, V. and Vatton, I. (1994) *Making structured documents active.* in Electronic Publishing Origination, Dissemination, and Design. vol. **7**(2): pp. 55-74.

[10] Quint, V. and Vatton, I. (2004) *Techniques for Authoring Complex XML Documents.* Proc. DocEng, ACM Symposium on Document Engineering. Milwaukee, Wisconsin, USA. October 28-30, 2004. vol. **1/1**: pp. 115-123.

[11] Wehrli, É. (1993) *Vers un système de traduction interactif.* in Bouillon, P. and Clas, A. (ed.), La traductique. Les Presses de l'Université de Montréal, AUPELF/UREF. pp. 423-432.

[12] Wood, M. M. (1989) *Japanese for speakers of English: The UMIST/Sheffield Machine Translation Project.* in Peckham, J. (ed.), Recent Developments and Applications of Natural Language Processing. Kogan Page Limited. London. pp. 56-64.

[13] Yamaguchi, M., Takiguchi, N., Kotani, Y. and Nisimura, H. (1993) *An Interactive Method for Semantic Disambiguation in Sentences by Selecting Examples.* Proc. NLPRS'93. Fukuoka, Japon. December 6-7, 1993. pp. 208-222.

[5] http://www.unlc.undl.org/unlsys/unl/