Étude de la segmentation de documents et première version de SegDoc liée à SECTra-v3

Projet Traouiéro, document L3.4.a

Ce document est le premier rapport d'avancement partiel de la sous-tâche 3.4 (il s'agit de sa phase a) et est joint au rapport d'avancement des tâches T2, T3 et T4 (L234.3). Il concerne le module de segmentation et normalisation de documents en cours de construction dans le cadre du projet Traouiero.

Contenu

Étude de la segmentation de documents et première version de SegDoc liée à SECTra-v3

Achille FALAISE, Ruslan KALITVIANSKI

RÉSUMÉ

Ce document est le premier rapport d'avancement partiel de la sous-tâche 3.4. Il est joint au rapport d'avancement L234.3 des tâches 2 à 4 à T0+18 (15/7/12). Il concerne le module de segmentation et normalisation de documents en cours de construction dans le cadre du projet Traouiero.

INTRODUCTION

1.1 CONTEXTE

La sous-tâche ST3.4 participe à la tâche 3 (IMAG++). Son but est de construire un segmenteur-normaliseur "propriétaire", multiple et récursif, meilleur que ceux qui existent, et nous garantissant l'autonomie. Pour l'instant, c'est toujours GoogleTranslate que nous utilisons indirectement comme segmenteur dans les plates-formes iMAG.

Nous rappelons d'abord la description de cette tâche dans la proposition. Dans la section suivante, Ruslan KALITVIANSKI présente une synthèse de l'étude théorique et exploratoire qu'il a menée, en partie en août 2011, et en partie dans le cadre de son M2R. Dans la section suivante, Achille FALAISE présente SegDoc-1, une première version simplifiée qu'il a réalisée et expérimentée dans le cadre de son implémentation de SECTra-v3.

1.2 OBJET DE LA SOUS-TÂCHE ST3.4 : SEGDOC

T3 : iMAG++ : Outils pour la multilinguisation collaborative de pages Web et de documents	
ST3.4 : t6-t18 — SegDoc: segmenteur-normaliseur	
Objectifs	Construire un segmenteur-normaliseur "propriétaire", multiple et récursif, meilleur que ceux qui existent, et nous garantissant l'autonomie
Critères d'évaluation	Conformité avec les spécifications et avec les jeux de test
Responsable	GETALP
Partenaires	GETALP, Floralis (mise au courant pour valorisation)
Activité	Quand une page Web ou un document est soumis à un système de TA, il faut d'abord en extraire les *segments textuels* (phrases ou titres), qu'il faudra traduire, et ensuite *normaliser* ces segments selon les systèmes auxquels on les soumet. La plupart des systèmes de TA adoptent le segment comme unité de traduction, s'interdisant ainsi de pouvoir rechercher les antécédents de pronoms dans les segments précédant ou suivant (cas de cataphore) le segment en cours de traduction, de gérer la concordance des temps en français, etc. Certains, comme ceux écrits en Ariane-G5, peuvent avoir des unités de traduction de l'ordre d'une ou deux pages, et Ariane-Y est construit pour, à l'instar de XIP de Xerox (C. Roux, XRCE), ou de SYGMART de J. Chauché, pouvoir traiter tout un document Xml (même de 200 pages) comme une seule unité de traduction. D'autres, comme la plupart des systèmes commerciaux, ne peuvent pas traiter des phrases de plus de 30 à 40 mots, et doivent donc utiliser des *infrasegments* comme unités de traduction, par exemple les fragments correspondant à des *puces* dans des listes à puces, quand ils rencontrent des phrases plus longues. ▪ Si l'on veut pouvoir appeler tous les systèmes de TA disponibles, il faut donc construire un segmenteur pouvant produire une *segmentation multiple,* sous forme d'un *graphe de segmentation.*

