TALN 2004, Fez, 19-22 April 2004

TALN 2004

L'outil de traitement de corpus LIKES

François ROUSSELOT
LIIA-INSA
24, Boulevard de la Victoire
67 084 Strasbourg-CEDEX
rousselot@liia.insa-strasbourg.fr
tél 03 88 14 47 53

Résumé – Abstract

LIKES (LInguistic and Knowledge Engineering Station) est une station d'ingénierie linguistique destinée à traiter des corpus, elle fonctionne pour l'instant sur la plupart des langues européennes et slaves en utilisant des ressources minimales pour chaque langue. Les corpus sont constitués d'un ou plusieurs textes en ASCII ou en HTML, l'interface donne la possibilité de constituer son corpus et d'y exécuter un certain nombre de tâches allant de simples tâches de découpage en mot, de tri ou de recherche de motifs à des tâches plus complexes d'aide à la synthèse de grammaire, d'aide au repérage de relations, d'aide à la construction d'une terminologie. Nous décrivons ici les principales fonctionnalités de LIKES en rapport avec le traitement des corpus et ce qui fait sa spécificité par rapport à d'autres environnements comparables : l'utilisation minimale de ressources linguistiques.

LIKES (LInguistic and Knowledge Engineering Station) is a linguistic engineering environment, build for corpora processing. Its provides different modules able to process most european and slavian languages. Corpora in Likes must be constituted by Texts in TXT format or in HTML texts of one particular. Tasks available are elementary likes classical basic corpora processing tasks (making list of forms, segmenting, sorting) and also more sophisticated as term extraction, help in relation extraction, pattern search, aimed at helping terminology building and ontology building. Main functionalities usefull for corpora processing are presented here.

Keywords – Mots Clés

Traitement de corpus, segments répétés, recherches de relations, automates, transducteurs
Corpus processing, repeated segments, search of semantic relations, automata, transducers

1 Introduction

LIKES (LInguistic and Knowledge Engineering Station) est une station d'ingénierie linguistique destinée à traiter des corpus, elle fonctionne pour l'instant sur la plupart des langues européennes : l'anglais et le français, le portugais, l'espagnol, l'allemand, le roumain et quelques langues slaves : le tchèque, le bulgare, le slovaque en utilisant des ressources minimales pour chaque langue. Les corpus sont constitués d'un ou plusieurs textes en ASCII ou en HTML, l'interface donne la possibilité de constituer son corpus et d'y exécuter un certain nombre de tâches allant de simples tâches de découpage en mot, de tri ou de recherche de motifs à des tâches plus complexes d'aide à la synthèse de grammaire, d'aide au repérage de relation, d'aide à la construction d'une terminologie. L'interface donne également la possibilité de constituer des filtres grammaticaux spécifiques à chaque utilisateur et/ou à chaque corpus qui sont utilisés dans les divers traitements de la station : découpage en phrases, lemmatisation et calcul de segments répétés.
Nous décrivons ici les principales fonctionnalités de LIKES et ce qui fait sa spécificité par rapport à d'autres environnements comparables. Likes possède un certain nombre de fonctionnalités supplémentaires utiles dans le domaine de la fouille de texte et la création d'ontologies que nous ne présenterons pas ici.
Likes bien que proche d'INTEX (Silberztein 1992) par certains aspects, adopte[1] une philosophie totalement différente vis à vis des ressources linguistiques[2]. En effet, LIKES vise à minimiser celles-ci et en conséquence n'utilise aucun dictionnaire. La parenté est indéniable en ce qui concerne les outils de recherches qui sont basés sur des automates et des transducteurs avec une syntaxe et un mode de fonctionnement très proche de celui d'INTEX.
Likes s'adresse à des utilisateurs linguistes ou terminologues aussi bien qu'informaticiens, étudiants aussi bien que chercheurs. Outre les tâches usuelles visées par le traitement de corpus : recherche de motifs, étude de distributions, Likes procure une aide à la construction de terminologie. Pour faciliter le travail du linguiste un principe permanent est adopté : le retour au contexte, toute entité de Likes, tout segment de texte est toujours visualisable dans son contexte. On peut élargir(zoom) le contexte et visualiser la phrase, puis le paragraphe, puis le texte où se trouve l'élément.
Pour faciliter le travail de l'informaticien et d'éventuels développements sur Likes, toutes les tâches sont exécutables à partir d'une console où on peut donner des commandes en ligne. Les exemples que nous fournirons ici seront plutôt tirés de corpus spécialisés comme le corpus MENELAS, car nous travaillons davantage sur les langues de spécialité que la langue

---

1 L'auteur qui a utilisé INTEX pour l'enseigner à des linguistes a pu constater l'intérêt de l'approche utilisée par INTEX pour les recherches de motifs : automates et transducteurs et de la syntaxe choisie. La partie recherche de la station a été conçue à un moment où INTEX n'était plus diffusée gratuitement et également à un moment où la librairie Java des expressions régulières n'existait pas encore.

