Laboratoire d'Intégration
des Systèmes et des Technologies

DTSI/Service Réalité virtuelle, Cognitique et Interfaces

DTSI/SRCI/XXX/06RT.YYY/Rév. 0 10 octobre 2006

OUTILEX
RAPPORT FINAL : REALISATION D'UN DEMONSTRATEUR D'INTERROGATION CROSSLINGUE

Romaric Besançon

Commissariat à l'énergie atomique
Centre de Fontenay-aux-roses - route du panorama BP 6 92265 Fontenay-aux-roses
Tél : 33 -1 46 54 91 17 - Fax : 33 -1 46 54 75 80

Etablissement public à caractère industriel et commercial
R.C.S. PARIS B 775 685 019

Laboratoire d'Intégration
des Systèmes et des Technologies

DTSI/Service Réalité virtuelle, Cognitique et Interfaces

TITRE :	**RAPPORT FINAL : REALISATION D'UN DEMONSTRATEUR D'INTERROGATION CROSSLINGUE**	DTSI/SRCI/XXX/06RT.YYY
AUTEUR :	**ROMARIC BESANÇON**	
GROUPE :	**LIC2M**	Page 2
PROJET :	**OUTILEX**	

NOMBRE DE PAGES : 24

RESUME :

Le projet Outilex a pour but de réaliser une plate-forme réunissant des outils, des dictionnaires et des grammaires pour le traitement automatique du langage naturel. Dans le cadre de ce projet, le laboratoire LIC2M du CEA-LIST réalise un démonstrateur applicatif utilisant les bibliothèques d'analyse à l'aide d'automates à états finis développées par les partenaires de la plate-forme : ce démonstrateur est un moteur de recherche multilingue, utilisant une analyse linguistique profonde des documents et des requêtes, en différentes langues (français, l'anglais et l'espagnol). Cette analyse linguistique est effectuée par l'outil d'analyse LIMA, développé au LIC2M, dans lequel s'intègre des modules de traitement particuliers qui s'appuient sur les technologies d'automates à états finis d'Outilex. S'appuyant sur cette analyse, le moteur de recherche de démonstration est interrogeable à partir d'une interface Web.

MOTS CLES : Recherche d'information crosslingue, Traitement automatique des langues, Automates à états finis

Rév.	Date	Rédacteur	Vérificateur	Emetteur	Approbateur	Pages modifiées
1						
0		Romaric Besançon			Arnauld Leservot	

Commissariat à l'énergie atomique
Centre de Fontenay-aux-roses - route du panorama BP 6 92265 Fontenay-aux-roses
Tél : 33 -1 46 54 91 17 - Fax : 33 -1 46 54 75 80

Etablissement public à caractère industriel et commercial
R.C.S. PARIS B 775 685 019

TABLE DES MATIERES

1. INTRODUCTION

Dans le cadre du projet Outilex, qui a pour but de réaliser une plate-forme réunissant des outils, des dictionnaires et des grammaires pour le traitement automatique du langage naturel, le CEA réalise un démonstrateur utilisant les bibliothèques d'analyse à l'aide d'automates à états finis développées par les partenaires de la plate-forme Outilex. Le démonstrateur est un moteur de recherche multilingue, utilisant une analyse linguistique profonde des documents et des requêtes, en différentes langues (les langues considérées pour ce démonstrateur sont le français, l'anglais et l'espagnol). Cette analyse linguistique est effectuée par l'outil d'analyse du LIC2M, appelé LIMA (*LIc2m Multilingual Analyzer*), qui intégrera des modules de traitement particuliers qui s'appuient sur les technologies d'automates à états finis d'Outilex.

Nous présentons dans ce rapport l'architecture général des outils d'analyse, d'indexation et de recherche développés au LIC2M. qui comprennent en particulier l'analyseur linguistique LIMA et le moteur de recherche multilingue. Nous présentons ensuite l'intégration dans cette architecture des outils d'analyse d'Outilex, en particulier les conversions nécessaires de format pour les ressources linguistiques, et l'intégration des traitements d'Outilex dans les chaînes de traitement de l'analyseur LIMA. Nous présentons enfin le démonstrateur proprement dit et son interface d'interrogation multilingue, pour effectuer une recherche d'information dans des dépêches de l'AFP publiées au mois d'août 2006 en français, anglais et espagnol.

2. ARCHITECTURE DU MOTEUR DE RECHERCHE MULTILINGUE

2.1. VUE GENERALE

L'architecture du moteur de recherche multilingue du CEA est présentée de façon schématique dans la figure 1.

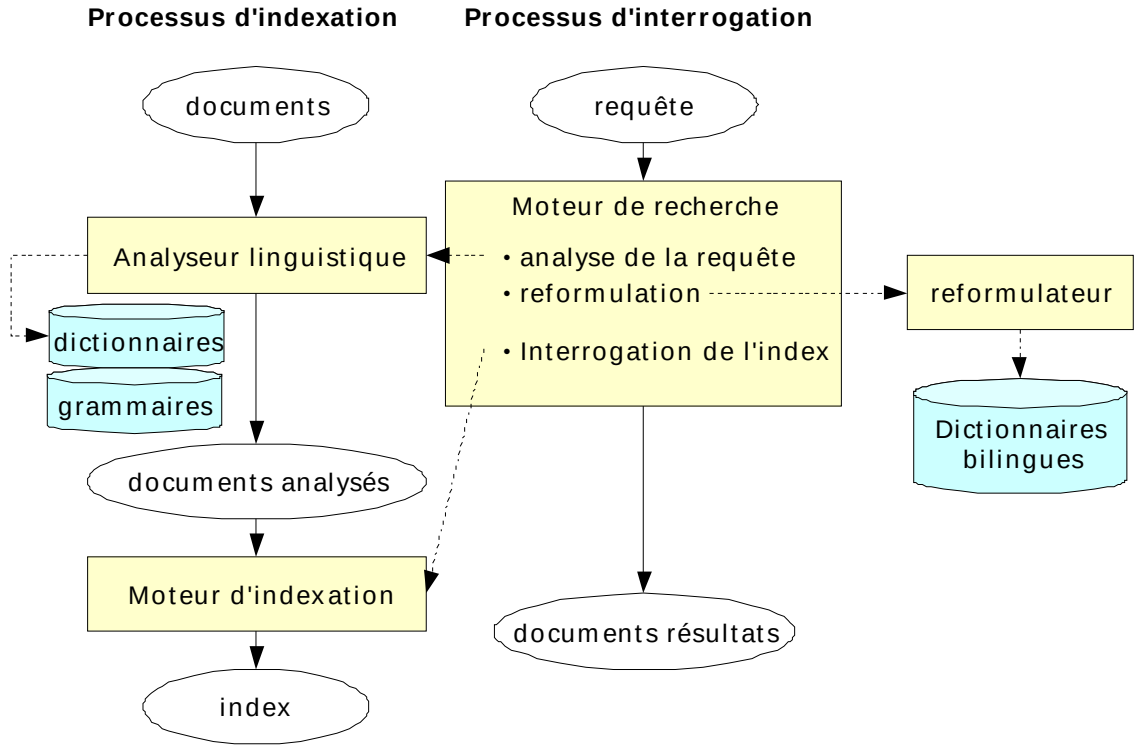


Fig.1 : fonctionnement général du démonstrateur de recherche multilingue du CEA pour le projet Outilex.

Les documents et les requêtes sont passées par l'analyseur linguistique LIMA. Les unités linguistiques pertinentes (lemmes des mots simples, mots composés, ou entités nommées) sont extraites des documents pour être indexés. Les unités linguistiques de la requête sont reformulées pour être confrontées à l'index créé après l'analyse des documents. Les mécanismes plus détaillés de l'analyse linguistique et du fonctionnement du moteur de recherche sont présentés dans les sections suivantes.

Les divers modules (analyse, indexation, reformulation, recherche) peuvent s'articuler soit par le chargement de librairies dynamiques (chargement de toutes les librairies nécessaires au sein du même exécutable), soit par la communication entre différents serveurs, dans une architecture CORBA.

2.2. ANALYSE LINGUISTIQUE

Le système LIMA est un système modulaire d'analyse linguistique permettant d'enchaîner dans une chaîne de traitement différentes unités de traitement pour l'analyse des textes, en vue de leur indexation. Chaque unité de traitement effectue une partie de l'analyse et modifie la représentation interne du résultat de l'analyse d'un texte, qui est représenté, comme dans Outilex, par un graphe permettant de stocker les différentes ambiguïtés d'analyse. Chaque unité de traitement est en pratique implémentée par une classe C++.

