Les documents auto-explicatifs : une voie pour offrir l'accès au sens aux lecteurs

Hervé Blanchon, Christian Boitet

Laboratoire CLIPS-GETA

B.P. 53, 38041 Grenoble Cedex 9, FRANCE

{herve.blanchon, christian.boitet}@imag.fr

Résumé – Abstract :

Dans le cadre du projet LIDIA, nous avons montré que, dans de nombreuses situations de traduction multicible, la Traduction Automatique Fondée sur le Dialogue (TAFD) peut être une très bonne alternative à des outils classiques d'aide au traducteur ou de traduction automatique, même pour des langages contrôlés. Nos premières expériences ont montré le besoin de conserver une mémoire des *intentions de l'auteur* au moyen d'*annotations de désambiguïsation* qui transforment le document source original en un *document auto-explicatif* (DAE). Dans cet article, nous présentons un moyen d'intégrer ces annotations dans un document XML et de les exploiter afin de permettre à des lecteurs de comprendre les intentions de l'auteur. Nous montrerons aussi qu'une fois traduit dans une langue cible, un DAE peut être transformé de façon automatique en un DAE dans cette même langue cible.

In the framework of the LIDIA project, we have shown that, in the context of multi-target translation, Dialogue-Based Machine Translation (DBMT) may be a good alternative to classical professional translator tools or machine translation tools, even in the context of controlled languages. Our first experiments have shown the need to keep a memory of the *author's intentions* using *disambiguating annotations* which transform the original source document into a *self-explaining document* (SED). In this paper we present a mean to integrate those annotations within an XML document and exploit then in order to allows readers to understand the author's intent. We will also show that, once translated into a target language, a SED may be automatically transformed into a SED in this target language.

MOTS-CLÉS : désambiguïsation interactive, traduction automatique fondée sur le dialogue, modèle de document, document actif

KEYWORDS: interactive disambiguation, dialogue-based machine translation, document model, active document

1. Introduction

On rencontre beaucoup de travaux sur le document électronique, que ceux-ci concernent les aspects multimédia ou l'annotation du contenu. Mais nous n'en connaissons aucun qui vise à permettre d'annoter par le sens désiré par l'auteur en cas d'ambiguïté. Si nous sommes, pour l'instant, les seuls à travailler sur cette idée, évidemment porteuse de nombreuses applications, c'est sans doute qu'elle n'a pu être concrètement envisagée qu'après avoir mis en place et expérimenté le nouveau paradigme de la Traduction Automatisée Fondée sur le Dialogue (TAFD).

La TAFD a été proposée dans le cadre du projet LIDIA [Boitet, C., 1990]. Il s'agit de permettre à un auteur monolingue de produire des traductions de qualité des documents qu'il rédige. Dès qu'il rencontre des ambiguïtés qu'il n'est pas capable de résoudre automatiquement, le système pose des questions à l'auteur afin qu'il désambiguïse interactivement son document. Ce type de système doit permettre la traduction de haute qualité de documents qui ne peuvent être traduits faute de temps, de traducteurs ou de solution automatisée satisfaisante.

Au cours du développement de la maquette LIDIA-1 [Boitet, C*., et al.*, 1994, Boitet, C*., et al.*, 1995a], nous avons naturellement été conduits à l'idée que les informations obtenues par le système lors de la phase de désambiguïsation interactive pourraient être conservées afin d'enrichir le document avec le sens qu'il véhicule. Un tel document enrichi serait alors un Document Auto-Explicatif (DAE). Un visualiseur de DAE pourrait montrer au lecteur où se trouvent les ambiguïtés et, à sa demande, préciser le sens choisi par l'auteur.

Dans cet article, nous présentons d'abord la notion d'ambiguïté en langue naturelle et en proposons une définition formelle. Nous décrivons ensuite le projet LIDIA et notre premier démonstrateur, en expliquant aussi comment produire un DAE en utilisant les informations collectées au cours de la phase de désambiguïsation interactive. Nous décrivons ensuite un nouveau démonstrateur qui permet de construire un DAE au cours de l'étape d'analyse, ainsi qu'une première réalisation d'un visualiseur de DAE. Nous concluons brièvement après avoir présenté les perspectives à court et moyen terme de ce travail.

2. Vers une définition formelle de l'ambiguïté

Pour « traiter informatiquement l'ambiguïté », il faut pouvoir définir une ambiguïté comme un objet. Or les définitions usuelles sont seulement du type : « un énoncé est ambigu s'il a, au moins, deux interprétations différentes ». Nous avons besoin d'une notion plus précise permettant en particulier de classer, de rechercher, et de visualiser les ambiguïtés, puis de les traiter pour construire des dialogues de désambiguïsation.

Nous rappelons d'abord ce qu'est l'ambiguïté en linguistique. Nous montrons ensuite les implications de l'ambiguïté pour le Traitement Automatique des Langues Naturelles. Nous proposons finalement une définition formelle utile pour le TALN.