2 Et également vis à vis de sa diffusion : c'est un programme écrit en Java qui tourne sur n'importe quel système, contrairement à INTEX (bientôt réécrit en C# et rebaptisé NOOJ) qui ne fonctionne que sur le système Windows. Likes est téléchargeable à l'adresse suivante : http://www-insa.u-strasbg.fr/liia/download avec une documentation en français (bientôt en anglais). Les sources et les API sont disponibles sur demande.Likes est utilisé depuis maintenant 3 ans par des chercheurs aussi bien linguistes qu'informaticiens qui travaillent sur différentes langues dont l'anglais, le français, le bulgare et le tchèque, il est également utilisé depuis 3 ans pour l'enseignement du TAL en sciences du langage et de la terminologie et des ontologies en Langues Etrangères Appliquées (LEA).

générale, toutefois, les outils présentés ici, peuvent évidemment être utilisés sur la langue générale.

2 Likes et le traitement de corpus

Les traitements possibles dans Likes vont des tâches élémentaires concernant les mots, les familles de mots, les mots composés, à celles plus complexes qui traitent des distributions et des relations.

2.1 Les tâches de base concernant les mots simples

Elles sont au nombre de quatre : constitution du corpus, découpage en mots et en phrases, tri alphabétique, tri par fréquence.
L'interface donne la possibilité de constituer un corpus. Dans un corpus donné, les textes doivent être de même nature, soit tous en texte (.txt), soit tous en XML, appartenant tous à une même langue et utilisant le même encodage[3]. L'interface demande à l'utilisateur la langue du corpus : cette information est ensuite utilisée par le programme pour déterminer l'encodage des textes.
Ces textes sont ensuite découpés. Les tokens sont séparés par les séparateurs habituels : signes de ponctuation faibles ou forts ou typographiques non alphanumériques. La langue du corpus détermine les ponctuations non coupantes : par exemple le tiret en français. Elle détermine également l'algorithme utilisé pour la segmentation en phrases du texte. Cet algorithme prend en compte les acronymes et un certain nombre de cas spécifiques au français (ces cas sont fournis dans un fichier du dossier des paramètres pour le français) : Pr. Dr. M. Mme J.C. J-C . etc.
Likes ne sépare pas les tokens des mots ainsi «124» est un token au même titre que «et», les tokens sont des formes, ainsi»Plus» est–il un token différent de «plus».
Le tri alphabétique prend en compte l'ordre lexicographique utilisé dans les dictionnaires :
Exemple «a<à<â<ä <A<A<Â< Ä» ou «e<é<è<ê<ë < E<É<è<Ê<Ë»[4]

2.2 Les tâches de base concernant des groupes de mots

Deux tâches de plus haut niveau concernent des entités formées de plusieurs mots : la fusion et le calcul des segments répétés.

2.2.1 La fusion

La fusion sert à ramener à une seule forme des familles de mots proches : l'usage spécifique est décidé par l'utilisateur. Elle peut servir à lemmatiser. L'utilisateur peut par exemple décider de regrouper les formes du pluriel et du singulier d'un même mot, il décide alors du représentant de cette nouvelle famille d'occurrences qui rassemblera bien évidemment les