	▪ Comme il y a parfois "du texte dans du texte", par exemple plusieurs paragraphes comme valeur d'un attribut TITRE dans une ancre en Html, il faut permettre la *segmentation récursive,* et donc trouver un moyen de représenter cela dans un graphe de segmentation. ▪ Comme ce problème est particulièrement difficile, en particulier parce que les textes réels contiennent des balises, du code (css, javascript, etc., le rtf et le LaTeX étant particulièrement difficiles à traiter), les segmenteurs actuels font énormément d'erreurs : il faut donc constuire un *éditeur de segmentations* (utilisable en local et sous un navigateur Web). D'autre part, les textes à traduire sont toujours prétraités, ou *normalisés.* Par exemple, les systèmes de TA probabiliste comme Google Translate et ceux écrits avec Pharaoh, Moses ou Joshua ôtent la casse (tout est en minuscules) et *tokénisent,* i.e. introduisent des blancs quand il y a élision (l'ail →l' ail, Il vient. → il vient .). Ils remplacent aussi les balises de présentation (comme …) et les liens ou ancres (<a>…), ainsi que les équations, marques de fabrique, etc., par des *occurrences spéciales,* et cela de façon différente pour chaque système. Par exemple, … sera normalisé par le préprocesseur MosesWeb de Moses en MOSESOPENTAG23 … MOSESCLOSETAG23. Enfin, il y a souvent plusieurs étapes de normalisation, la première consistant à passer en UFT-8 et à remplacer les entités (comme é) par leurs valeurs (é). Au total, il faut donc : ▪ définir un formalisme permettant de représenter le flot d'octets initial, sa forme normalisée de base, son graphe de segmentation (avec l'alignement de chaque segment sur la forme de base), les différentes normalisations, et les correspondances (minidictionnaires) entre les occurrences spéciales et leurs valeurs. ▪ définir et implémenter un formalisme (un petit LSPL) permettant d'écrire des règles de segmentation, de les appliquer à un flot de caractères, et d'affecter des poids aux résultats. Une bonne nouvelle est que le processus de segmentation peut souvent s'appuyer sur des *fichiers compagnons* (par exemple, les fichiers UNL associés aux fichiers Aspx/Html contenant les articles de l'encyclopédie EOLSS), ou sur les mémoires de traductions (par exemple, dans le cas d'une iMAG-S dédiée à un site S). Ainsi, si l'on traite une page Web dynamique, on peut la segmenter en recherchant d'abord si elle contient des sous-chaînes qui coincident avec certains segments stockés dans la MT, puis en partant de ces "îlots de confiance" pour segmenter le reste.
Livrables	▪ Sources des programmes et documentation (sur la forge du LIG). ▪ Site d'essai (et de comparaison avec Ariane-G5).
Gestion du risque	Cette sous-tâche est la seule qui comporte un aspect recherche. Mais, même si l'on n'arrive pas à tout réaliser de façon efficace, on est sûr de produire un segmenteur bien meilleur que les outils existants, que nous connaissons et dont, pour certains, nous avons étudié le code.

2. ÉTUDE THÉORIQUE, ÉVALUATION D'OUTILS, OUTIL PROTOTYPE

2.1 EXEMPLES, PRÉSENTATION INTUITIVE, ET DÉFINITIONS

Un segment est l'unité de traduction de base de traducteurs humains. Il s'agit d'une phrase, d'un titre ou d'un terme dans une nomenclature. On définit un infrasegment comme une portion d'un segment et un supersegment comme un morceau de texte qui contient plusieurs segments [1].

Un document est constitué de texte, éventuellement de code, et d'objets satellites (images, formulaires, etc.). Mis à part le texte immédiatement visible au lecteur, un document peut contenir des sous-textes, qui peuvent, par exemple, être des attributs textuels d'une balise. Ces sous-documents peuvent eux-mêmes être constitués de plusieurs segments et avoir des sous-documents.

Exemple :

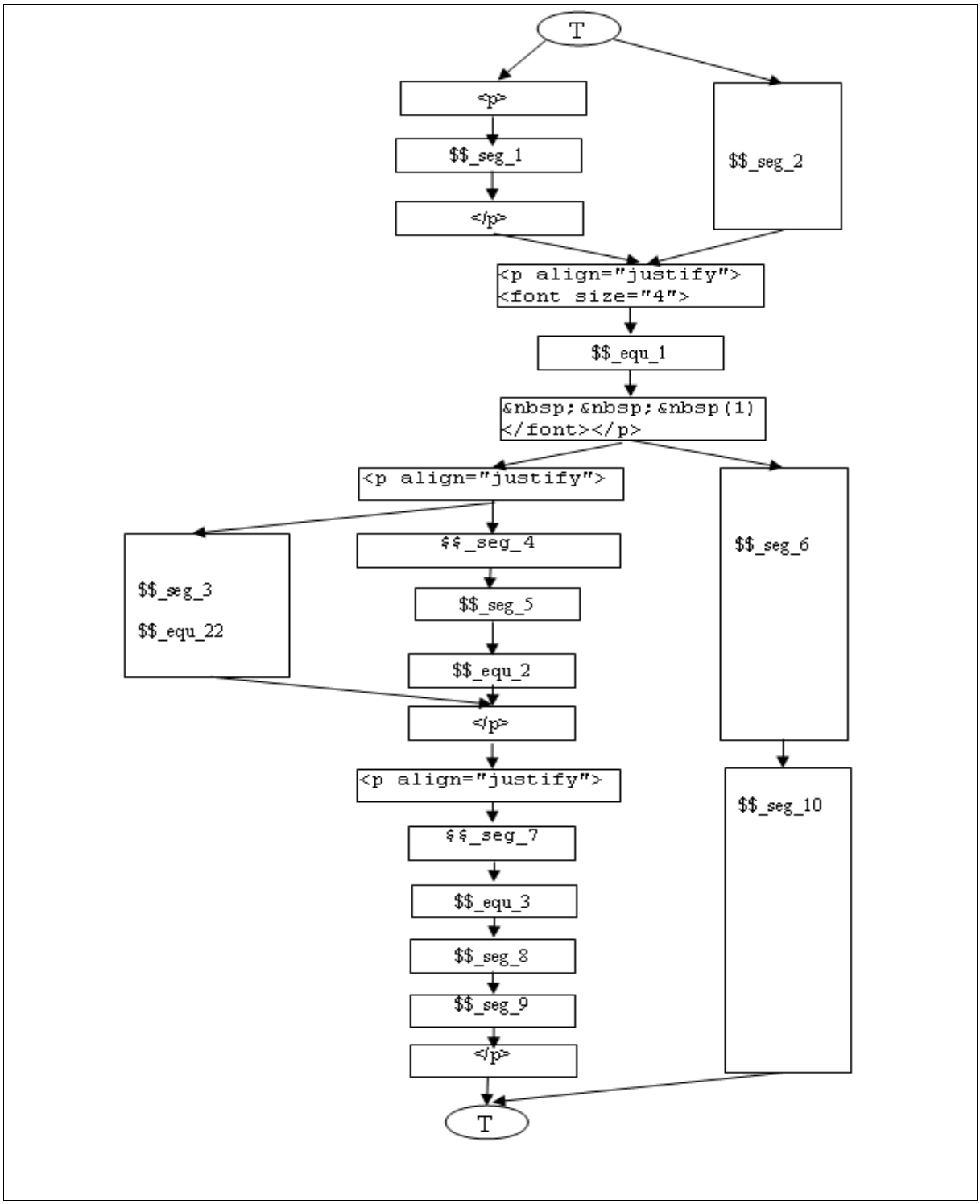
```
Une phrase courte avec au milieu une ancre
    <a href="ttot.fr" title="Un long texte, avec <em>éventuellement</em>
    des balises et du sous-sous-texte comme une note en bas de page, peut
    être contenu dans un attribut TITLE d'une ancre.">
peut indirectement contenir un très long texte.
```