2.1. L'ambiguïté en linguistique

Pour Catherine Fuchs [Fuchs, C., 1996], « pour qu'il y ait **ambiguïté linguistique**, il faut que les différentes significations en jeu soient prédictibles en langue, c'est-à-dire que l'analyse linguistique puisse en rendre compte : tous les **dictionnaires** du français doivent consigner le fait que la suite graphique *bière* correspond à deux unités lexicales, désignant respectivement la « boisson » et le « cercueil » ; toute **grammaire** du français doit prendre en compte le fait que la séquence « N1 faire V-infinitif N2 à N3 »[1] recouvre deux types de relations sous-jacentes correspondant respectivement à « N1 faire que X V N2 à N3 »[2] et à « N1 faire que N3 V N2 »[3]. »

L'ambiguïté peut être **virtuelle** ou **effective** : elle est virtuelle lorsque le contexte linguistique sélectionne l'une des significations ; elle est effective lorsque le contexte linguistique autorise plusieurs interprétations. Dans les exemples suivants, l'ambiguïté de bière reste virtuelle : [exemple 1] *la soif les poussa à commander deux bières au bar* (bière=boisson) et [exemple 2] *les croque-morts descendirent la bière dans la fosse* (bière=cercueil.) Par contre, dans la phrase *comme il faisait très chaud le jour de l'enterrement, on sortit la bière* (bière= ?), l'ambiguïté est effective.

2.2. L'ambiguïté en analyse automatique

Un analyseur automatique fournit deux classes de réponses vis-à-vis de l'ambiguïté : il peut être incapable de repérer certaines ambiguïtés effectives en langue (par exemple s'il connaît uniquement l'unité lexicale de bière qui correspond à boisson) ; il peut aussi considérer certaines ambiguïtés virtuelles comme des ambiguïtés effectives (par exemple il n'a pas les connaissances nécessaires pour choisir le sens boisson pour bière dans l'exemple 1). Dans le premier cas, l'analyseur manifeste un défaut de couverture de la langue. Dans le second, l'analyseur manifeste un défaut d'« intelligence ».

2.3. L'ambiguïté comme un objet formel

Jusqu'à présent, nous avons dit qu'une phrase est ambiguë. En fait, on peut presque toujours réduire la localisation d'une ambiguïté à une partie de la phrase. Nous allons formaliser cela [Boitet, C*., et al.*, 1995b].

Prenons par exemple la phrase suivante :

(1) Do you know where the international telephone services are located?

Le fragment souligné contient une ambiguïté d'attachement qui peut être représentée par deux squelettes [Black, E., et al., 1993] :

[international telephone] services / international [telephone services]

Cependant, il n'est pas suffisant de considérer cette séquence isolément. Prenons comme exemple la phrase suivante :

(2) The international telephone services many countries.

L'ambiguïté a disparu ! Dans la pratique, il est très fréquent que l'ambiguïté relative à un fragment apparaisse, disparaisse puis réapparaisse lorsque l'on

---

[1] Exemple : *j'ai fait porter des fleurs à Lucie*.

[2] soit : *j'ai fait que quelqu'un porte des fleurs à Lucie*, pour notre exemple

[3] soit : *j'ai fait que Lucie porte des fleurs*, pour notre exemple

augmente le contexte du fragment. Ainsi, pour définir proprement une ambiguïté, il faut considérer le fragment à l'intérieur d'une phrase et clarifier l'idée que le fragment utile est le plus court fragment sur lequel l'ambiguïté puisse être observée.

De manière formelle, on peut donc dire qu'un fragment **F** présente une ambiguïté de degré **n** (**n≥2**) dans une phrase **U** s'il possède **n** représentations différentes qui peuvent être utilisées pour produire une représentation complète de **U**. Pour être le support de l'ambiguïté, **F** doit être minimal relativement à l'ambiguïté considérée. Cela signifie que **F**, et les **n** représentations qui lui sont associées, ne peut être réduit à un fragment strictement plus petit **F'**, et ses **n** sous-représentations associées, sans perdre la première propriété.

Dans l'exemple **(1)**, le fragment "the international telephone services" associé à ses deux représentations the [international telephone] services / the international [telephone services], n'est pas minimal car il peut être réduit au fragment "international telephone services", associé à ses deux représentations [international telephone] services / international [telephone services], qui est minimal.

Nous proposons la définition formelle suivante [Boitet, C., 1994] :

Une ambiguïté **A** de degré **n** (n≥2) relative à un système de représentation **R**, peut être formellement définie comme :

A = (U, F, <S_1, S_2, …, S_m>, <s_1, s_2, … , s_n), m≥n où :

– **U** est une phrase complète, appelée le contexte de l'ambiguïté.

– **F** est un fragment de **U**, habituellement, mais non nécessairement connexe, le ***support*** de l'ambiguïté.

– les **S_1, S_2, …, S_m** sont les représentations distinctes de **U** dans **R**, et les **s_1, s_2, … , s_n** leurs sous-parties représentant **F** telles que ∀i,j ; **s_i≠s_j**.

– Condition de minimalité :

Soit **F'** un fragment de **U** strictement contenu dans **F**, et **s'$_1$**, **s'$_2$**, … , **s'$_n$** les parties respectives de **s_1**, **s_2**, … , **s_n** correspondant à **F'**. Il existe alors au moins une paire **s'$_i$**, **s'$_j$** (i≠j) telle que **s'$_i$=s'$_j$**.

Le ***type*** de l'ambiguïté **A** dépend de la différence qui caractérise les **s_i**. Il doit être défini relativement à chaque **R** particulier.