3 ISO-Latin1, ISO-Latin2 etc.

4 Les données pour la fusion et le découpage en phrase seront bientôt fournis par des fichiers pour chaque langue.

occurrences des deux formes dans le corpus. Exemple : «médecin» et «médecins» sera représenté par «médecin_S».
On peut généraliser cette lemmatisation semi-automatique. Il suffit de spécifier que les mots qui existent avec un certain nombre de suffixes sont à fusionner. Ainsi «Xe et Xeront +Xeait+Xeons» calculera tous les ensembles de mots pour lesquels il existe des occurrences avec au moins deux de ces suffixes. Par exemple l'ensemble: «mange»,»mangeons», «mangeait». Chaque ensemble validé sera fusionné et le représentant de toutes ses occurrences sera par exemple : mange_V. L'utilisateur peut également, s'il le désire, rassembler deux synonymes, exemple «ECG» et «électrocardiogramme».
L'interface de fusion permet également de régler des problèmes de variations orthographiques. Par exemple, on peut collecter aisément tous les couples de mots qui ont même préfixe et même suffixe, mais ne varient que par la partie interne par exemple «c» dans l'un et «ç» dans l'autre. Le programme fournit alors une liste de couples possibles que l'utilisateur peut valider pour fusion. Exemple «facon» ou «façon». Les fusions permettent d'opérer une réduction du nombre des formes, elles vont être utilisées dans la suite, notamment avant le calcul des segments répétés.

2.2.2 Le calcul des segments répétés

La technique des segments répétés a été utilisée relativement fréquemment dans des approches orientées plutôt statistique d'abord pour le dépouillement d'enquêtes (Lebart, Salem, 1994), puis pour de l'aide à l'extraction de termes (Justeson, Katz, 1995). Nous l'avons implémentée (Frath & al.1995) en 1995 d'une façon très particulière, après avoir remarqué que la donnée de deux listes filtres distinctes permettait de découper des segments répétés correspondant à des termes composés ou plutôt à des «candidats termes» à valider. Notons que LEXTER (Bourigault, 1996) utilise des listes de mots semblables pour segmenter, mais sur un texte préalablement étiqueté.
Il s'agit de repérer des séquences de mots répétées dans le corpus. L'hypothèse est la suivante: si un polyterme[5] du domaine est intéressant, il sera utilisé plusieurs fois dans le corpus, sous exactement la même forme. En effet, dans les langues de spécialités, les termes sont très figés.
Le programme sélectionne des segments correspondant à des groupes nominaux qui sont susceptibles d'être des termes. C'est ainsi que la première liste filtre dite «filtre coupant» est construite avec des verbes et des adverbes, des pronoms relatifs et des conjonctions. Dans un souci d'économie ne figurent au départ dans cette liste que les formes les plus fréquentes de «être», «avoir», «devoir» et «pouvoir» (3^{ième} personne), l'utilisateur complétera cette liste au vu des premiers résultats de l'algorithme. Cette phase itérative lui permettra également de repérer les verbes les plus intéressants du domaine et leur emploi. On adjoint à ce filtre un certain nombre de mots qui prennent en compte le statut particulier du terme: il doit pouvoir représenter une dénomination décontextualisée, c'est ainsi qu'il ne peut donc contenir, ni adjectifs possessifs, ni démonstratifs, ni certains adjectifs de jugement: "mauvais","bon"... Un second filtre permet la mise en forme de finition: il supprime des mots aux bornes des séquences trouvées. Des résultats précédents, on supprime les articles en première place et de

---

[5] L'algorithme présuppose des termes composés, c'est à dire composés de mots distincts, ainsi il donne de nettement moins bons résultats pour les langues germaniques qui forment beaucoup de mots composés par concaténation.

même, en dernière place, les articles, les prépositions. Ainsi, le segment trouvé « l'artère gauche de » est ramené à la forme « artère gauche ».
Chaque langue est dotée au départ de deux filtres initiaux minimaux, le traitement des segments répétés s'opère interactivement, les filtres sont complétés éventuellement, après chaque examen des résultats.
Ce choix a été effectué afin que l'utilisateur contrôle au maximum ce que fait le programme et observe les caractéristiques du corpus traité. Par exemple, on ne peut y mettre la liste de tous les verbes, ceux-ci peuvent être ambigus, exemple "montre" verbe et "montre "nom. L'adjonction des verbes doit être toujours contrôlée.
Les résultats de ce traitement sont fournis sous forme d'arbre : une tête avec des extensions. Un clic permet un accès au contexte de chaque occurrence.
Un exemple de résultat sous forme d'arbre tiré du corpus MENELAS

```
taux  8
            d'expression  2
            de transcription 6
technique  14
                      de 10
                                  Southern 6
                                  Maxam et Gilbert  4
                     d'amplification  PCR  4
```