Les principales unités de traitement intégrées dans LIMA sont les suivantes :

- **segmentation (*tokenisation*)** : découpage du texte en éléments (*tokens*), se basant sur des éléments de séparation tels que les espaces ou les ponctuations. A cette étape, chaque token est également associé à une catégorie, sur la base de caractéristiques typographiques (présence de majuscules, chiffres etc) ;
- **accès dictionnaire** : récupération, pour chaque mot du texte, de ses différentes interprétations possibles (catégories morphosyntaxiques et lemmes possibles) ;
- **reconnaissance des expressions idiomatiques** : sur la base de patrons morphosyntaxiques, les expressions idiomatiques sont reconnues, ainsi que les formes verbales composées (verbes réflexifs, verbes à particules etc). Les tokens correspondant à ces expressions sont regroupés pour ne former qu'une unité ;
- **traitement des mots inconnus** : découpage des mots à tirets, affectations de catégories morphosyntaxiques par défaut aux mots inconnus en fonction de leur catégorie de tokenisation ;
- **reconnaissance des entités nommées** : sur la base de patrons morphosyntaxiques, les entités nommées de type nom de personnes, de lieux ou d'organisation sont repérées, ainsi que les entités de type date ou montants numériques.
- **désambiguïsation morphosyntaxique (*POS-tagging*)** : les catégories morphosyntaxiques des mots sont désambiguïsés sur la base d'un modèle statistique de fréquence de n-grams. Ce traitement peut être paramétré pour garder plus ou moins d'ambiguïté ;
- **analyse syntaxique** : une analyse syntaxique est ensuite effectuée pour déterminer les relations de syntaxiques entre les éléments de la phrase, sur la base d'une grammaire de dépendance. Cette analyse s'appuie sur un découpage préliminaire des phrases en chaînes nominales et chaînes verbales, et sur la détermination des relations de dépendance syntaxique s'appuyant sur des patrons morphosyntaxiques.
- **construction des mots composés** : des mots composés sont créés, sur la base des relations syntaxiques, en vue de leur indexation.

D'autres unités de traitements sont également disponibles pour traiter des cas particuliers pour d'autres langues (découpage des mots composés en allemand, voyellation ou découpage des enclitiques et proclitiques en arabe, segmentation des phrases en chinois et japonais,...).

L'enchaînement des unités d'indexation est spécifié dans un fichier de configuration au format XML. Chaque unité de traitement peut être paramétrée avec des paramètres spécifiques au traitement, et est associée, dans le même fichier de configuration, aux ressources nécessaires (à l'initialisation de chaque unité, un accès aux ressources

nécessaires est effectuée et les ressources sont chargées si besoin est : ainsi, les ressources sont partagées et ne sont pas chargées inutilement en mémoire).

Lors de l'analyse d'un document complet (au format XML), la chaîne de traitement est appliquée sur chaque unité d'analyse prédéfinie (par exemple, le paragraphe, si les paragraphes sont définis dans le format XML d'entrée).

Un échantillon du fichier XML de configuration de l'analyse est présenté ici, spécifiant une chaîne de traitement particulière (nommée ici *indexer*), et spécifiant quelques unités de traitement de cette chaîne (*tokenizer*, *simpleWord*), ainsi que quelques resources utilisées (le dictionnaire principal *mainDictionary*):

```
<group name="indexer" class="ProcessUnitPipeline" >
  <list key="processUnitSequence">
    <item value="tokenizer"/>
    <item value="simpleWord"/>
    <item value="hyphenWordAlternatives"/>
    <item value="idiomaticAlternatives"/>
    <item value="defaultProperties"/>
    <item value="specificEntitiesRecognizer-beforepos"/>
    <item value="specificEntitiesRecognizer"/>
    <item value="viterbiPostagger-freq"/>
    <item value="sentenceBoundsFinder"/>
    <item value="syntacticAnalyzerChains"/>
    <item value="syntacticAnalyzerDeps"/>
    <item value="compoundBuilderFromSyntacticData"/>
  </list>
</group>
...
<group name="tokenizer" class="Tokenizer">
  <param key="automatonFile" value="LinguisticProcessings/fre/tokenizerAutomaton-fre.xml"/>
  <param key="charChart" value="charchart"/>
</group>
<group name="simpleWord" class="SimpleWord">
    <param key="dictionary" value="mainDictionary"/>
    <param key="confidentMode" value="true"/>
    <param key="charChart" value="charchart"/>
    <param key="parseConcatenated" value="false"/>
</group>
...
<group name="mainDictionary" class="EnhancedAnalysisDictionary">
  <param key="accessKeys" value="globalFsaAccess"/>
  <param key="dictionaryValuesFile" value="LinguisticProcessings/fre/dicoDat-fre.dat"/>
</group>
<group name="globalFsaAccess" class="FsaAccess">
  <param key="keyFile" value="LinguisticProcessings/fre/dicoKey-fre.dat"/>
</group>
```

2.3. INDEXATION ET RECHERCHE

Le module d'indexation prend en entrée le résultat de l'analyse linguistique des documents, à partir duquel il crée des fichiers inversés associant à chaque terme identifié la liste des documents dans lesquels il apparaît. Ces fichiers inversés sont stockés dans une base de données (le gestionnaire de bases de données open source *Firebird*[1] est utilisée dans le démonstrateur).

Pour chaque requête posée, le moteur en effectue l'analyse (avec LIMA), et en extrait les concepts intéressants (mots simples ou composés, entités nommées). Chaque concept est reformulé dans chaque langue cible, en utilisant des dictionnaires de reformulation

[1] http://www.firebirdsql.org/

monolingues et bilingues. Pour les mots composés, une reformulation est construite de façon incrémentale à partir des reformulations des mots simples, avec validation à chaque étape de composition, en testant l'existence du mot composé candidat dans la base sur laquelle on recherche. Les documents contenant les reformulations sont retournés et sont classés : les documents contenant les mêmes concepts de la requête sont mis dans la même classe de pertinence. Comme cette classification se fait sur la base des concepts de la requête et non de leurs reformulations, les classes sont naturellement multilingues. Un poids est attribué à chaque classe en fonction des concepts couverts, avec des poids plus importants pour les mots composés et les entités nommées. Les documents sont donc retournés ordonnés par classe de pertinence, les documents ne sont pas triés à l'intérieur des classes.

3. INTEGRATION DES BIBLIOTHEQUES OUTILEX POUR LA REALISATION DU DEMONSTRATEUR

Le démonstrateur intègre les bibliothèques d'analyse linguistique d'Outilex pour effectuer l'analyse des documents et des requêtes. Du fait de l'architecture flexible et modulaire présente dans LIMA et des similarités d'approches pour l'analyse linguistique, l'intégration des traitements linguistique utilisant les ressources d'Outilex a pu se faire de façon naturelle, dès lors que les ressources ont été converties au format utilisé par ces bibliothèques. Nous présentons ici les conversions de ressources effectués, et l'intégration des outils Outilex dans les chaînes de traitement.

3.1. CONVERSION DES RESSOURCES

Comme Outilex, l'analyseur linguistique LIMA utilise des dictionnaires de langue pour donner les possibilités d'interprétation de chaque mot d'un texte.
Ce dictionnaire contient des traits linguistiques attachés à chaque mot, qui dépendent éventuellement de la langue. Ces traits sont définis dans un fichier de configuration. LIMA utilise aussi, pour le repérage des expressions idiomatiques et des entités nommées en particulier, des patrons morphosyntaxiques écrits sous forme de règles, comparables aux grammaires locales utilisées dans Outilex.

Dans le démonstrateur, ces trois types de ressources sont converties au format Outilex.

3.1.1. CONVERSION DES FICHIERS DE DEFINITION DES PROPRIETES LINGUISTIQUES

Comme dans Outilex, les propriétés linguistiques sont définies dans LIMA dans un fichier de configuration XML. Dans le format LIMA, ce fichier définit des listes de propriétés avec leurs valeurs possibles, certaines de ces propriétés pouvant être considérées comme des spécialisations de valeurs pour d'autres propriétés : en particulier, les catégories grammaticales principales sont définies dans une propriété, et une autre propriété spécialise, pour chacune de ces catégories principales, des catégories secondaires (par exemple, *déterminant* se spécialise en *déterminant article défini*, *déterminant article indéfini*, *déterminant démonstratif* etc). Les spécialisations sont mises à plat, c'est-à-dire que les différents attributs de spécialisation ne sont pas distingués (en ce sens, ce formalisme est plus pauvre que le formalisme utilisé dans Outilex). Dans le format Outilex, on définit ces spécialisations comme des types d'attributs qui sont associés à chaque catégorie grammaticale : chaque catégorie grammaticale a alors un seul attribut associé, qui est la liste de ses spécialisations. Un programme de conversion a été écrit pour effectuer ce changement automatiquement (permettant de généraliser l'opération pour toutes les langues).

Un exemple de la mise en correspondance est donné dans le tableau **Tab 1**. On voit dans ce tableau que les deux formalismes sont très proches et que le programme opère une réécriture simple des définitions.