Figure 1 : Définition formelle d'une ambiguïté

3. LIDIA-1 : vers les DAE

3.1. LIDIA : un projet de TA Fondée sur le Dialogue

Les efforts passés visant à améliorer la qualité des traductions produites par des systèmes de TA ont montré que la TA de haute qualité est possible, mais seulement pour des typologies de textes (domaine, style) très contraintes. On peut citer, par exemple, les bulletins météorologiques (METEO, TAUM, anglais↔français), les bulletins boursiers (ALT/Flash, NTT, japonais→anglais), ou les documents techniques (BV/aéro/FE pour les manuels de maintenance d'avions, Systran pour des documents XEROX en anglais contrôlé).

La Traduction Automatique Fondée sur le Dialogue de haute qualité est un nouveau paradigme pour des situations traductionnelles pour lesquelles les autres

approches — fondées sur la langue, fondées sur la connaissance — ne sont pas appropriées [Boitet, C., *et al.*, 1995a]. En TAFD, bien que les sources de connaissances linguistiques soient encore cruciales, et que des connaissances extra-linguistiques puissent être utilisées si elles sont disponibles, l'emphase est mise sur la pré-édition indirecte au moyen d'un dialogue de désambiguïsation avec l'auteur afin d'obtenir des traductions de haute qualité sans révision.

La première situation que nous avons considérée est la production de documents multilingues sous la forme de documents HyperCard. HyperCard est un environnement de production de documents hypertextes dont les pages sont des « cartes ». Les cartes contiennent différents types d'objets, dont des champs textuels. Du point de vue linguistique, nous utilisons une approche fondée sur un transfert multiniveau avec des acceptions, propriétés, et relations interlingues. Notre première maquette, LIDIA-1, démontre l'idée avec 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 bien le potentiel de l'approche.

3.2. LIDIA-1 : un premier démonstrateur

L'utilisateur peut activer les traitements LIDIA les plus fréquents grâce à une palette d'outils. La première ligne d'outils (Figure 2), considérée de gauche à droite, permet de traduire l'objet sélectionné, et de voir la progression des traitements, les annotations, et la rétrotraduction en français. La seconde ligne permet de se déplacer parmi les cartes qui composent le document.

Après l'analyse, un bouton (**? !!** - Figure 3) apparaît sur de l'objet à traduire si son contenu est ambigu et nécessite donc une désambiguïsation interactive.

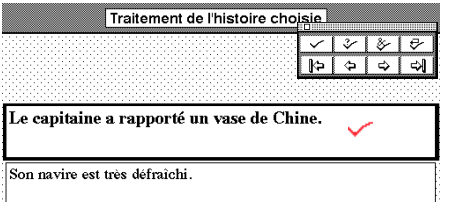

Figure 2 : sélection d'un champ textuel à traduir

Figure 3 : Signalement de questions de désambiguïsation en suspens

Lorsqu'il décide de résoudre les ambiguïtés concernant un objet particulier, l'utilisateur clique sur ce bouton et les questions sont proposées comme ci-dessous, à l'aide de « rephrasages » simples.

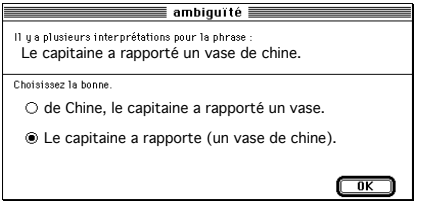

Figure 4 : Désambiguïsation structurale

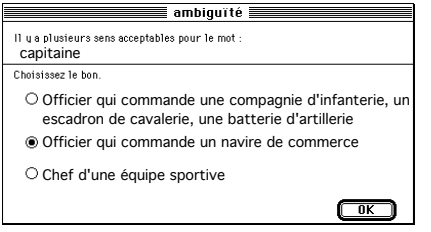

Figure 5 : Désambiguïsation de polysémie

La Figure 6 montre la traduction de la phrase « le capitaine a rapporté un vase de Chine » dans deux contextes différents.

Figure 6 : Traduction en allemand d'une phrase dans deux contextes différents

3.3. Production d'un DAE dans le contexte de la TAFD

Le concept de DAE a été proposé et motivé dans [Boitet, C., 1994]. Nous donnons ici un bref aperçu (Figure 8) des étapes de traitement mises en œuvre et des structures de données produites dans le cadre de notre architecture pour LIDIA-1. Nous montrerons aussi comment la production de DAE en langues source et cible s'y intègre.

Chaque phrase du texte en langue source est d'abord analysée pour produire une structure *mmc-source* (multisolution, multiniveau[4], concrète[5]). 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 abstraite en une structure *uma-source* (unisolution, multiniveau, abstraite[5]).