Les documents peuvent contenir des éléments non linguistiques, mais qui peuvent avoir des rôles linguistiques, comme les figures ou les expressions (sortes de substantifs) et les relations (sortes de noyaux verbaux ou de substantifs) mathématiques, les marques de fabrique, des balises de formatage « douces » comme gras, italique, souligné, espacé, les débuts et fins de marques d'index, les champs de citation EndNote, etc. Avant de pouvoir soumettre un texte à des systèmes de TA, il faut traiter ces « hors-texte » - les remplacer par des occurrences spéciales munies de leur catégories grammaticales possibles et les répertorier dans un dictionnaire local d'occurrences.

Les systèmes de TA diffèrent par le format de données qu'ils peuvent traiter. Systran peut traiter des documents html, et on a donc intérêt à garder les balises html dans le texte qu'on lui soumet. D'autres systèmes, comme ProMT ou Reverso ne peuvent traiter que du texte brut. Par conséquent, il est nécessaire de normaliser les segments pour tel ou tel système de TA.

Il faut pouvoir produire des segmentations de granularités différentes. D'un côté, on cherche à produire des corpus parallèles alignés au niveau de segments les plus courts possibles « faisant sens », et donc en pratique au niveau des phrases ou des groupes nominaux indépendants. D'un autre côté les systèmes de TA diffèrent par la taille de l'entrée (certains ne prennent qu'une phrase, d'autres peuvent prendre plusieurs pages), et il faut adapter la taille des unités de traduction aux exigences de chaque système, en visant à lui fournir le plus long texte possible, afin qu'ils puissent traiter la concordance des temps ou les anaphores et cataphores. Cela induit une segmentation différente selon les systèmes visés. Nous appelons graphe de segmentation un graphe qui représente les segmentations multiples d'un document. L'exemple ci-dessous montre un texte html et un graphe de segmentation possible.

```
<p> The wave propagates from the source with a velocity of long gravity water
waves in accordance with the equation </p>
<p align="justify"><font size="4">
C<sub>G</sub>=(g H)<sup>1/2</sup>,   (1)</font></p>
<p align="justify">where g is the acceleration due to gravity, and H is the depth
of the basin. Because the average depth of the world ocean is 4 km, the typical
velocity of tsunami in the ocean is 200 m s<sup>-1</sup> or 720 km h<sup>-
1</sup>.</p>
<p align="justify">Such a wave, propagating with the velocity of an airplane,may
traverse the Pacific ocean in 10-12 hours and bring down a wall of water 10m high
with a velocity of more than 70 km h<sup>-1</sup> upon a calm ocean beach. The
wave velocity is decreased near the coastline because of shallower water and the
slowing of the wave by the roughness of the bottom.</p>
```



***$$_seg_1**: The wave propagates from the source with a velocity of long gravity water waves in accordance with the equation*

***$$_equ_1**: $C_G = (g\ H)^{1/2}$*

***$$_seg_4** : where g is the acceleration due to gravity, and H is the depth of the basin.*

***$$_seg_5** : Because the average depth of the world ocean is 4 km, the typical velocity of tsunami in the ocean is*

***$$_equ_2** : 200 m s^{-1} or 720 km h^{-1}.*

***$$_seg_7:** Such a wave, propagating with the velocity of an airplane, may traverse the Pacific ocean in 10-12 hours and bring down a wall of water 10 m high with a velocity of more than*