Format LIMA	***Format Outilex***
<Linguistic_codes> <property name="L_GENDER"> <value name="L_MASC"/> <value name="L_FEM"/> <value name="L_GEN_NEUTRE"/> </property> <property name="L_NUMBER"> <value name="L_SING"/> <value name="L_PLUR"/> <value name="L_NUM_NEUTRE"/> </property> <property name="L_PERSON"> <value name="L_1"/> <value name="L_2"/> <value name="L_3"/> </property> <property name="L_SYNTAX"> <value name="L_TRANS"/> <value name="L_INTRANS"/> <value name="L_PRONOMINAL"/> <value name="L_IMPERS"/> </property> <property name="L_TIME"> <value name="L_PRES"/> <value name="L_IMPFT"/> <value name="L_PASS"/> <value name="L_FUTUR"/> <value name="L_COND_PRES"/> <value name="L_PC"/> <value name="L_PQPFT"/> <value name="L_PASS_ANT"/> <value name="L_FUTUR_ANT"/> <value name="L_COND_PASS"/> </property> <property name="L_MACRO"> <value name="L_NC"/> <value name="L_NP"/> <value name="L_DET"/> *[...]* <subproperty name="L_MACRO_MICRO" parent="L_MACRO"> <subvalues value="L_DET"> <value name="L_DET_ARTICLE_DEF"/> <value name="L_DET_ARTICLE_INDEF"/> <value name="L_DET_ARTICLE_CONTRACTE"/> <value name="L_DET_ARTICLE_PARTITIF"/> <value name="L_DET_POSSESSIF"/> <value name="L_DET_DEMONSTRATIF"/> <value name="L_DET_INDEFINI"/> <value name="L_DET_INDEFINI_SENS_NEG"/> <value name="L_DET_NUMERAL_CARD"/> <value name="L_DET_INTERROGATIF"/> <value name="L_DET_EXCLAMATIF"/> <value name="L_DET_RELATIF"/> </subvalues> <subvalues value="L_NC"> <value name="L_NC_GEN"/> <value name="L_NC_NOTION_TEMPS"/> ...	<lingdef name="fr"> <attrtype name="L_GENDER" type="enum"> <value name="L_MASC"/> <value name="L_FEM"/> <value name="L_NEUTRE"/> </attrtype> <attrtype name="L_NUMBER" type="enum"> <value name="L_SING"/> <value name="L_PLUR"/> </attrtype> <attrtype name="L_PERSON" type="enum"> <value name="L_1"/> <value name="L_2"/> <value name="L_3"/> </attrtype> <attrtype name="L_SYNTAX" type="enum"> <value name="L_TRANS"/> <value name="L_INTRANS"/> <value name="L_PRONOMINAL"/> <value name="L_IMPERS"/> </attrtype> <attrtype name="L_TIME" type="enum"> <value name="L_PRES"/> <value name="L_IMPFT"/> <value name="L_PASS"/> <value name="L_FUTUR"/> <value name="L_COND_PRES"/> <value name="L_PC"/> <value name="L_PQPFT"/> <value name="L_PASS_ANT"/> <value name="L_FUTUR_ANT"/> <value name="L_COND_PASS"/> </attrtype> <pos name="L_DET" cutename="DET"> <attribute name="subcat" type="MICRO_L_DET" [...]/> </pos> <pos name="L_NC" cutename="NC"> <attribute name="subcat" type="MICRO_L_NC" [...]/> </pos> <pos name="L_NP" cutename="NP"> <attribute name="subcat" type="MICRO_L_NP" [...]/> </pos> *[...]* <attrtype name="MICRO_L_DET" type="enum"> <value name="L_DET_ARTICLE_DEF"/> <value name="L_DET_ARTICLE_INDEF"/> <value name="L_DET_ARTICLE_CONTRACTE"/> <value name="L_DET_ARTICLE_PARTITIF"/> <value name="L_DET_POSSESSIF"/> <value name="L_DET_DEMONSTRATIF"/> <value name="L_DET_INDEFINI"/> <value name="L_DET_INDEFINI_SENS_NEG"/> <value name="L_DET_NUMERAL_CARD"/> <value name="L_DET_INTERROGATIF"/> <value name="L_DET_EXCLAMATIF"/> <value name="L_DET_RELATIF"/> </attrtype> <attrtype name="MICRO_L_NC" type="enum"> <value name="L_NC_GEN"/> <value name="L_NC_NOTION_TEMPS"/> ...

Tab 1: Correspondance de formats de définition des propriétés linguistiques entre les systèmes LIMA et Outilex

3.1.2. CONVERSION DES DICTIONNAIRES

Les dictionnaires de langue dans LIMA sont définis par un certain nombre de données initiales, qui sont utilisées dans une chaîne de compilation pour la construction du dictionnaire. Ces données initiales sont des listes de lemmes, avec leurs catégories, et des

tables de flexions. Les formes fléchies sont générées, puis certaines formes provenant de listes fermées additionnelles sont ajoutées au dictionnaire. Le dictionnaire ainsi construit est au format XML. Un programme de conversion a été conçu pour convertir ce format XML en format XML des dictionnaires Outilex. Principalement, les différences entre les deux formats reposent sur des différences de choix des entrées: dans le dictionnaire au format Outilex, une entrée est un lemme, avec sa catégorie grammaticale, auquel sont associées toutes ses formes fléchies et leurs traits morphosyntaxiques particuliers ; dans le dictionnaire au format LIMA, une entrée est une forme fléchie, à laquelle est associée son lemme et l'ensemble de ses propriétés linguistiques, codées dans un format symbolique compact inspiré du format MULTEXT[2] (utilisé en particulier dans la campagne d'évaluation GRACE[3]), mais qui intègre toute la gamme de notre jeu de catégories morphosyntaxiques. Le programme de conversion effectue donc cette inversion d'entrées. Un exemple des entrées relatives au mot « *petit* » dans les deux format de dictionnaires est proposé dans le tableau **Tab 2**, en page suivante.

On voit que la taille du format Outilex des dictionnaires est en général plus importante, principalement en raison de l'énumération explicite de toutes les propriétés linguistiques : le format LIMA a été conçu pour être plus adapté à cette représentation des propriétés linguistiques qui est peu particulière, parce que le choix a été fait d'utiliser beaucoup d'ambiguïtés dans les catégories morphosyntaxiques (en introduisant en particulier des informations positionnelles et/ou fonctionnelles dans les catégories), de façon à rendre la désambiguïsation plus efficace : la désambiguïsation s'appuyant sur des séquences de catégories possibles, le fait de préciser les catégories permet de diminuer le nombre de séquences possibles. En pratique, la taille du fichier XML du dictionnaire au format LIMA est d'environ 54 Mo pour le français, celle du fichier XML du dictionnaire au format Outilex est de 310 Mo. En format compilé, les tailles sont comparables : le dictionnaire Outilex du français fait 9 Mo, celui de LIMA est composé de deux fichiers dont la somme fait environ 7Mo.

Dans le fonctionnement de LIMA, le dictionnaire ne contient que des entrées simples (mots simples), les expressions figées étant repérées par des traitements ultérieurs (reconnaissance des expressions idiomatiques). Bien qu'Outilex permette d'avoir des entrées composées dans son dictionnaire, cette fonctionnalité n'a pas été utilisée ici, par souci de simplicité. On a donc dans le démonstrateur le même fonctionnement que dans LIMA (mots simples dans le dictionnaire, expressions figées reconnues par des grammaires locales).

2 MULTEXT (Multilingual Text Tools and Corpora) : http://aune.lpl.univ-aix.fr/projects/multext

3 GRACE (Grammaire et Ressources pour les Analyseurs de Corpus et leur Évaluation), campagne d'évaluation des analyseurs morpho-syntaxiques pour le français : http://www.limsi.fr/RS99FF/CHM99FF/TLP99FF/tlp10