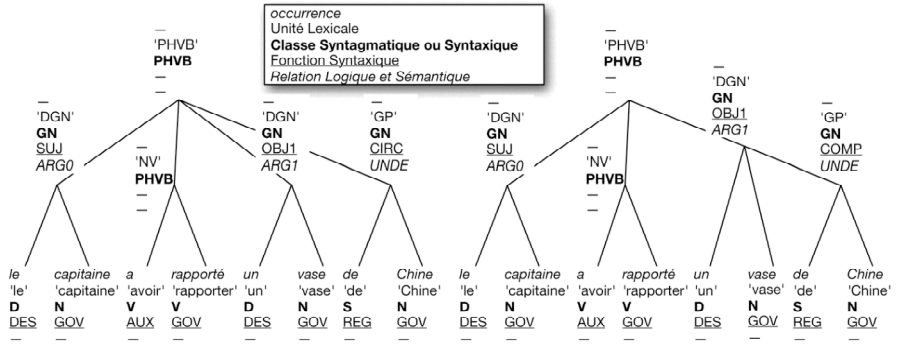

Figure 7 : Exemple de structure mmc

Un composant de transfert lexical et structural produit maintenant une structure *gma-cible* (génératrice, multiniveau, abstraite[5]). Une structure *gma* est plus générale

[4] La structure contient trois niveaux d'interprétation linguistique : le niveau des classes syntaxiques et syntagmatiques, le niveau des fonctions syntaxiques, et le niveau des relations logiques et sémantiques.

[5] Une représentation « concrète » d'un texte est telle qu'on retrouve le texte représenté grâce à un parcours canonique de la structure (mot des feuilles pour un arbre de constituants, parcours infixé pour un arbre de dépendances). Sinon, la structure est dite « abstraite ».

et génératrice qu'une structure *uma* car les niveaux de surface (fonctions syntaxiques, catégories syntagmatiques …) peuvent ne pas être renseignés. Dans ce cas, ce sont des préférences du transfert qui les instancieront.

L'étape de sélection de paraphrase produit une structure *uma-cible* qui est homogène à la structure qui serait produite en analysant puis en désambiguïsant interactivement le texte cible qui va être généré. Le processus de traduction se termine avec les générations syntaxique et morphologique.

Lors des étapes de la traduction, ou de l'analyse uniquement, les informations nécessaires à la construction d'un DAE sont conservées. La Figure 8 montre un diagramme fonctionnel des processus que nous venons de décrire.

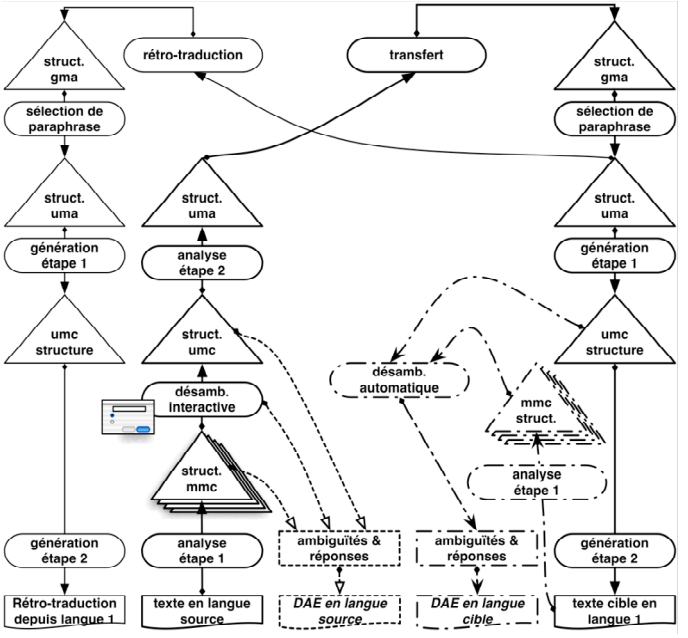

Figure 8 : Organisation linguicielle en TAFD et production de DAE

4. Production d'un DAE avec la maquette LIDIA-2

LIDIA-2 est une version en Java de l'environnement d'accès aux services de TADF. Dans cette version, nous avons utilisé un désambiguïseur de l'anglais [Blanchon, H., 1995] directement utilisable dans la nouvelle architecture d'intégration des composants que nous avons mise en œuvre.

4.1. Exemple de session

L'auteur personnalise d'abord son environnement. Il peut alors créer un nouveau document ou ouvrir un document existant. La fenêtre du document est divisée en deux sections : la partie supérieure est la fenêtre d'édition, la partie inférieure affiche des informations relatives à l'état du traitement du document.

Après que l'auteur a demandé l'analyse du document (Figure 9), les phrases ambiguës sont colorées en brun et les phrases non ambiguës (comme la première phrase de la Figure 9) en vert. On peut lire dans la Figure 9 que le texte contient sept phrases ambiguës et une phrase qui ne l'est pas.

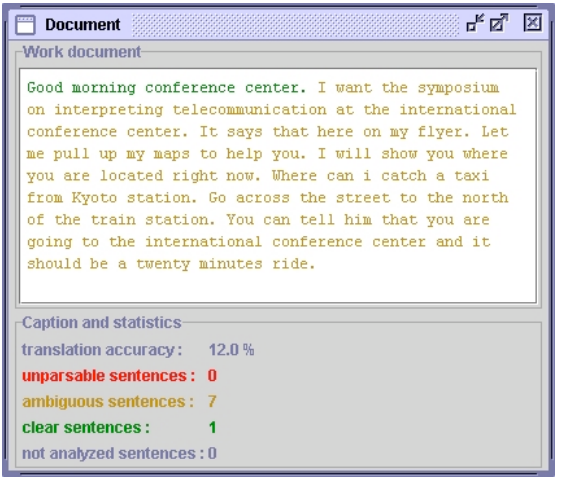

Figure 9 : Fenêtre de document LIDIA-2 (après analyse)

Lorsque l'auteur clique deux fois sur une phrase ambiguë, le dialogue de désambiguïsation relatif à cette phrase est activé. L'ordre des questions correspond à un parcours depuis la racine jusqu'à une feuille dans l'arbre des questions.