***$$_equ_3**: 70 km h^{-1}*

***$$_seg_8** : upon a calm ocean beach.*

***$$_seg_9**: The wave velocity is decreased near the coastline because of shallower water and the slowing of the wave by the roughness of the bottom.*

***$$_seg_2 :** <p> $$_seg_1 </p>*

***$$_seg_6:** <p align="justify"> $$_seg_4+$$_seg_5+ $$_equ_2 </p>*

***$$_seg_10 :** <p align="justify"> $$_seg_7+ $$_equ_3+$$_seg_8+ $$_seg_9 </p>*

***$$_seg_3 :** $$_seg_4+$$_seg_5*

Avec des hors-textes définis comme suit :

***$$_equ_1**: C_G=(g H)^{1/2}*

***$$_equ_2** : 200 m s⁻¹ or 720 km h⁻¹*

***$$_equ_3**: 70 km h⁻¹*

On peut voir une segmentation possible comme une trajectoire dans un graphe de segmentations. Nous pouvons représenter un graphe de segmentation récursif par une liste de graphes de segmentation non récursifs, chaque graphe correspondant à un document sans ses sous-documents.

2.2 DIFFICULTÉS ET APPROCHES POSSIBLES

Ici nous décrivons les problèmes de traitement et de représentation liés à la segmentation multiple et récursive, et proposons des solutions à certains d'entre eux.

2.2.1 SEGMENTATION EN PHRASES

Parmi les segmentations que nous devons produire, il nous faut certainement la segmentation en phrases. C'est un problème difficile du TAL. Pour les langues européennes l'utilisation des ponctuations donne en général une assez bonne approximation, mais certaines des ponctuations sont ambiguës, comme le point-virgule et le deux points – ils peuvent être séparateurs de phrases ou de parties de phrase. Le point est particulièrement ambigu, car il peut représenter une décimale dans un nombre, un séparateur dans une date, une fin de phrase, une abréviation ou les deux simultanément. Ce dernier cas est problématique, car certains types de textes, comme les textes scientifiques, peuvent avoir plus de points d'abréviation que de points de fin de phrase. Il est possible de reconnaître les non abréviations avec des heuristiques portant sur la longueur du mot et en ayant une liste de mots qui ne peuvent jamais se trouver en fin de phrase (ex. M.). Il est nécessaire de construire une telle liste pour chaque langue.

2.2.1.1 Approche par règles

Pour la segmentation en phrases des langues où la séparation est indiquée par des ponctuations l'approche la plus couramment utilisée est l'approche par règles. Le format SRX (Segmentation Rules eXchange), crée par la Localization Industry Standards Association (LISA), définit une méthode standard de description de règles de segmentation échangées entre utilisateurs.

Un fichier SRX est un fichier XML qui contient une liste ordonnée de règles sous forme d'expressions régulières. Le fichier est divisé en deux parties. La première, représentée par l'élément <languagerules> spécifie les règles de segmentation. La seconde, représentée par <maprules> spécifie les langues auxquelles une règle s'applique. Il y a deux types de règles, celles qui définissent les endroits de rupture (break) et celles qui définissent les exceptions. Chaque règle contient zéro ou un élément <beforebreak> et zéro ou un élément

<afterbreak> qui définissent les expressions régulières qui doivent correspondre au texte avant une position de rupture et après la position de rupture de la règle à appliquer. La valeur « yes » ou « no » de l'attribut <break> indique si cette expression définit une fin de phrase ou une exception. L'ordre des règles est important, elles sont appliquées dans l'ordre de leur définition.

Le standard SRX permet de choisir si les balises de formatage du format initial doivent être inclues dans le segment. On peut indiquer au segmenteur SRX s'il doit aussi segmenter les sous-documents à l'aide de l'attribut «segmentsubflows».

Ce format est utilisé par de nombreux systèmes de TAO, dont OmegaT, Okapi et MemoQ. Il existe des règles en SRX pour plus d'une dizaine de langues, dont l'anglais, le japonais et le russe. Cependant, il est impossible d'en créer pour une langue comme le thaï, qui n'a pas de séparateurs entre mots ou phrases. Autre limitation des SRX, les approches de surface comme les expressions régulières ne peuvent pas bien traiter la récursivité, comme les phrases imbriquées, par exemple une phrase qui contient une citation contenant plusieurs phrases séparées par des poins.

2.2.1.2 Approche probabiliste :

Il est possible d'utiliser une approche probabiliste, mais cela nécessite d'avoir un corpus d'entrainement segmenté manuellement de manière consistante, ce qui est un travail à renouveler pour chaque nouvelle langue. Cette approche donne de bons résultats sur du texte brut, mais ne fonctionnera pas sur des données balisées. De plus, il n'y a pas de standard d'échange de données produites par l'apprentissage automatique.