Format LIMA	***Format Outilex***
<entry k="petit"> <p v="Afb--ms"/> <p v="Afd--ms"/> <p v="Afha-ms"/> <p v="Afhp-ms"/> <p v="Afz--ms"/> <p v="Ncgms-"/> <p v="Nczms-"/> </entry> <entry k="petite"> <p v="Afb--fs"/> <p v="Afd--fs"/> <p v="Afha-fs"/> <p v="Afhp-fs"/> <p v="Afz--fs"/> <p v="Ncgfs-"/> <p v="Nczfs-"/> </entry> <entry k="petites"> <p v="Afb--fp"/> <p v="Afd--fp"/> <p v="Afha-fp"/> <p v="Afhp-fp"/> <p v="Afz--fp"/> <p v="Ncgfp-"/> <p v="Nczfp-"/> </entry> <entry k="petites"> <p v="Afb--fp"/> <p v="Afd--fp"/> <p v="Afha-fp"/> <p v="Afhp-fp"/> <p v="Afz--fp"/> <p v="Ncgfp-"/> <p v="Nczfp-"/> </entry>	<entry> <lemma>petit</lemma> <pos name='L_ADJ'/> <inflected> <form>petit</form> <feat name='L_MACRO_MICRO' value='L_ADJ_QUALIFICATIF_ATT_DU_S'/> <feat name='L_GENDER' value='L_MASC'/> <feat name='L_NUMBER' value='L_SING'/> </inflected> <inflected> <form>petit</form> <feat name='L_MACRO_MICRO' value='L_ADJ_QUALIFICATIF_EPITHETE_DETACHEE'/> <feat name='L_GENDER' value='L_MASC'/> <feat name='L_NUMBER' value='L_SING'/> </inflected> <inflected> <form>petit</form> <feat name='L_MACRO_MICRO' value='L_ADJ_QUALIFICATIF_EPITHETE_PRENN'/> <feat name='L_GENDER' value='L_MASC'/> <feat name='L_NUMBER' value='L_SING'/> </inflected> <inflected> <form>petit</form> <feat name='L_MACRO_MICRO' value='L_ADJ_QUALIFICATIF_EPITHETE_POSTN'/> <feat name='L_GENDER' value='L_MASC'/> <feat name='L_NUMBER' value='L_SING'/> </inflected> <inflected> <form>petit</form> <feat name='L_MACRO_MICRO' value='L_ADJ_QUALIFICATIF_ATT_DU_COD'/> <feat name='L_GENDER' value='L_MASC'/> <feat name='L_NUMBER' value='L_SING'/> </inflected> *[...]* <inflected> <form>petits</form> <feat name='L_MACRO_MICRO' value='L_ADJ_QUALIFICATIF_ATT_DU_COD'/> <feat name='L_GENDER' value='L_MASC'/> <feat name='L_NUMBER' value='L_PLUR'/> </inflected> </entry> <entry> <lemma>petit</lemma> <pos name='L_NC'/> <inflected> <form>petit</form> <feat name='L_MACRO_MICRO' value='L_NC_GEN'/> <feat name='L_GENDER' value='L_MASC'/> <feat name='L_NUMBER' value='L_SING'/> </inflected> <inflected> <form>petit</form> <feat name='L_MACRO_MICRO' value='L_NC_ATT_COD'/> <feat name='L_GENDER' value='L_MASC'/> <feat name='L_NUMBER' value='L_SING'/> </inflected> <inflected> <form>petits</form> <feat name='L_MACRO_MICRO' value='L_NC_GEN'/> <feat name='L_GENDER' value='L_MASC'/> <feat name='L_NUMBER' value='L_PLUR'/> </inflected> <inflected> <form>petits</form> <feat name='L_MACRO_MICRO' value='L_NC_ATT_COD'/> <feat name='L_GENDER' value='L_MASC'/> <feat name='L_NUMBER' value='L_PLUR'/> </inflected> </entry>

Tab 2: Correspondance de formats de définition des dictionnaires entre les systèmes LIMA et Outilex

3.1.3. CONVERSION DES PATRONS MORPHOSYNTAXIQUES.

Les patrons morphosyntaxiques pour la reconnaissance des expressions idiomatiques, des formes verbales composées et des entités nommées se présentent dans le système LIMA sous la forme de règles avec la syntaxe suivante :

DECLENCHEUR : *PARTIE DROITE* : *PARTIE GAUCHE* : *TYPE D'EXPRESSION*

Le déclencheur est l'élément à partir duquel la règle est testée, les parties droites et gauche représentent les contextes précédent et suivant du déclencheur dans le texte. Ces contextes sont définis par des expressions (à la syntaxe proche des expressions régulières) qui représentent des patrons utilisant des mots (formes de surface), des lemmes, des catégories morphosyntaxiques, des catégories de tokenisation, des listes de mots etc. Ces éléments s'agencent avec des opérateurs usuels de répétition, d'alternatives, de négation etc...

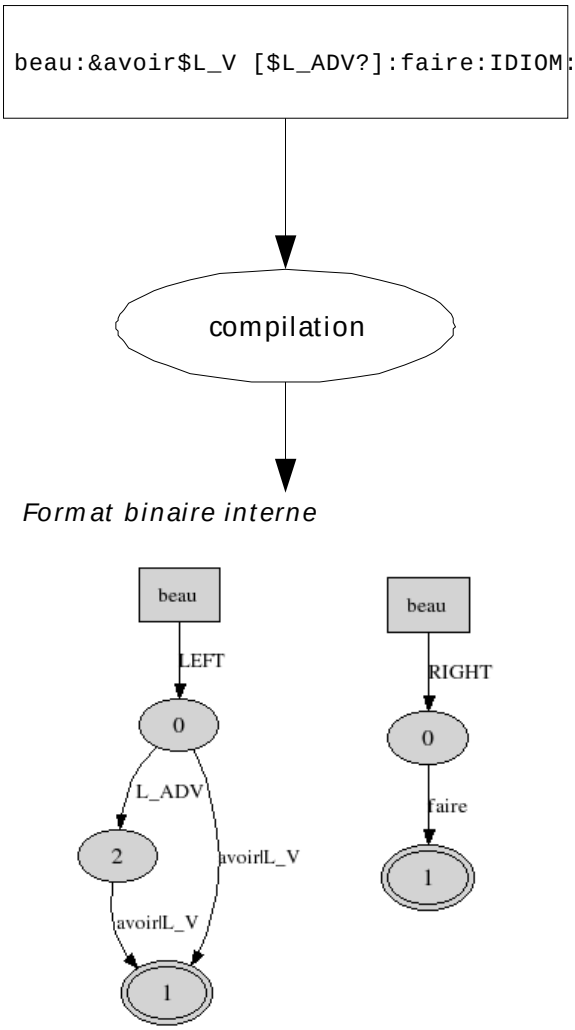
Voici un exemple de règle pour reconnaître l'expression « *avoir beau faire* » : le déclencheur est *beau*, il doit être précédé d'une forme du verbe avoir, éventuellement suivi d'un adverbe, et doit être suivi du mot faire (le signe & indique que la tête de l'expression est le verbe avoir, ce qui indique les traits linguistiques associés à l'expression seront repris de ce mot) :

```
beau:&avoir$V [$ADV?]:faire:IDIOM:
```

Même si le format est moins intuitif qu'un patron linéaire, l'utilisation de la notion de déclencheur permet de choisir l'élément sur lequel la règle est testée (pour éviter de tester systématiquement sur le premier élément, lorsque celui-ci est trop fréquent : on peut ainsi choisir l'élément le plus discriminant du patron pour rendre la recherche plus efficace).

A chacune de ces règles peut être associée une action spécifique, qui correspond à une fonction s'appliquant sur le résultat de l'expression reconnue. Les actions peuvent porter par exemple sur la normalisation des expressions (algorithmes ad hoc pour la normalisation des nombres ou des dates), ou sur l'impact des règles sur la structure d'analyse (création/suppression d'une alternative dans l'automate du texte, ajout/suppression de propriétés linguistiques possibles sur une entrée).

Ces règles sont compilées par un programme spécifique de LIMA, et sont stockées dans une représentation interne au format binaire (utilisant des automates à états finis pour représenter les expressions régulières des parties droites et gauche de la règle). La conversion des règles au format Outilex est réalisée par un programme de conversion prenant en entrée ce format interne du CEA/LIC2M et donnant en sortie des grammaires au format *wrtn* d'Outilex (format XML). Un exemple de cette conversion (pour une règle reconnaissant l'expression idiomatique « *avoir beau faire* ») est proposé dans le tableau **Tab 3**.

Format LIMA	***Format Outilex***
	<wrtn-grammar main="rule.bin_1" sz="5"> <wrtn-pattern sz="5"> <name>rule.bin_1</name> <q id="0"> <tr to="1"> <in><epsilon></epsilon></in> <out w="0">[L_NC</out> </tr> </q> <q id="1"> <tr to="2"> <in> <lex> <form>avoir</form> <pos v="L_V"/> <f n="MICRO_L_V" v="L_V-NONE"/> </lex> </in> <out w="0"></out> </tr> </q> <q id="2"> <tr to="4"> <in> <lex> <pos v="L_ADV"/> <f n="MICRO_L_ADV" v="L_ADV-NONE"/> </lex> </in> <out w="0"></out> </tr> <tr to="3"> <in> <lex> <form>beau</form> </lex> </in> <out w="0"></out> </tr> </q> <q id="3"> <tr to="5"> <in> <lex> <form>faire</form> </lex> </in> <out w="0"></out> </tr> </q> <q id="4"> <tr to="3"> <in> <lex> <form>beau</form> </lex> </in> <out w="0"></out> </tr> </q> <q id="5"> <tr to="6"> <in><epsilon></epsilon></in> <out w="0">]</out> </tr> </q> <q f="1" id="6"/> </wrtn-pattern> </wrtn-grammar>

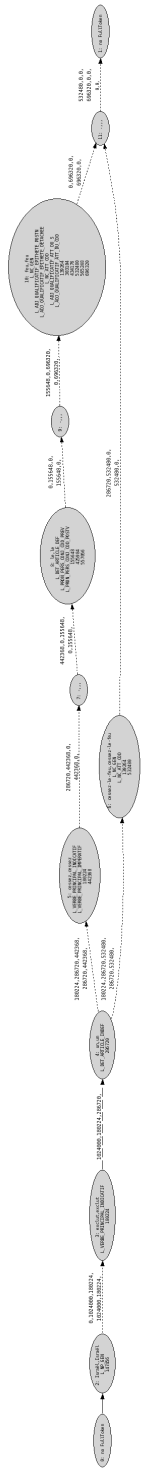
Tab 3: Conversion de format pour la définition des patrons entre les systèmes LIMA et Outilex

Chaque règle utilisée est stockée dans un fichier séparé. Pour donner un ordre d'idées, la transcription de toutes les règles pour la reconnaissance des expressions idiomatiques et des verbes composés en français représentent 1948 patrons ; pour les entités nommées, le nombre de patrons produits est de 7759.