Par exemple, lorsque l'auteur choisit la phrase "I want the symposium on interpreting telecommunications at the international conference center", une première question (Figure 10) lui est proposée.

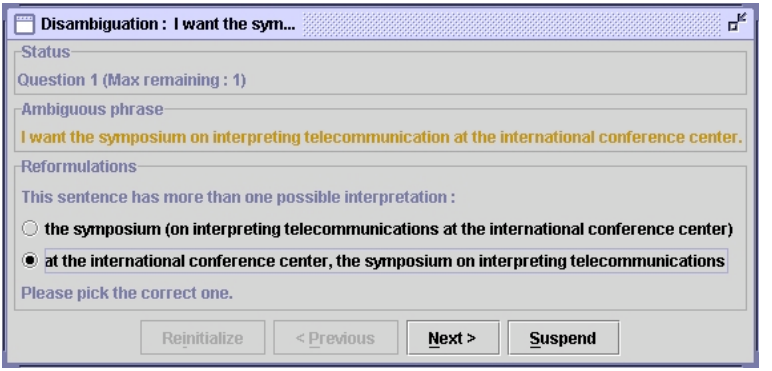

Figure 10 : Première question de désambiguïsation de la phase exemple[6]

[6] La phrase concernée par le dialogue de désambiguïsation peut être comprise de deux façons : « je veux le symposium sur l'interprétation de télécommunications au centre de conférences international(es) » [un objet direct] ou « je veux le symposium sur l'interprétation de télécommunications *qui se déroule* **au centre de conférences international(es)** » [un objet direct et **un circonstant**].

La partie inférieure de la fenêtre lui montre qu'il répond à la première question et qu'il devra ensuite répondre à au plus une autre question. À tout moment, il peut aussi arrêter la session de désambiguïsation (bouton *Suspend*). Lorsqu'il a répondu à une question, il passe à la question suivante avec le bouton *Next*.

En cours de session (Figure 11), l'auteur peut aussi revenir à la question précédente (bouton *Previous*), ou alors recommencer la session (bouton *Reinitialize*). Lorsque qu'il a répondu à toutes les questions (Figure 12), l'utilisateur peut clore la session (bouton *Close*). On doit répondre à toutes les questions en une seule fois.

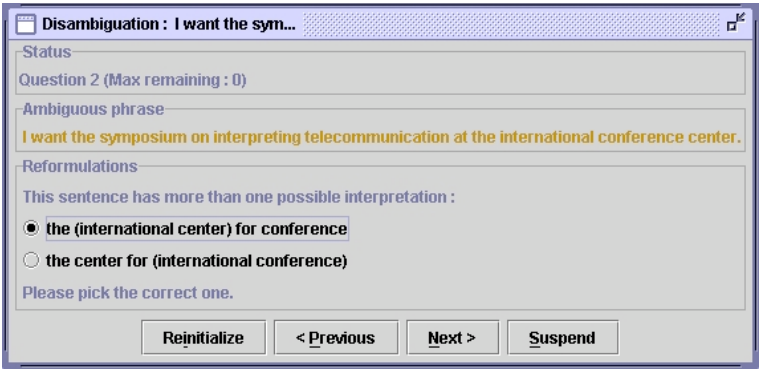

Figure 11 : Seconde question de désambiguïsation de la phrase exemple[7]

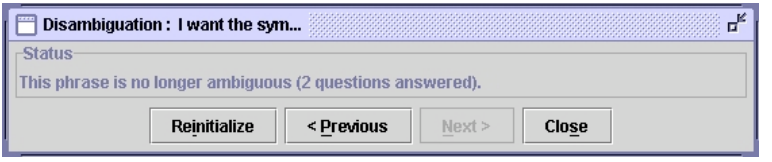

Figure 12 : Fin de la session de désambiguïsation de la phrase exemple

4.2. Document LIDIA-2

Le document XML produit par LIDIA-2 est manipulé par l'API DOM. Lorsque l'auteur ouvre un document existant, sa syntaxe est vérifiée avec l'API SAX.

Ce document contient une entête (**`<description>`**, Figure 13) et un contenu (**`<content>`**, Figure 14) — le texte — sous forme de paragraphes (**`<paragraphe>`**) et de phrases (**`<phrase>`**), entrées par l'utilisateur, qui sont enrichies par les informations collectées lors des différentes étape de traitement.

Pour chaque phrase du texte, le contenu comprend la langue source, le texte de la phrase, l'arbre des questions, ainsi que la ou les traductions obtenues dans les langues cibles choisies. L'arbre des questions est une représentation *à la Lisp* de

[7] S'agit-il d' « un centre international de conférence(s) » ou d' « un centre de conférence(s) internationale(s) » ?

l'arbre des questions produit par le module de désambiguïsation, enrichie par une trace du chemin suivi par l'utilisateur lors de la désambiguïsation effective.

```
<description>
  <title><![CDATA[A trip to Tokyo]]></title>
  <language><![CDATA[ENG]]></language>
  <auteur>
    <firstname><![CDATA[herve]]></firstname>
    <lastname><![CDATA[blanchon]]></lastname>
  </auteur>
 </description>
```