Les «têtes» (parties communes à plusieurs segments répétés comme ici «technique») permettent de repérer des termes simples. Le système fournit donc comme résultat supplémentaire, la liste des têtes et donne accès à toutes leurs occurrences.
La qualité finale des résultats dépend des compétences de l'utilisateur dans le domaine et l'on ne pourra en général s'abstenir de faire appel à un expert du domaine pour sélectionner la liste définitive des termes.
Efficacité: L'algorithme est linéaire en fonction de la taille du corpus, son fonctionnement est très voisin de l'algorithme SEQUITUR utilisé pour compresser des textes (Witten 2000). La méthode, quelle que soit son implémentation, est confrontée à un problème inévitable de repérage de redondances[6] qui implique une deuxième passe dans les résultats réduit son efficacité. Pour fixer les idées, les implémentations actuelles[7] permettent de travailler confortablement sur des corpus relativement moyens d'au maximum 5 M octets.(la gestion des contextes est en effet consommatrice en espace mémoire). Les corpus plus importants devront être segmentés manuellement et traités par parties.
Evaluation des résultats : L'évaluation de la qualité d'un extracteur de termes est difficile, car pour un humain décider ce qu'est un terme est une décision difficile et parfois subjective. Une précédente évaluation (Frath & al. 2000) montre que le bruit peut être très réduit : de 15% environ. Nous désignons par bruit le pourcentage de candidats termes rejetés par la validation humaine d'un expert (après la sélection faite par l'utilisateur) qui sont à rapprocher des taux de bruits d'autres méthodes (Bourrigault 1999)(cf. LEXTER qui oscille entre 60 et 80%). Notons aussi que les termes produits par Likes ne sont pas structurés : les dépendances qui lient les éléments d'un terme composé sont ignorées.
Le silence, lui aussi difficile à mesurer, n'a malheureusement pas été évalué. Par son principe même la méthode induit du silence, elle ne repère pas tous les termes simples, ni les

---

[6] Si on trouve 4 occurrences de « revascularisation de l'artère fémorale » et 4 occurrences de « artère fémorale », il s'agit forcément des mêmes occurrences et l'entrée artère fémoral est alors redondante.

[7] PC 512K de RAM Pentium 4 2,4 GHz avec la machine virtuelle Java 1.4.

polytermes non répétés, ni les variantes.[8] La qualité des résultats dépend évidemment de la qualité nécessaire à l'application visée, ainsi en indexation les résultats rapidement obtenus sont très satisfaisants, dans une application terminologique, ils doivent être complétés. Si on travaille dans un corpus incrémental où les situations de productions des nouveaux textes sont les mêmes, au bout d'un moment, les filtres se stabilisent et le programme peut fonctionner de manière totalement automatique.
Implémentation d'un nouvelle langue : Celle–ci est aisée, elle concerne la mise en place des deux filtres initiaux à partir des mots les plus fréquents de la langue (Catach, 1995), elle ne nécessite pas une très grande compétence linguistique.

2.3 Les tâches de recherche.

2.3.1 Recherches simples

Elles sont réalisées grâce à un outillage expressions régulières équivalent à des **automates**[9] (et transducteurs) récursifs comparables à ceux qu'on trouve dans la station INTEX et utilisant une syntaxe proche. Les entités des expressions sont des séquences de lettres et non forcément des mots[10] et on peut utiliser également des métacaractères comme <L> pour lettre[11]. On peut ainsi manipuler des schémas morpho-syntaxiques aisément.
Exemple: l'expression " <L><L>*(ion+ment) " recherchera dans le corpus tous les mots composés d'une lettre suivie de zéro ou plusieurs lettres suivies de «ion» ou de «ment».

2.3.2 Recherche de collocations

Un module spécifique permet de rechercher des cooccurrences et d'exprimer des contraintes de distances entre les mots choisis : on définit une recherche concernant deux mots distants d'au plus n mots, ou présents dans la même phrase, ou dans le même paragraphe etc.

2.4 Les tâches évoluées concernant les distributions

2.4.1 La synthèse de relations

Un certain nombre de programmes traitent de la synthèse de schémas morpho-syntaxiques exprimant des relations comme par exemple (Séguéla, Aussenac, 1999) ou (Morin 1999). Elles sont en général orientées vers le repérage de relations génériques comme l'hypo-hyperonymie et utilisent souvent des techniques statistiques. Nous avons choisi ici une autre

---

8 Des améliorations sont en cours d'implémentations, pour la récupération de termes recherchés à partir des têtes de segments répétés et le repérage des variantes.