3.2. INTEGRATION DES UNITES DE TRAITEMENT OUTILEX

Plusieurs unités de traitement spécifiques ont été créées pour l'intégration d'Outilex dans l'outil d'analyse LIMA :

- **Tokenisation** : une unité de traitement utilisant le module de segmentation en mots présent dans Outilex. La sortie de ce module est au format XML (cette sortie au format XML est gardée en mémoire pour servir d'entrée au module suivant d'accès au dictionnaire) ;
- **Application des dictionnaires** : cette unité de traitement prend en entrée la sortie au format XML de la segmentation, applique les dictionnaires et construit l'automate du texte de la phrase. Cet automate du texte est stocké dans un fichier externe temporaire (l'utilisation de buffers en mémoire pour le stockage des automates de texte temporaires dans la chaîne de traitement n'est pas opérationnel les bibliothèques Outilex) ;
- **Application des grammaires** : cette unité de traitement applique des grammaires Outilex sur l'automate du texte. Pour être cohérent avec la politique de la version de LIMA existante (qui conditionne en particulier l'écriture des grammaires), l'application des grammaires est utilisée avec l'option de remplacement des nœuds de l'expression retenue par le nœud nouvellement créé portant l'information de l'expression. Cette application des grammaires est a priori utilisée pour deux unités de traitement dans la chaîne : la reconnaissance des expressions idiomatiques et la reconnaissance des entités nommées. Ce traitement utilise aussi des fichiers temporaires pour le stockage des automates de texte modifiés ;
- **Conversion de format** : une conversion du format d'automate de texte Outilex au format LIMA : pour effectuer la suite des traitements linguistiques de LIMA (désambiguïsation morphosyntaxique et analyse syntaxique), l'automate du texte est converti pour être transformé dans le format interne des automates de texte du système LIMA (ce format interne ayant été adapté pour ajouter en particulier des informations d'analyse syntaxique). Les autres unités de traitement de LIMA sont ensuite conservées, pour la création des mots composés à indexer. Cette conversion s'appuie sur le format XML des automates du texte d'Outilex (pour garder une indépendance par rapport à la représentation des automates dans Outilex). La conversion des automates se traduit en pratique par une interversion des états/transitions des automates : le choix a été fait dans Outilex d'attacher les mots du texte aux transitions du graphe, avec une transition par interprétation possible du mot, les nœuds correspondent alors aux successions des mots ; le format LIMA attache quand à lui les mots du texte aux états (toutes les interprétations du mot sont stockés dans le nœud), et les transitions correspondent aux successions. Un exemple de la conversion réalisée est proposé page suivante dans le tableau **Tab 4**, pour la phrase « *Israël exclut un cessez-le-feu* ».

Automate de texte Outilex	***Automate de texte LIMA***
	

Tab 4: Conversion de format pour l'automate du texte entre les systèmes LIMA et Outilex

Les ressources utilisées par plusieurs de ces unités de traitement Outilex, comme la définition des propriétés linguistiques (*lingdef*) sont définies dans LIMA comme des ressources partagées (les autres sont définies comme des ressources propres, spécifiées dans les paramètres de l'unité concernée : par exemple les dictionnaires sont définis comme une ressource propre, utilisé seulement dans l'unité de traitement d'application des dictionnaires et de construction initiale de l'automate du texte).

La définition de la chaîne de traitement pour l'analyse du français en utilisant les bibliothèques Outilex est présentée ici, en spécifiant les unités de la chaîne de traitement, ainsi que la configuration de chacune des unités spécifiques à l'intégration d'Outilex dans le traitement :

```
<group name="indexer" class="ProcessUnitPipeline" >
  <list key="processUnitSequence">
    <item value="tokenizer"/>
    <item value="simpleWord"/>
    <item value="idiomaticExpressions"/>
    <item value="namedEntities"/>
    <item value="analysisStructureConversion"/>
    <item value="viterbiPostagger-freq"/>
    <item value="sentenceBoundsFinder"/>
    <item value="syntacticAnalyzerChains"/>
    <item value="syntacticAnalyzerDeps"/>
    <item value="compoundBuilderFromSyntacticData"/>
  </list>
</group>
...
<group name="tokenizer" class="OutilexTokenizer">
</group>
<group name="simpleWord" class="OutilexMorpho">
  <param key="dictionary" value="Outilex/fre/dicos/dico-lic2m-2.idx"/>
  <param key="lingdef" value="outilexLingDef"/>"
</group>
<group name="idiomaticExpressions" class="OutilexApplyRules">
  <param key="rulesDir" value="/home/besancon/Projets/Outilex/lic2mOutilex/data/fre" />
  <param key="rules" value="idiom-fre.rules" />
  <param key="lingdef" value="outilexLingDef"/>"
</group>
<group name="namedEntities" class="OutilexApplyRules">
  <param key="rulesDir" value="/home/besancon/Projets/Outilex/lic2mOutilex/data/fre" />
  <param key="rules" value="ne-fre.rules" />
  <param key="lingdef" value="outilexLingDef"/>"
</group>
<group name="analysisStructureConversion" class="OutilexToLimaConverter">
</group>
...
<group name="outilexLingDef" class="OutilexLingDef">
  <param key="lingDefFile" value="Outilex/fre/lingdef.xml"/>
</group>
```

3.3. PERFORMANCES

Pour des applications industrielles d'indexation et de recherche, les critères de performance sont importants. En particulier, les temps d'analyse et d'indexation sont importants, étant donné que les corpus envisagés sont en général importants. Dans le démonstrateur, le corpus utilisé est un ensemble de dépêches de l'AFP en français, anglais et espagnol, collectées durant le mois d'août 2006 (sur le fil RSS de l'AFP). Ce corpus contient 824 dépêches en français, 862 dépêches en anglais et 469 en espagnol.

Les dépêches de ce corpus représentent des textes d'environ 400 mots (2,5ko). Le temps moyen d'analyse complète d'une dépêche en français en utilisant l'identification des expressions idiomatiques par les automates d'Outilex est d'environ 7 minutes. Ce temps

augmente jusqu'à 24 minutes lorsqu'on intègre la reconnaissance des entités nommées (beaucoup plus coûteuse car elle contient beaucoup plus de règles). Cette lenteur est en particulier due au choix d'intégration utilisant des fichiers temporaires pour stocker les automates du texte au format Outilex (la création, relecture et suppression de ces fichiers temporaires prenant du temps), et au choix d'utiliser le format d'échange XML des automates de texte, demandant, pour l'analyse de chaque document, la sortie XML de la structure interne efficace et l'analyse de cette sortie avec un parseur XML pour recréer une autre structure interne efficace (le parseur utilisé dans cette intégration est Xerces). Pour comparaison, le temps d'analyse d'un document avec le système LIMA est de l'ordre de quelques secondes. En raison de ce temps de traitement, la reconnaissance des entités nommées n'a pas été intégrée dans l'analyse utilisée dans le démonstrateur.