Figure 13 : Descripteur d'un document LIDIA-2

```
<phrase source="ENG" stamp="51054803544695">
  <original><![CDATA[ I will show you where you are located right now.]]></original>
  <question>
    <reformulation choix="NON"><![CDATA[I will show you (where you are located right now).]]>
      <analyse><![CDATA[…]]></analyse>
    </reformulation>
    <reformulation choix="OUI"><![CDATA[right now, I will show you where you are located.]]>
      <analyse><![CDATA[…]]></analyse>
    </reformulation>
  </question>
  <traduction cible="FRA"><![CDATA[Je vais tout de suite vous montrer où vous êtes.]]></traduction>
</phrase>
```

Figure 14 : Extrait d'un fichier LIDIA-2 pour une phrase

4.3. Filtrage vers un DAE

Pour produire le DAE associé au document LIDIA-2 en cours, celui-ci est filtré. Le DAE conserve l'entête du document LIDIA-2. On conserve du contenu son organisation en paragraphes et phrases. Pour chaque phrase, on retient le texte d'origine et la trace du parcours de l'auteur dans l'arbre des questions.

5. Visualisation d'un DAE

Nous concevons un DAE comme un document autonome et « portable » qui doit pouvoir être diffusé sur PC et PDA.

5.1. Objectif et contraintes

Afin de permettre la lecture d'un DAE, nous proposons donc un visualiseur sous la forme d'une application indépendante.

Un tel visualiseur doit permettre à un lecteur de lire le contenu du document et d'appréhender le « sens exact » de que ce l'auteur a voulu dire. À cette fin, le lecteur doit être prévenu que certains segments peuvent avoir plusieurs interprétations (« sens »). Le visualiseur doit alors être capable de révéler, à la demande, le « sens » choisi par l'auteur lors de la phase de désambiguïsation interactive.

Nous avons implémenté, en Java, une première version d'un tel visualiseur.

5.2. Lecture active à l'aide du visualiseur

Le visualiseur permet au lecteur d'ouvrir un DAE dont le contenu textuel est alors affiché (Figure 15).

À ce point, les segments ambigus ne sont pas surlignés (cf. section 6.1). Pour obtenir les informations relatives aux différentes interprétations possibles d'une phrase, l'utilisateur doit cliquer deux fois sur son texte. Une boîte de dialogue apparaît alors. Elle permet au lecteur de naviguer dans l'arbre des questions en voyant les rephrasages sélectionnés par l'auteur (**=> ... <=**) lors de la désambiguïsation, comme le montre la Figure 16.

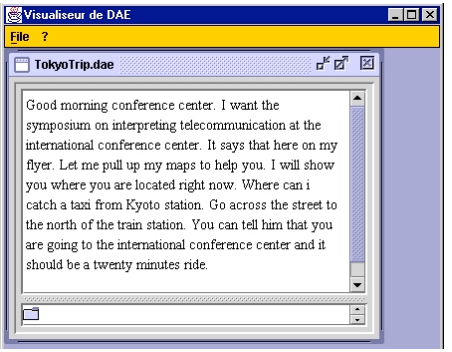

Figure 15 : Interface du visualiseur de DAE

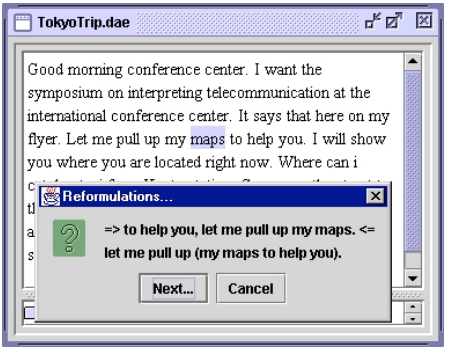

Figure 16 : Affichage de la lecture à retenir pour la phrase sélectionnée[8]

5.3. Perspectives à court terme

Afin d'améliorer l'implémentation de LIDIA-2, nous avons plusieurs objectifs à court terme. Les perspectives à long terme sont détaillées dans la section 6.

Intégrer les linguiciels et le désambiguïseur du français

Notre premier objectif à court terme est d'intégrer dans la nouvelle architecture les modules d'analyse, de désambiguïsation interactive [Boitet, C., *et al.*, 1994, Boitet, C., *et al.*, 1995a], de transfert et de génération (vers l'anglais, l'allemand et le russe) développés pour la maquette LIDIA-1. Cela nous permettrait d'avoir une plate-forme d'expérimentation plus riche.

Rendre la désambiguïsation modifiable

Dans certains cas, il peut être intéressant de refaire la désambiguïsation interactive, soit pour corriger un résultat de traduction (la désambiguïsation interactive aurait dans ce cas été mal faite), soit pour produire une nouvelle traduction afin de montrer l'intérêt de la désambiguïsation.

---

[8] Celui qui parle veut-il dire : « pour vous aider, laissez moi vous présenter mes cartes » ou « laissez moi vous présenter mes cartes conçues pour vous aider ».

Toutes les informations nécessaires sont déjà disponibles dans un document LIDIA-2. Ainsi, une nouvelle désambiguïsation pourrait être effectuée de manière autonome (hors ligne). Si le nouveau parcours de désambiguïsation est le même que le précédent, les bonnes traductions auront déjà été calculées (si l'on vise plusieurs langues cibles). Si le parcours est différent les traductions antérieures devront être écartées et de nouvelles traductions devront être produites (en ligne).