2.2.1.3 Fichiers-compagnons

Certains documents sont munis de fichiers-compagnons, que l'on peut utiliser pour guider la segmentation, comme les fichiers .unl associées à certains documents de l'encyclopédie EOLSS. La mémoire de traductions peut également être utilisée, surtout si la page, ou une page similaire, a déjà été traduite.

2.2.2 Reconstruction de phrases :

En faisant appel au segmenteur de Google ou en appliquant des règles on obtient souvent des phrases coupées par la présence d'un hors-texte, par exemple une image ou une formule à l'intérieur de la phrase. Un autre cas est celui des les listes à puces, qui peuvent être précédées et suivies par un début et une fin de phrase. Les segmenteurs actuels segmentent chaque élément des listes à puces comme des segments à part entière, or souvent ce n'est pas le cas. Une telle construction peut représenter une phrase complexe, que l'on devrait traiter en entier. Il est nécessaire de trouver une méthode pour identifier ces cas.

2.2.3 Segmentations multiples :

Si l'on a déjà obtenu une segmentation, par exemple en phrases, on doit pouvoir réutiliser les segments obtenus dans la définition de plus gros segments. Si l'on a utilisé plusieurs segmenteurs, on veut pouvoir grouper ces segmentations dans un graphe, mais aussi pouvoir calculer des intersections de segmentations. Un segmenteur multiple doit pouvoir prendre en entrée des fichiers décrivant des règles de segmentation multiple et des instructions pour le traitement des hors-texte. La description d'une segmentation multiple doit consister en un graphe de segmentations, un dictionnaire de segments (contenant le nom, l'origine, la longueur et la valeur normalisée), un dictionnaire des hors-textes (de forme : occurence-valeur, ou adresse si le hors-texte est non textuel).

2.2.4 Segmentation récursive :

La récursivité peut apparaître à plusieurs niveaux dans un document. Outre des attributs de balises html tels que « alt » ou « title » qui peuvent contenir des sous-documents, on peut

avoir de la récursion dans les phrases-mêmes, par exemple dans des phrases contenant une citation. Les citations peuvent d'ailleurs être dans une langue différente de celle de la phrase, et dans ce cas on peut vouloir ne pas les traduire. (Systran permet de définir des zones à ne pas traduire dans les textes en entrée). Un autre exemple difficile de récursion est représenté par les listes à puce, qui peuvent elles-même contenir des sous-listes à puce. Il faut trouver une représentation de la récursion dans un graphe de segmentation.

2.2.5 NORMALISATION POUR LES APPELS À LA TA :

Les documents .html peuvent contenir des zones en différentes langues et différents codages. Puisque les systèmes de TA diffèrent par les formats et les codages attendus en entrée, avant d'appeler un système de TA, il faut identifier ces zones. Nous pouvons le faire grâce à l'outil SANDOH [2].

2.2.5.1 Balises de formatage

Un premier exemple de traitement de balises dans les segments soumis à la TA est celui de MosesWeb. Moses est un système de TA probabiliste, qui traduit du texte brut phrase par phrase. MosesWeb [3] est un ensemble de scripts Perl qui permettent d'utiliser Moses pour traduire des pages Web, en en extrayant toutes les phrases, en leur ôtant et mémorisant tout balisage, en les soumettant à Moses, puis en les réinsérant dans la page Web de départ. Pour ce faire, MosesWeb remplace toutes les balises ouvrantes et fermantes par ses balises MOSESOPENTAGi et MOSESCLOSETAGi, en gardant dans sa mémoire le nom de la balise html et la liste de ses attributs. L'étude de ce logiciel a fait partie du stage de Nahla Laribi [4], qui a eu lieu au GETALP du 01.04.09 au 30.09.09.

2.2.5.2 Traiter les éléments non textuels

Pour traiter les éléments non textuels, on devra les transformer en occurrences spéciales munies de leur catégories grammaticales possibles. Pour cela, on substituera les hors-texte par des représentations normalisées (ex. *$$_expr_math_12* pour une expression mathématique), les hors-texte étant stockés dans un dictionnaire d'occurrences spéciales. L'une des difficultés de cette approche est l'identification correcte des catégories grammaticales des occurrences.

2.3 ÉVALUATION D'OUTILS DÉCRITS DANS LA LITTÉRATURE OU EXPÉRIMENTÉS

Plusieurs outils de TAL existent qui permettent de segmenter des documents. Nous en faisons ici une brève revue en considérant en particulier deux aspects : le format d'entrée, la possibilité de fournir une segmentation multiple et/ou récursive.

2.3.1 GOOGLETRANSLATE

Google Translate se décline en deux outils de traduction : le service Web disponible à tous (http://translate.google.com/), et Google Translation Toolkit, nécessitant de s'enregistrer.

Le GETALP utilise actuellement le service Web. GoogleTranslate permet de traduire et post-éditer des pages html, mais aussi des documents en d'autres formats textuels, tels que .doc ou .odt, qui sont convertis en html. Le texte à traduire est segmenté en phrases et la page résultat contient à la fois les segments source et les traductions. GoogleTranslate ne produit pas de segmentation multiple. Les sous-documents sont traduits, mais seul le Translation Toolkit permet de les éditer (mais il est possible de les récupérer de la page traduite).