9 Les expressions régulières sont traduites en automates qui sont déterminisés puis minimalisés.

10 INTEX a deux modes de fonctionnement distincts : soit les étiquettes d'arcs sont des lettres, soit ce sont des mots, mais on ne peut facilement écrire de schémas morpho syntaxiques.

11 Nous avons essayé d'être dans la mesure du possibles compatibles avec INTEX.

approche, orientée vers le repérage de relations spécifiques au domaine, et utilisant une technique basique. Celle-ci implique un contrôle et une réédition de l'utilisateur.
Le principe en est relativement simple : on repère deux mots donnés pour laquelle la relation hypothétique tient souvent. Comme par exemple dans MENELAS, on a remarqué la présence très fréquente de «coronarographie montre lésion», on donne «coronarographie» et «lésion» en entrée au module de synthèse. L'idée est de condenser toutes les façons d'exprimer la relation «montre». Le module de synthèse collecte tous les contextes[12] repérés entre deux occurrences de «coronarographie» et de «lésion», ces contextes sont ensuite comparés deux à deux. Chaque comparaison donne une chaîne de mots communs éventuellement vide, on sélectionne parmi toutes les chaînes obtenues celles qui ne sont pas constituées uniquement de mots du filtre non coupant. On élimine ainsi les chaînes constituées uniquement de mots grammaticaux.
L'interface du programme donne alors à l'utilisateur la possibilité d'éditer les schémas morpho-syntaxiques obtenus afin d'éliminer certains mots parasites : négations, adverbes, etc. et de construire ainsi un schéma synthétisant toutes les façons d'exprimer la relation.
Ce schéma est alors lancé sur le corpus tout entier pour rechercher les classes distributionnelles auxquelles appartiennent respectivement les deux mots.
sur l'exemple coronaire lésion, le programme rends

```
met en évidence des
montre des
objective des
a montré des
réalisée immédiatement montrait des
```

et après édition `(met en évidence+montre+objective+a montré+montrait)`

2.4.2 La recherche des classes distributionnelles

La recherche effectuée au moyen du motif précédent permet de récupérer un certain nombre de contextes où le sujet n'est pas coronarographie : comme «examen coronarographique», «échographie», «ECG», «électrocardiogramme», «épreuve d'effort» etc. et également où l'objet n'est plus lésion : «sténose», «resténose» etc. Cette distribution permet de formuler une hypothèse de classe sémantique: on peut en effet inférer l'existence de la classe des «examens médicaux». On en déduit que la relation examinée est bien la relation sémantique «montre» qui tient entre des «examens médicaux» et des «signes».

2.5 Les tâches d'analyses

Dans le traitement de corpus, le besoin de travailler sur des textes étiquetés totalement ou partiellement désambiguïsés se fait assez vite sentir. Aussi avons nous tenté d'apporter une réponse à ce besoin par différents modules. Nous avons donc conçu, d'une part, un module traditionnel de résolution d'ambiguïtés lexicales locales, construit dans un but pédagogique, capable d'étiqueter partiellement un texte, d'autre part, un segmenteur appelé encore chunker (Voutilainen, Tapanainen P., 1993) (Vergne, Giguet 1999). Ce dernier, reprends les idées défendues par Vergne sur l'inutilité d'un dictionnaire pour segmenter et sur l'intérêt de segmenter avant tout étiquetage voire avant toute analyse syntaxique. L'étiqueteur qui devrait lui faire logiquement suite est en cours de réalisation.

---

[12] La taille maximum des contextes considérés est paramétrable, par défaut elle est de 8 mots.

2.5.1 Application de dictionnaire et levée des ambiguïtés lexicales

Le premier qui renie quelque peu le principe de ressources minimales, est un module qui, d'une part, applique un dictionnaire au texte et qui, d'autre part, permet d'appliquer sur ce texte des transducteurs de désambiguïsations locales (encore appelées «grammaires locales» dans INTEX).
L'application d'un dictionnaire sur un texte doit en effet être possible, s'il y en a un de disponible. Les dictionnaires de Likessont gérés dans une base de données mySQL, un autre environnement appelé DICOMANAGER permet de gérer la création de dictionnaires et les flexions.
L'expressivité des transducteurs de levée d'ambiguïté est supérieure à celle des transducteurs d'INTEX, car il est possible en Likes d'exprimer par une étiquette des ensembles en intension complexes. On peut par exemple exprimer que un mot attendu doit être une entité du dictionnaire ambiguë entre Adjectif et Nom (A/N), là où INTEX oblige à énumérer une liste de mots en extension. Par contre la visualisation des résultats et uniquement textuelle, on ne peut visualiser les graphes résultats, ni les automates.