Créer automatiquement des corpus multilingues auto-explicatifs alignés

Comme dit plus haut, LIDIA-2 peut accepter des demandes de traduction vers plusieurs langues cibles. Les traductions sont conservées dans le document LIDIA-2.

Il pourrait donc être intéressant d'exporter un document multilingue déjà aligné au niveau de sa structure. On pourrait même envisager de conserver dans le document LIDIA-2 toutes les structures intermédiaires produites lors du processus de traduction pour calculer automatiquement différents alignements pour chaque phrase (au niveau des mots, des segments, des syntagmes).

Cela pourrait être utile, par exemple, en apprentissage des langues, et aussi pour l'étude contrastive des ambiguïtés. On sait également que les besoins de corpus alignés croissent avec le développement de moteurs statistiques pour le traitement de la langue naturelle, notamment en traduction.

6. Perspectives à long terme

Nos objectifs à long terme sont ceux qui ont un impact sur les modules de la chaîne de traduction (analyse, transfert, génération), et sur le module de désambiguïsation interactive.

6.1. Présenter le support de l'ambiguïté dans un DAE

Afin d'améliorer la présentation d'un DAE, il convient de localiser précisément les ambiguïtés en utilisant leur support (voir la définition au 2.3). Nous présentons brièvement la façon dont les ambiguïtés sont actuellement détectées.

Détection d'ambiguïtés dans les modules de DI actuels

Dans les modules actuels de préparation des arbres de questions, un type d'ambiguïté est décrit comme la cooccurrence, parmi les différentes solutions présentes dans la structure *mmc*, de différents schémas d'arbre — appelés aussi patrons — contenant des variables de nœud (**X**, **Y**, **Z**, **U**, **V**) ou de forêt (**p0**, **p1**, **p2**, **p3**). Ces différents schémas sont regroupés dans un faisceau comme dans la Figure 17.

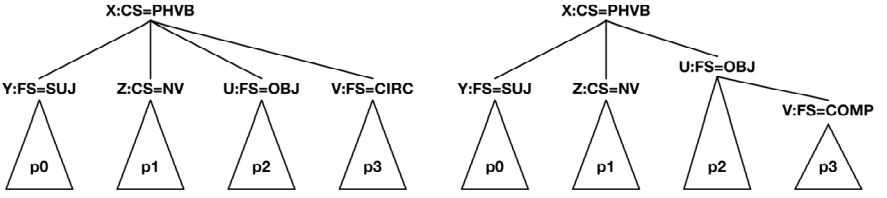

Figure 17 : Faisceau de description d'un type d'ambiguïté

Un item de question est produit pour chacun des patrons qui composent le faisceau sous forme de « rephrasage ». Chaque « rephrasage » est produit en faisant un certain nombre d'opérations sur les variables de forêt apparaissant dans le patron. Ainsi, pour le faisceau de la Figure 17, les méthodes de rephrasage associées au deux patrons sont : Texte(**p2**)Virgule() Texte(**p0**) Texte(**p1**) Texte(**p2**) pour le patron de gauche et Texte(**p0**) Texte(**p1**) Parenthèse(Texte(**p2**), Texte(**p3**)) pour le patron de droite (cf. Figure 4 pour une réalisation de ces méthodes de rephrasage).

Vers un processus de désambiguïsation interactive utilisant le support

Les patrons que nous utilisons actuellement n'utilisent pas la notion de support de l'ambiguïté. En effet, les patrons capturent très souvent un segment plus grand que le support afin de permettre un rephrasage plus compréhensible de l'ambiguïté.

Pour qu'un DAE soit vraiment utile, il faudrait que les supports des ambiguïtés soient indiqués afin que le système puisse indiquer clairement les segments qui posent un problème d'interprétation. Il faut donc que le processus de désambiguïsation interactive fournisse cette information. Deux voies sont possibles.

Dans une première approche, on peut choisir d'attacher la description du support de l'ambiguïté à chaque faisceau. Le support peut, en effet, être défini à partir des variables utilisées dans les patrons.

La seconde approche, proposée dans [Boitet, C.*, et al.*, 1995a], oblige à changer la description des ambiguïtés, en utilisant pour ce faire uniquement leur support. Cependant, pour faire des rephrasages du même type[9] que ceux que nous produisons actuellement, il faudrait décrire les informations supplémentaires à utiliser lors de la fabrication de ceux-ci.

Une première étude a été conduite dans ce sens. On propose de construire automatiquement les patrons sur le support à partir d'un corpus d'analyses multiples. Les informations manquant pour le rephrasage seraient retrouvées au moyen d'heuristiques [Irgadian, F., 2002].

6.2. Autoriser une désambiguïsation interactive incomplète

Dans le contexte de vraies applications, l'analyseur rencontrera un grand nombre d'ambiguïtés. Il est donc possible que, pour certaines phases, l'arbre des questions ait une telle profondeur que l'auteur n'accepte de répondre qu'aux questions cruciales.