2.3.2 OMEGAT

OmegaT (http://www.omegat.org/fr/omegat.html) est un logiciel de TAO libre et multiplate-forme qui permet de traduire des documents dans un grand nombre de formats balisés et bruts (html, doc, tex, txt, etc.). Le segmenteur de OmegaT applique des règles en format SRX pour segmenter le texte en phrases. L'utilisateur traduit ensuite ces segments, automatiquement, manuellement ou par post-édition, avec possibilité d'importer une

mémoire de traductions en format TMX, et construit le fichier traduit. OmegaT n'utilise pas la mémoire de traductions pour segmenter, ne permet pas d'exporter les segments dans un format réutilisable, ne peut fournir ni segmentation multiple, ni édition de segments, mais segmente les sous-documents.

2.3.3 LE SEGMENTEUR DE MOSESWEB

Moses est un système de TA probabiliste, qui traduit du texte brut phrase par phrase. MosesWeb (http://www.statmt.org/moses/?n=Moses.WebTranslation) est un ensemble de scripts Perl qui permettent d'utiliser Moses pour traduire des pages Web. Pour l'analyse, MosesWeb utilise les concepts de balises molles et dures : les balises molles sont tolérées à l'intérieur d'une phrase, les balises dures sont des « casseurs » de phrases, qui segmentent les documents en phrases indépendantes. Un script Perl parcourt la page, identifie les zones textuelles (qui contiennent éventuellement des balises molles), et les stocke dans sa table de segments. Les balises contenues dans ces parties textuelles sont remplacées par des balises MOSESOPENTAGi et MOSESCLOSETAGi, (en mémorisant le nom et les attributs des balises html originales). Ainsi, la page est segmentée en zones textuelles (pas encore segmentées en phrases) et zones de code. Les segments textuels sont « *tokenisés* » (introduction de blancs entre les éléments (ex : l'ail → l' ail, il vient → il vient .) et segmentés en phrases (en comparant les *tokens* avec les symboles définis comme délimiteurs de phrases (ponctuations, parenthèses, etc.)). Les balises Moses sont ôtées des phrases obtenues, qui sont soumises à Moses pour la traduction. Puis, les balises sont réintroduites dans les phrases traduites, et la page traduite est construite. MosesWeb ne fait pas de segmentation multiple, mais segmente les sous-documents.

2.3.4 LINGPIPE

LingPipe (http://alias-i.com/lingpipe/) est un ensemble d'outils de TALN développé en Java. Son interface Web permet entre autres de segmenter un texte brut, ou en html ou xml en phrases en utilisant des modèles heuristiques de langues (anglais pour l'interface Web). L'outil permet de choisir le codage d'entrée et de sortie, et produit un fichier xml contenant les segments. Bien qu'intéressant pour des textes médicaux ou des articles d'actualité (pour lesquels ces segmenteurs ont été adaptés), l'outil ne permet ni l'édition de segmentation, ni segmentation multiple, ni récursive.

2.3.5 RSTTOOL

RSTTool de WagSoft (http://www.wagsoft.com/RSTTool/) est un outil d'étiquetage de la structure d'un texte pour Windows (les versions anciennes existent pour Mac et Unix/Linux). Il permet de segmenter du texte brut manuellement ou d'appliquer quelques règles simples de segmentation, basés sur les ponctuations. La segmentation automatique peut se faire soit en phrases, soit en paragraphes. Le fichier obtenu en sortie est un fichier texte dans lequel sont insérées des balises et . Bien que ne fournissant ni segmentation multiple paramétrable ni segmentation récursive, un tel outil est un exemple intéressant d'éditeur de segmentation.

2.3.6 ANALYZEASSIST

AnalyzeAssist (http://ginstrom.com/AnalyzeAssist/) est un outil pour Windows qui sert à compter le nombre de segments d'un document qui sont présents dans une mémoire de traduction. Bien qu'il utilise le format TMX pour l'import de mémoires de traductions, il ne permet pas d'importer des règles de segmentation en format SRX, qui lui est complémentaire. L'outil traite des documents Word, Excel, PowerPoint, ainsi que du html, xml et du texte brut. Il utilise des règles de segmentation en phrases très basiques (ponctuations). Il ne permet ni la multiplicité ni la récursivité de segmentations.SegDoc-1, une première version simplifiée

2.4 INTRODUCTION

SegDoc est le composant de segmentation de base de SECTra-v3. Son développement est progressif, commençant par les fonctionnalités essentielles et les étendant par la suite. On distingue deux niveaux dans le choix des fonctionnalités de base :

1. SegDoc 1.0 : les fonctionnalités indispensables pour un scénario de traitement de traduction de documents HTML dans SECTra-v3. Ces fonctionnalités garantissent *a minima* le bon fonctionnement de SECTra-v3.
2. SegDoc 1.1 : les fonctionnalités de SegDoc 1.0, plus celles déjà présentes dans l'ancien SECTra-v1. Ici, on vérifie l'absence de régression par rapport à SECTra-v1.