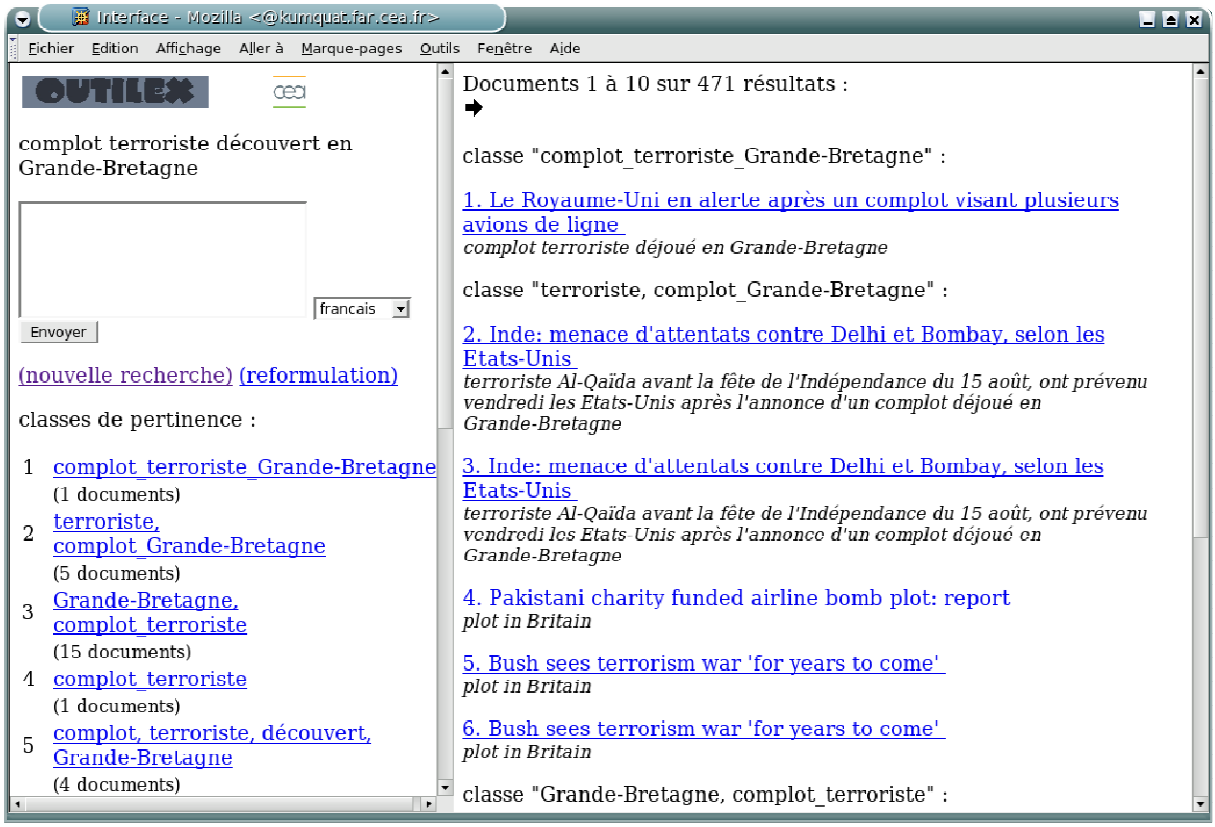
4. INTERFACE DU DEMONSTRATEUR

Le démonstrateur est un moteur de recherche multilingue, dans lequel l'analyse des documents et des requêtes se fait en utilisant les bibliothèques Outilex, et dont l'architecture générale a été présentée dans la section 2.

Un interface d'interrogation a été réalisée pour ce démonstrateur, qui s'appuie sur le cadre de développement d'applications Web *TurboGears* (qui s'appuie sur le framework python *cherryPy*). Cette interface a été adaptée de l'interface de développement et de démonstration réalisée par NewPhenix[4] pour le moteur de recherche du LIC2M. Les classes python de l'interface communiquent avec le serveur C++ du moteur de recherche via CORBA.

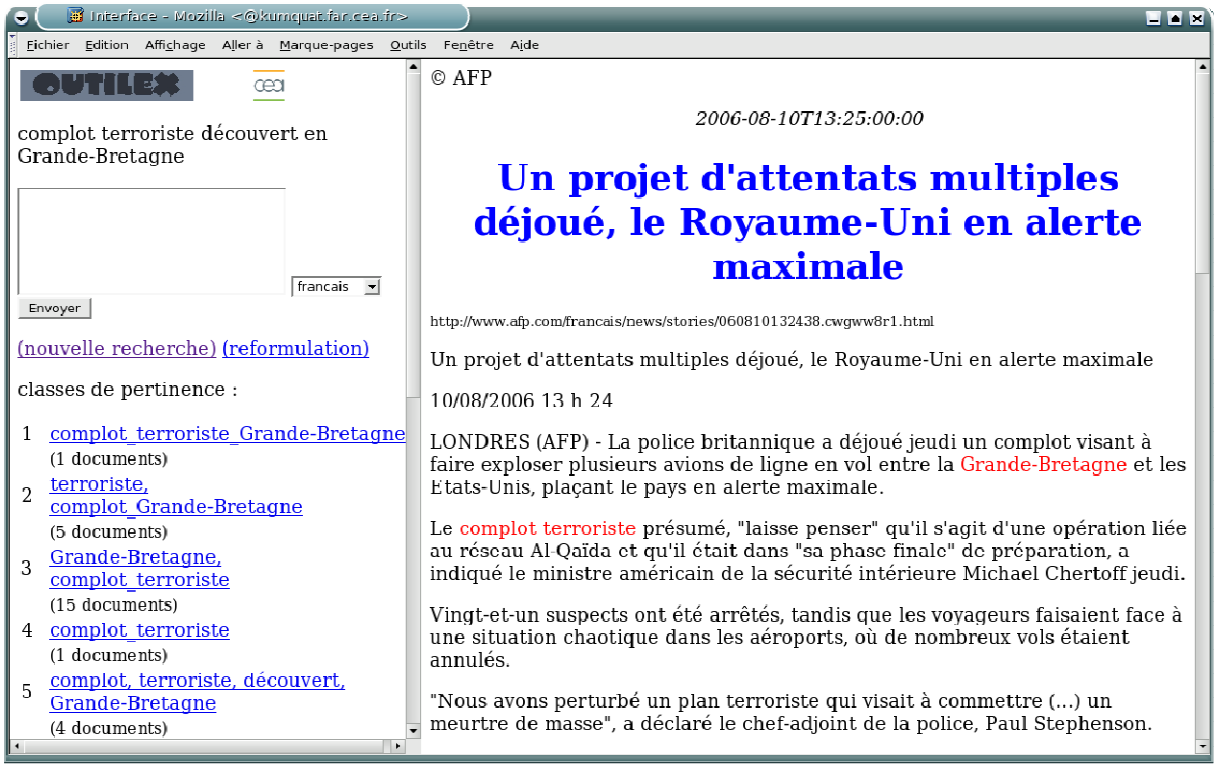
Le démonstrateur utilise comme corpus des dépêches de l'AFP en français, anglais et espagnol, publiées sur le fil RSS de l'AFP pendant le mois d'août 2006 (du 31 juillet au 22 août).

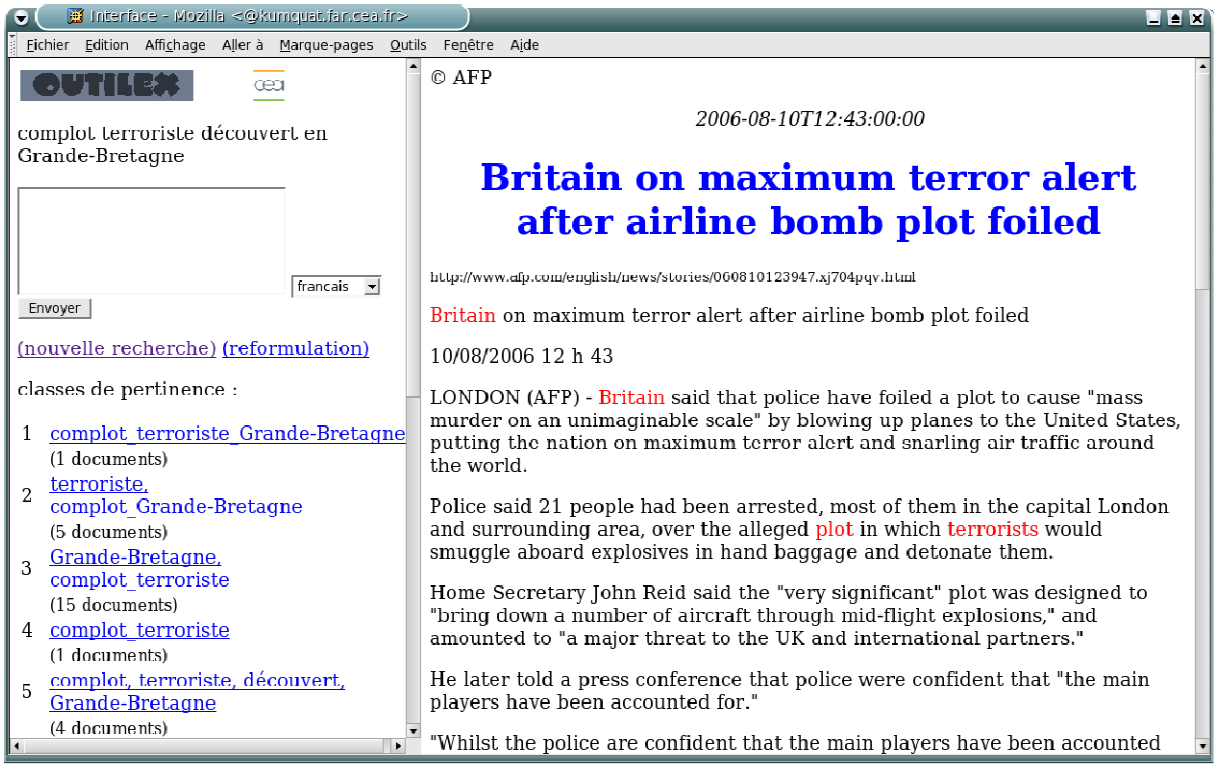
Des copies d'écran lors d'une recherche multilingue sont présentées ci-dessous. La requête utilisée pour cette recherche est « *complot terroriste découvert en Grande-Bretagne* ».