2.5.2 Le segmenteur (ou chunker) à faibles ressources

En attendant l'implémentation d'un étiqueteur, nous avons réalisé un segmenteur qui découpe une phrase en un ensemble de fragments élémentaires destiné à faciliter les traitements ultérieurs. Celui-ci se base, d'une part sur une liste fermée de mots grammaticaux pour déterminer le début des segments à isoler et d'autre part devine la catégorie de certains mots suivant leur morphologie pour détecter les fins de segments. Les unités segmentées sont : les SNR (syntagmes nominaux non récursifs) syntagmes nominaux sans syntagme prépositionnel ni relative, les SV, syntagmes verbaux contenant un verbe et éventuellement un ou des adverbes et des négations, les Pr pour propositions etc.
L'implémentation actuelle utilise les transducteurs de Likes, les items fournis en entrée des transducteurs qui sont appliqués en cascade (Roche, Shabes,1997) , insèrent dans le texte des balises qui sont utilisées par les transducteurs suivants. Les expressions des transducteurs qui segmentent le français ont été construites très rapidement en trois semaines et donnent des résultats très corrects sur MENELAS par exemple. Ils ont été également essayés sur des textes plus difficiles à traiter comme des poèmes, par exemple le Jaseroque [13] de Levis Carol voir l'extrait qui suit.

```
      Le jaseroque
<pn>Il brilgue µ: |<Ck plur>les toves <Adj>lubricilleux |<V plur>Se gyrent
|<Ck>en <V ant>vrillant |<Ck>dans <e>le guave µ, <N>Enmimés |<V plur>sont
µ|<Ck plur>les <Adj>gougebosqueux µ, |<Pr>Et |<Ck sing>le momerade
horsgrave.
```

<Adj> signale que le mot suivant est un adjectif, <pn> que le mot suivant est un pronom, <V plur > que le mot suivant est un verbe au pluriel, <Ck> signal le début d'un segment au pluriel, il s'agit du début d'un SNR au pluriel, <Ck> sans nombre indique le début d'un syntagme prépositionnel. µ et | indiquent une fin de segment.
L'implémentation de ce module s'avère extrêmement intéressante, car elle permet de bien voir l'intégralité des règles du segmenteur et éventuellement des adaptions à un usage particulier. Cette technique pose toutefois deux problèmes : d'une part, celui de la lisibilité des règles, par

---

[13] Traduction française du Jaberwock poème de Levis Caroll dans Alice au pays des merveilles.

des informaticiens et a fortiori par des linguistes, d'autre part celui de leur validité. C'est pourquoi nous travaillons à l'élaboration d'une méthode pour construire ces règles de façon objective à partir d'un dictionnaire et non plus par introspection. Nous menons également une réflexion sur une formulation déclarative et donc plus lisible des règles.

2.6 La construction d'une terminologie

Nous nous plaçons ici dans une perspective où la construction d'une terminologie est vue comme un problème linguistique et non comme un problème de compilation de termes étiquettes de concept comme dans une école terminologique connue (Kleiber 2003). Les outils présentés ici peuvent être utilisés soit pour faire une terminologie monolingue, dans un domaine spécialisé, en prélude éventuel à la construction d'une ontologie, soit pour construire une terminologie multilingue.

Nous ne détaillerons pas ici les fonctionnalités d'aide à la construction terminologique, nous en resterons aux principes. Notre implémentation fait une séparation claire entre le linguistique et le conceptuel. En effet, nous proposons à l'utilisateur une gestion de fiches terminologiques permettant de prendre en compte les différentes étapes de construction de la terminologie. On part d'une liste de candidats termes éventuellement fournis par les modules de Likes, pour les étudier et leur donner éventuellement le statut de terme.

La création d'une fiche terminologique s'effectue après sélection par l'utilisateur d'une famille de mots formée de formes différentes et de synonymes d'un même terme, par exemple «ECG» et «électroencéphalogramme». La fiche correspondante permet d'accéder à tous les contextes des occurrences d'un terme (et de ses synonymes) dans le corpus : un éditeur permet de ne sélectionner que les occurrences représentatives d'un terme si désiré. Chaque terme peut être attaché à un concept, grâce à une fiche qui permet de le définir et de l'implanter dans la hiérarchie conceptuelle, par la donnée de ses frères et de son père, cette fiche donne accès aux termes qui le désignent dans plusieurs langues éventuellement. Dans les applications multilingues, les langues sont séparées (à chaque langue correspond un corpus et un ensemble de termes) et ne sont liées que grâce aux concepts.