Essentiellement, SegDoc 1.0 propose une segmentation grossière en paragraphes, qui est utilisable mais n'est pas toujours optimale. SegDoc 1.1 affine cette segmentation en séparant les phrases, ce qui est plus pratique pour l'utilisateur et permet à SECTra d'être plus fin dans la recherche de traductions déjà existantes.

2.5 CAHIER DES CHARGES

Au niveau du cahier des charges, les versions 1.0 et 1.1 de SegDoc ne se distinguent que par le grain de la segmentation (paragraphe *ou* phrase).

Ces versions de SegDoc sont indépendantes de SANDOH (composant de détection de la langue et du codage) ; on suppose en effet que le texte source est uniformément dans une langue déjà identifiée, et encodé en UTF-8. En effet, pour être utilisé efficacement, SANDOH nécessite une segmentation en réseau, plus complexe à traiter (voir Conclusion et perspectives, page 11).

On suppose que les documents sont en HTML *bien formé* (mais pas forcément en HTML ou XHTML valide), et codés en UTF-8. D'autres composants de SECTra ont la charge de convertir les documents HTML, bien formés ou non, en document bien formés et codés en UTF-8.

On distingue 3 fonctionnalités essentielles :

- la segmentation de documents HTML (création de squelettes de documents et de listes de segments),
- l'opération inverse (habillage de squelettes à partir de listes de segments traduits)
- et l'envoi à SECTra des squelettes et des segments.

Un squelette doit contenir les liens vers les segments qui l'*habillent*. C'est une sorte de « document à trous », avec des variables qui sont les liens vers les segments.

2.5.1 SEGMENTATION DE DOCUMENTS HTML

Cette fonctionnalité consiste à séparer la *structure* d'un document et son *contenu textuel*.

Voici un exemple des sorties que l'on souhaite obtenir :

Données en entrée	Données en sortie
Un document HTML en français : <html> <head> <title>Titre de la page</title> </head> <body> Bonjour tout le monde ! </body> </html>	*Un squelette de document HTML :* <html> <head> <title>{_1_}</title> </head> <body> {_2_} </body> </html>
	Une liste de segments : Segment 1 (français) : Titre de la page Segment 2 (français) : Bonjour tout le monde !

SegDoc ne fait que la segmentation. Pour la mémorisation, les squelettes et les segments sont confiés à SECTra.

Il faut faire attention à n'extraire que le contenu textuel. Il ne suffit donc pas de prendre tout ce qui apparaît entre les balises : le code JavaScript ou CSS, par exemple, n'est pas un texte, il fait partie du squelette de la page. D'un autre côté, certaines balises, par exemple les liens hypertextuels, les indications de formatage textuel (gras, italique, etc.), font partie du contenu textuel, et non du squelette.

2.5.2 Opération inverse : habillage de squelettes

L'opération inverse consiste à habiller des squelettes.

Voici un exemple des sorties que l'on souhaite obtenir :

Données en entrée	Données en sortie

Un squelette de document HTML :
 	
`<html>`
`  <head>`
`  <title>{_1_}</title>`
`  </head>`
`  <body>`
`  <div>{_2_}</div>`
`  </body>`
`</html>`
	Un document HTML en français :
	
	`<html>`
	`  <head>`
	`  <title>Titre de la page</title>`
	`  </head>`
	`  <body>`
	`  <div>Bonjour tout le monde !</div>`
	`  </body>`
	`</html>`

Une liste de segments : Segment 1 (français) : Titre de la page Segment 2 (français) : Bonjour tout le monde !	

Cette fonctionnalité prend tout son intérêt lorsqu'on lui fournit des traductions de segments, et non les segments originaux. Pour cela, SegDoc demande simplement à SECTra la « meilleure » traduction de chaque segment. Cette notion de « meilleure » traduction est développée dans le document sur SECTra-v3.

Données en entrée	Données en sortie

Un squelette de document HTML :
 	
`<html>`
`  <head>`
`  <title>{_1_}</title>`
`  </head>`
`  <body>`
`  <div>{_2_}</div>`
`  </body>`
`</html>`
	Un document HTML en français :
	
	`<html>`
	`  <head>`
	`  <title>Page title</title>`
	`  </head>`
	`  <body>`
	`  <div>Hello world !</div>`
	`  </body>`
	`</html>`

Une liste de segments : Segment 1 (anglais) : Page title Segment 2 (anglais) : Hello world !	

2.5.3 COMMUNICATION AVEC SECTRA-V3

Comme dit plus haut, la mémorisation des squelettes et des segments, leur identification, et l'obtention d'une traduction, sont délégués à SECTra-v3.

Afin de gérer efficacement les mémoires de traductions, SECTra-v3 a besoin d'informations précises sur les segments (leur contexte) et les squelettes (leur origine). Voir , . Ces informations doivent donc être fournies par SegDoc à SECTra-v3.