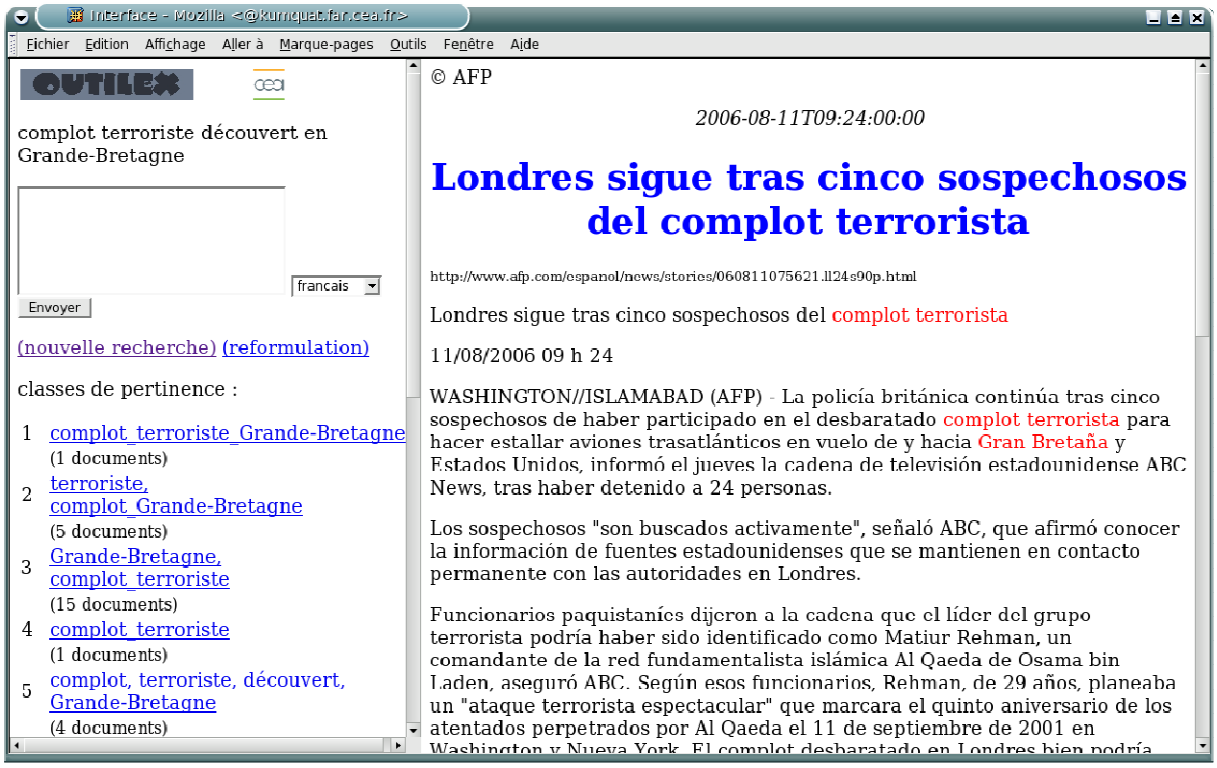


[4] NewPhenix est la start-up qui industrialise et commercialise les technologies développées par le LIC2M (http://www.new-phenix.com/)

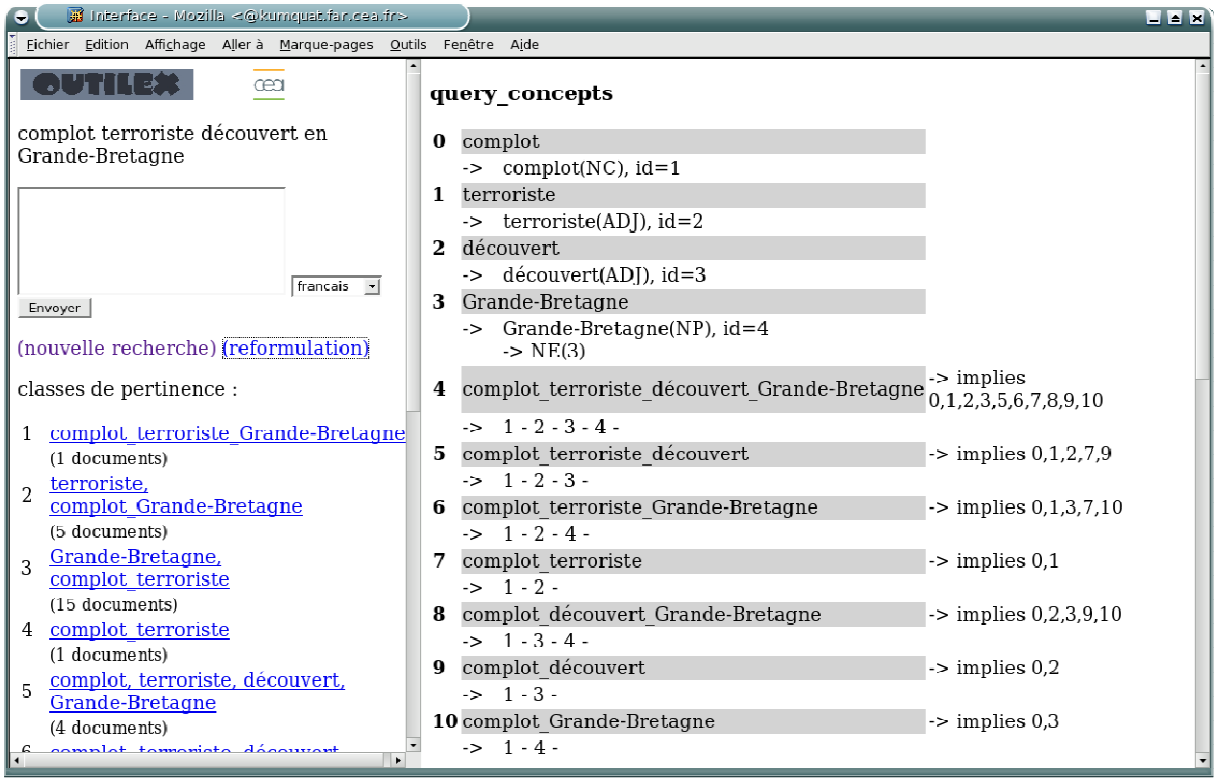
La partie gauche de la fenêtre présente les classes de pertinence, en fonction des termes simples ou composés de la requête ; la partie droite de la fenêtre contient une liste des documents retournés, avec leur classe de pertinence, leur titre, et la forme des concepts de la requête tels qu'ils apparaissent dans le document. Ainsi, la première classe contient les documents dans lesquels on trouve le mot composé « *complot_terroriste_Grande-Bretagne* », et on voit dans la partie droite que ce mot composé a été retrouvé sous la forme « *complot terroriste déjoué en Grande-Bretagne* », ce qui met en évidence que l'analyse syntaxique a permis de retrouver la relation de dépendance syntaxique entre *complot* et *Grande Bretagne* dans la phrase. Les classes de pertinence suivantes contiennent d'autres concepts de la requête. Toutes les classes de pertinence sont multilingues : voici par exemple trois documents en français, anglais, espagnol, qui font tous partie de la classe de pertinence « *Grande-Bretagne, complot_terroriste* ». Les concepts de la requête retrouvés dans les documents sont surlignés.