3 Conclusion

Nous avons présenté ici un atelier de développement d'outils de traitement de corpus qui permet de réaliser un certain nombre de tâches utiles avec peu de ressources linguistiques. Les situations sont nombreuses où les ressources linguistiques adaptées ne sont pas disponibles et sont trop coûteuses à construire. Ces situations concernent certaines langues pour lesquelles les ressources sont faibles et bien évidemment certains langages de spécialité. Likes permet rapidement de donner des résultats qui bien qu'imparfaits sont significatifs et utiles. Les outils de Likes sont très utiles tant pour les traitements de corpus que pour la formation à certains concepts de TAL et à l'extraction d'informations à partir de textes.

Pour l'instant Likes ne permet de ne traiter que des textes non formatés ou en HTML. Des développements sont en cours pour permettre le traitement des textes en RTF et en PDF.

Likes n'est pas voué à une théorie particulière et procure des outils qui peuvent être adaptés à des tâches diverses. On peut par exemple utiliser très facilement les transducteurs pour ôter les balises des fichiers (en XML ou en HTML). On peut bien sûr y utiliser des ressources linguistiques si on le désire.

Nous continuons actuellement de développer des modules dans le même esprit, chaque module doit pouvoir être utilisé pour des buts éventuellement différents, ses fonctionnalités doivent être simples à comprendre et être utiles en TAL.

Mentionnons à titre d'exemple, le module d'annotation que nous n'avons pas décrit ici faute de place qui permet d'annoter manuellement des textes (avec des balises XML) et ensuite d'analyser les segments de textes annotés de la même manière pour en synthétiser des automates. En effet, une de nos applications consiste à annoter des textes de biologie, des articles scientifiques sur la structure des protéines pour faciliter l'accès à leur contenu par les chercheurs en biologie. Nous avons donc commencé par leur fournir un ban d'annotation manuelle. Le module de synthèse de contexte existant sera affiné pour permettre l'aide à la synthèse d'expressions régulières exprimant les régularités de ces contextes.

Références

Bourigault D. (1996), LEXTER a natural language processing tool for terminology extraction. In Proceedings of the *7th EURALEX International Congress,* Goteborg .

Catach N. (1995), *L'orthographe française*. Nathan.

Frath P., Oueslati F., Rousselot F. (1995), Identification de relations sémantiques par repérage et analyse de cooccurrences de signes linguistiques. Actes de *JAVA* 1995 (Grenoble) réédité (2000) dans *Ingénierie des Connaissances, évolutions récentes et nouveaux défis*. Eyrolles.

Justeson J., Katz S. (1995), Technical terminology: some linguistic properties and W. an algorithm for identification in text. *Natural Language Engineering,* Vol 1(1), pp.9-28.

Morin E. (1999), Acquisition de patrons lexico-syntaxiques caractéristiques d'une relation sémantique. *TAL (Traitement Automatique des Langues)* Vol.(1) pp 143-166, Paris .

Kleiber G. Exposé introductif au Colloque TIA 2003 Strasbourg.

Roche E., Schabes Y., (1997), *Finite state language Processing*, MIT Press.

Rousselot F., Frath P. , Oueslati R. (1996) Extracting concepts and relations from corpora . Proceedings of *Workshop on Corpus-oriented Semantic Analysis European Conference on Artificial Intelligence ECAI 96,* Budapest.

Séguéla P., Aussenac_Gilles N. (1999), Extraction de relations sémantiques entre termes et enrichissement de modèles du domaine. Actes de *IC'99 (Ingénierie des Connaissances),* 79-88, Paris.

Silberztein M. (1993) *Dictionnaires électroniques et analyse automatique des textes.* Paris. Masson.

Vergne J. et Giguet E. (1998), Regards Théoriques sur le "Tagging". Actes de *TALN98*. Nancy.

Voutilainen A ., Tapanainen P.(1993) Ambiguity resolution in a reductionistic parser. Proceedings of the *Sixth Conference of the European Chapter of the ACL*, Utrecht.

Witten I.H. (2000), Adaptative Text Mining: Infering Structure from Sequences. *Journal of Discrete Algorithms* Vol 0 (0), pp.1-23.