Supposons, par exemple, qu'une phrase de longueur **N** possède $\mathbf{k}^{\mathbf{N}}$ interprétations et que les descripteurs d'ambiguïté sont constitués en moyenne de **p** patrons. **(k/b).N** questions désambiguïseraient complètement cette phrase. Si par exemple **(k/b)=1/2**, il y aurait alors 120 questions pour une page de 240 mots. Bien qu'il ne faille pas plus de 10 minutes pour répondre à toutes ces questions[10], si

[9] Nous avons montré [Blanchon, H.*, et al.*, 1996, Blanchon, H.*, et al.*, 1997] que les rephrasages actuellement produits permettent aux utilisateurs de distinguer clairement les différentes interprétations et de choisir la bonne.

[10] Ce temps doit être comparé aux mesures de temps de travail fournies par les traducteurs professionnels. Pour une page et pour chaque langue cible, il faut compter 1 heure pour produire une première traduction et 20 minutes de postédition.

chaque réponse prend 5 secondes, l'auteur peut vouloir consacrer moins de temps à la désambiguïsation interactive.

En d'autres termes, étant donnée une structure *mmc*, quelques réponses à des questions de désambiguïsation et, éventuellement, des préférences utilisateur, le système doit être capable de faire des choix et de produire une traduction unique ou alors une représentation factorisée explicitant les différentes traductions possibles en langue cible. Afin d'implémenter une telle stratégie, il est nécessaire que les modules utilisés puissent mettre en œuvre des techniques heuristiques de désambiguïsation automatique ou soient capables de manipuler des structures ambiguës.

6.3. Certifier le sens

À partir du degré de complétion de la désambiguïsation d'une phrase, et en prenant en compte la « crucialité pour la traduction » des ambiguïtés non résolues, il est sans doute possible de calculer un « niveau de certification du sens » associé à la traduction, et de le calculer au niveau des paragraphes, des sections, etc., jusqu'au document lui-même.

6.4. Créer des DAE en langues cibles

Montrons maintenant comment l'architecture que nous proposons permet aussi de produire des DAE en langue cible. Comme l'étape de génération produit une structure intermédiaire équivalente à une structure d'analyse désambiguïsée (*umc*), il suffit de faire une analyse multiple (*mmc*) des phrases effectivement générées, puis de construire un arbre des questions concernant ces phrases.

Sachant que l'on connaît la structure *umc* à retenir, on peut calculer automatiquement les réponses aux questions de désambiguïsation :pondre à la place d'un lecteur en langue cible. On pourra donc produire un DAE en langue cible sans intervention humaine.

Atteindre cet objectif est cependant difficile en pratique puisqu'il faut disposer d'un analyseur multiple dans chacune des langues traitées (source et cibles). Nous espérons construire un prototype complet implémentant cette idée grâce à des coopérations internationales.

7. Conclusion

Nous avons montré une première implémentation du concept de document auto-explicatif. Cette idée se situe dans le champ de la recherche sur les documents actifs [Quint, V*., et al.*, 1994]. Nous travaillons sur un environnement LIDIA intégré à un éditeur de documents XML *à la Thot* (http://opera.inrialpes.fr/Thot.en.html).

Notre structure de DAE est assez simple car toute l'information contenue dans un tel document n'est pas encore au format XML. Par exemple, la structure *mmc* et l'arbre des questions sont représentés dans un formalisme *à la Lisp*, ce qui nécessite des modules de gestion spécifiques, alors qu'un traitement avec DOM serait plus efficace et portable.

Cependant, ces deux premières étapes (le nouvel environnement LIDIA-2 et le visualiseur de DAE) représentent des résultats originaux, et les perspectives de ce travail sont variées, tant au plan pratique qu'au plan théorique.

8. Références bibliographiques

[Black, E., *et al.*, 1993] Black, E., Garside, R. & Leech, G. (1993). *Statistically-Driven Grammars of English: the IBM/Lancaster Approach.* Rodopi. Amsterdam. 248 p.

[Blanchon, H., 1995] 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.

[Blanchon, H., *et al.*, 1996] Blanchon, H. & Fais, L. (1996). *How to ask Users About What they Mean: Two Experiments & Results. Proc.* MIDDIM'96. Le col de porte, Isère, France. 12-14 Août 1996. vol. **1/1:** pp. 238-259.

[Blanchon, H., *et al.*, 1997] Blanchon, H. & Fais, L. (1997). *Asking Users About What They Mean: Two Experiments & Results. Proc.* HCI'97. San Francisco, California. August 24-29, 1997. vol. **2/2:** pp. 609-912.

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

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

[Boitet, C., *et al.*, 1994] Boitet, C. & Blanchon, H. (1994). *Promesse et problèmes de la "TAO pour tous" après LIDIA-1, une première maquette. in* Langages, "Le traducteur et l'ordinateur"l. vol. **116**, décembre 1994: pp. 20-47.

[Boitet, C., *et al.*, 1995a] Boitet, C. & Tomokiyo, M. (1995b). *Ambiguities & ambiguity labelling: towards ambiguity databases. Proc.* RANLP'95 (Recent Advances in NLP). Tzigov Chark, Bulgaria. 14-16 September, 1995. vol. **1/1:** pp. 13-26.

[Fuchs, C., 1996] Fuchs, C. (1996). *Les ambiguïtés du français.* Ophris. Paris. 184 p.

[Irgadian, F., 2002] Irgadian, F. (2002). *Traduction Interactive Fondée sur le Dialogue. Rap.* Université Stendhal. Rapport de stage de Maîtrise en Industries de la Langue. 20 juin 2002. 109 p.

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