Interface - Mozilla <@kumquat.far.cea.fr>
Fichier Edition Affichage Aller à Marque-pages Outils Fenêtre Aide
OUTILEX
ceal
complot terroriste découvert en Grande-Bretagne
francais
Envoyer
(nouvelle recherche) (reformulation)
classes de pertinence :
1 complot_terroriste_Grande-Bretagne (1 documents)
2 terroriste, complot_Grande-Bretagne (5 documents)
3 Grande-Bretagne, complot_terroriste (15 documents)
4 complot_terroriste (1 documents)
5 complot, terroriste, découvert, Grande-Bretagne (4 documents)
© AFP
2006-08-10T12:43:00:00
Britain on maximum terror alert after airline bomb plot foiled
http://www.afp.com/english/news/stories/060810123947.xj704pqv.html
Britain on maximum terror alert after airline bomb plot foiled
10/08/2006 12 h 43
LONDON (AFP) - Britain said that police have foiled a plot to cause "mass murder on an unimaginable scale" by blowing up planes to the United States, putting the nation on maximum terror alert and snarling air traffic around the world.
Police said 21 people had been arrested, most of them in the capital London and surrounding area, over the alleged plot in which terrorists would smuggle aboard explosives in hand baggage and detonate them.
Home Secretary John Reid said the "very significant" plot was designed to "bring down a number of aircraft through mid-flight explosions," and amounted to "a major threat to the UK and international partners."
He later told a press conference that police were confident that "the main players have been accounted for."
"Whilst the police are confident that the main players have been accounted

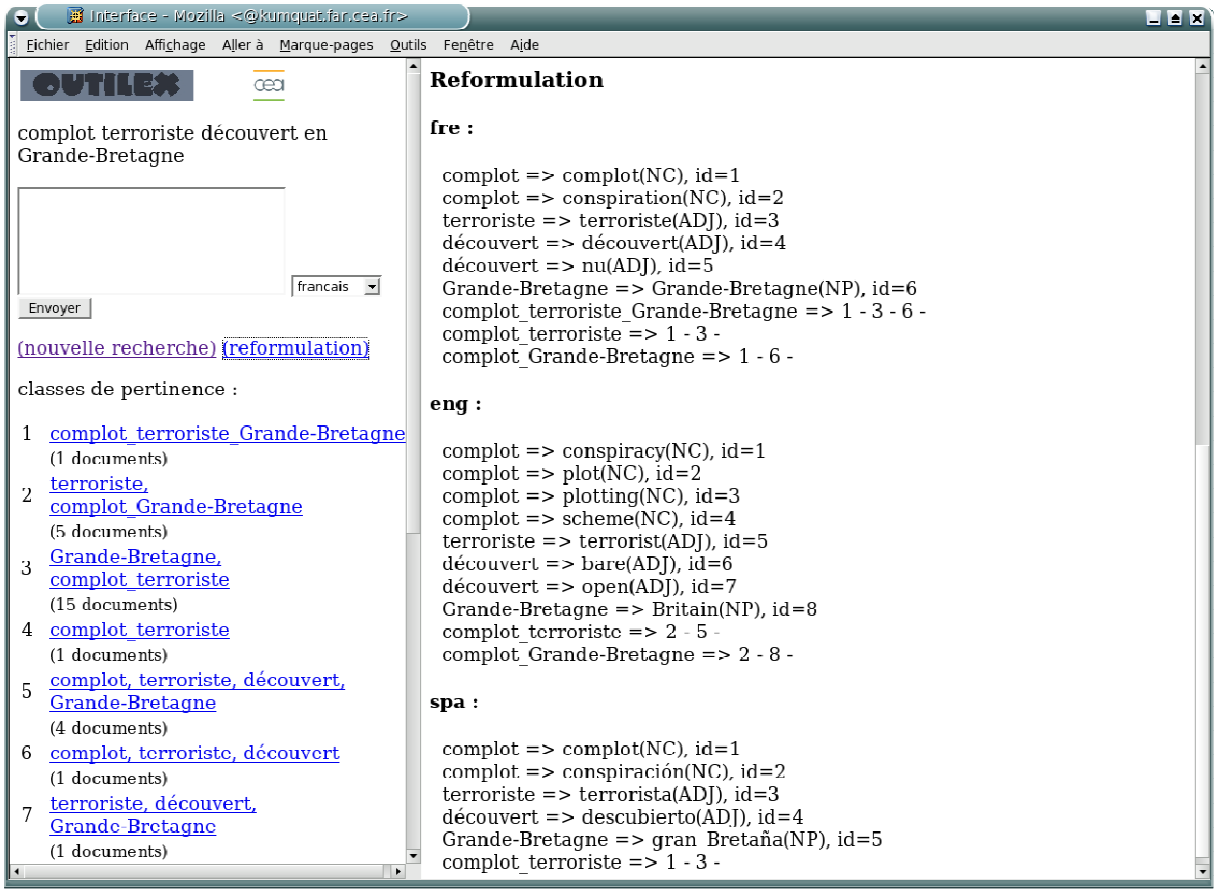

Interface - Mozilla <@kumquat.far.cea.fr>
Fichier Edition Affichage Aller à Marque-pages Outils Fenêtre Aide
OUTILEX
ceal
complot terroriste découvert en Grande-Bretagne
francais
Envoyer
(nouvelle recherche) (reformulation)
classes de pertinence :
1 complot_terroriste_Grande-Bretagne (1 documents)
2 terroriste, complot_Grande-Bretagne (5 documents)
3 Grande-Bretagne, complot_terroriste (15 documents)
4 complot_terroriste (1 documents)
5 complot, terroriste, découvert, Grande-Bretagne (4 documents)
© AFP
2006-08-11T09:24:00:00
Londres sigue tras cinco sospechosos del complot terrorista
http://www.afp.com/espanol/news/stories/060811075621.ll24s90p.html
Londres sigue tras cinco sospechosos del complot terrorista
11/08/2006 09 h 24
WASHINGTON//ISLAMABAD (AFP) - La policía británica continúa tras cinco sospechosos de haber participado en el desbaratado complot terrorista para hacer estallar aviones trasatlánticos en vuelo de y hacia Gran Bretaña y Estados Unidos, informó el jueves la cadena de televisión estadounidense ABC News, tras haber detenido a 24 personas.
Los sospechosos "son buscados activamente", señaló ABC, que afirmó conocer la información de fuentes estadounidenses que se mantienen en contacto permanente con las autoridades en Londres.
Funcionarios paquistaníes dijeron a la cadena que el líder del grupo terrorista podría haber sido identificado como Matiur Rehman, un comandante de la red fundamentalista islámica Al Qaeda de Osama bin Laden, aseguró ABC. Según esos funcionarios, Rehman, de 29 años, planeaba un "ataque terrorista espectacular" que marcara el quinto aniversario de los atentados perpetrados por Al Qaeda el 11 de septiembre de 2001 en Washington y Nueva York. El complot desbaratado en Londres bien podría

L'analyse de la requête, et l'extraction des concepts est aussi visible dans ce démonstrateur (par le biais du lien *reformulation* présent dans la partie gauche de la fenêtre). L'écran suivant montre l'analyse de la requête précédente.



On voit le termes simples qui sont extraits de la requête, et leur catégorie morphosyntaxique associée (on remarque aussi que *Grande-Bretagne* a été repéré comme une entité nommée –NE--, de type lieu –type numérique 3--), ainsi que les termes complexes produits à partir des relations syntaxiques trouvées : la structure des termes est indiquée avec des références aux termes simples ou sous-termes complexes.

La reformulation de chacun des concepts de la requête dans chacune des langues cibles de la collection de documents est aussi accessible sur la même fenêtre. On y voit les reformulations dans toutes les langues des termes simples, ainsi que les reformulations des termes composés, qui se font par partie, par combinaison des reformulations des termes simples (la structure des reformulations de mots composés est indiquée avec des identifiants référençant les reformulations des termes simples). Les reformulations présentées sont celles qui existent dans la collection de documents (les autres ont été filtrées).



Ce démonstrateur est disponible à l'adresse suivante :
`http://`

5. CONCLUSION

Les bibliothèques pour le traitement automatique du langage naturel disponibles dans la plate-forme Outilex ont été intégrées dans un démonstrateur réalisé par le laboratoire LIC2M du CEA-LIST, pour la recherche d'information crosslingue. Cette intégration a permis de mettre en évidence que la plate-forme Outilex répond à un besoin fonctionnel d'analyse des textes telle qu'elle peut être exploitée dans une application complexe et que cette plate-forme peut être utilisée dans le cadre de la réalisation d'une application à caractère industriel. Cette application d'Outilex a également montré la flexibilité de la plate-forme Outilex pour prendre en compte le traitement de différentes langues : le démonstrateur réalisé traite les documents en français, anglais et espagnol. De plus, l'interface graphique de manipulation des grammaires locales est un outil complémentaire à l'approche d'écriture de règles à base d'expressions régulières utilisée originellement dans LIMA (dont la syntaxe peut devenir lourde) . Cette interface peut en effet permettre une visualisation plus intuitive des patrons d'extraction. Néanmoins, les performances de la plate-forme limitent son utilisation pour l'analyse de grandes quantités de texte, qui peut être nécessaire pour ce type d'application. Une intégration plus fine des bibliothèques (qui n'a pas été réalisée par manque de temps), qui permettrait de se passer de l'utilisation de fichiers temporaires et d'éviter les transferts de structures devrait néanmoins permettre d'améliorer ces inconvénients. Cette intégration pourrait aller, dans notre cas, jusqu'à la réalisation d'une désambiguïsation morphosyntaxique et une analyse syntaxique s'appuyant directement sur la structure du texte au format Outilex : l'analyse syntaxique telle qu'elle est intégrée dans LIMA utilise également, pour le repérage des relations de dépendance syntaxique, des patrons morphosyntaxiques (avec certaines propriétés particulières associées), qui pourraient être également mis au format Outilex, qui permettrait de développer un traitement linguistique complet des textes s'appuyant entièrement sur les formats Outilex.