2.6 SPÉCIFICATIONS EXTERNES

2.6.1 NOTIONS ESSENTIELLES

L'intégration avec SECTra-v3 exige de raffiner certaines notions : les notions de segment, de contexte et de langue. Ces notions ont été définies dans le précédent rapport d'avancement intermédiaire (à T0+12) T7o-L2.2.b-SECTra_w, nous les rappelons ici.

La notion de segment est centrale. Un segment est à la base un objet, textuel ou plus complexe (par exemple un graphe de chaînes), monolingue, dont la langue est identifiée. De nombreux types de segments dérivent de cette définition de base, et sont présentés dans la thèse de HUYNH Cong Phap (voir notamment p. 169 et suivantes), notamment la notion de multisegment (collection de segments dont la relation entre eux n'est pas explicite). Plusieurs de ces dérivations font appel de façon impérative à la notion de langue-source, ce qui pose problème dans la mesure où la pratique montre que cette dernière n'est pas toujours connue.

C'est pourquoi, en se basant sur les définitions de la thèse, on établit pour la nouvelle version une notion plus générique de *multisegment contextualisé,* c'est à dire une collection de segments qui, dans un contexte donné, entretiennent une relation. Cette relation pourra être plus ou moins spécifique selon les cas : lien de traduction, lorsque la langue-source est connue (cela correspond alors au segment multilinguisé contextualisé de la thèse), mais aussi lien d'équivalence sémantique entre segments de langues différentes, voire entre reformulations jugées équivalentes dans une même langue. Dans tous les cas, le contexte de la relation est présent, et on appelle occurrence l'appariement d'un segment (isolé ou membre d'un multisegment) et d'un contexte donné.

La notion de *contexte* mérite d'être encore affinée. Actuellement, il s'agit, comme dans la version précédente, du document dont est issu le segment. On ajoute à cette définition le chemin du segment, qui correspond dans le cas d'un document XML ou HTML au chemin d'accès du segment dans le DOM du document. On espère ainsi pouvoir distinguer plus efficacement, par exemple entre une occurrence d'un segment qui apparaît toujours dans le menu d'une page Web d'un site donné, et une occurrence de ce même segment lorsqu'il apparaît dans le corps du texte du même site.

La notion de *langue* est aussi à préciser par rapport à la thèse de Phap. Il faut en effet distinguer, en plus de la langue proprement dite, ses variantes linguistiques (dialecte, spécialité, etc.) et scripturales (système d'écriture, transcription). Le codage ISO-639-2 utilisé ne permet pas, par exemple, de distinguer clairement le mandarin du cantonais. Pour le mandarin, il ne permet pas de savoir si ce dernier est écrit en chinois simplifié (majeure partie de la république populaire de Chine, mais avec d'importantes exceptions) ou en chinois traditionnel (Taïwan, diaspora chinoise, mais aussi Macao et Hong Kong). Il faudrait aussi parler des transcriptions en alphabet latin comme le pinyin ou Bopomofo. C'est pourquoi nous suivons pour la nouvelle version les recommandations du W3C sur le codage des langues[1].

2.6.2 ARCHITECTURE

Comme SECTra-v3, SegDoc est utilisable à deux niveaux : en tant qu'API Java, ou bien en tant que service Web.

[1] http://www.w3.org/International/articles/language-tags/Overview.en.php

2.6.3 DÉPENDANCES

SegDoc repose sur SECTra-v3 pour de nombreux aspects, et en dépend donc directement. Le couplage se fait avec l'API Java de SECTra-v3 : SegDoc utilise les classes exposées par l'API Java de SECTra-v3 (Segment, Doc, etc.).

2.6.4 FONCTIONNALITÉS

SegDoc expose les fonctionnalités suivantes :

- seg(URI docURI, **short** docType, String docContent, String defaultLang):String : segmentation d'un document dont on précise l'URI, le type, le contenu, et la langue par défaut.
- wear(URI docURI, String skel, String targetLang):String : reconstitution d'un document à partir d'une URI, d'un squelette, et d'une langue cible.

SECTra-v3 s'occupe de gérer les segments multilinguisés, et de rechercher ou au besoin générer (via Tradoh) des traductions dans la langue voulue par la fonction *wear*.

2.7 SPÉCIFICATIONS INTERNES

2.7.1 ALGORITHME DE SEGMENTATION

À ce stade, le document doit être en HTML bien formé et codé en UTF-8. Des caractères de début de segment et de fin de segment sont choisis, et sont remplacés par des entités HTML s'ils sont présents dans le document original.

Le document est dans un premier temps découpé en tokens. Un token est soit :

- une *balise* :
  - *forte*, qui délimite, selon la sémantique HTML, des blocs de texte relativement autonomes (≈paragraphes) ;
  - *non textuelle*, qui délimite des fragments non textuels (scripts, feuilles de style, contenu binaire, etc., intégrés dans le HTML) ;
  - *toute autre balise (balise faible)*, qui ne délimite généralement rien (balises de mise en forme : italique, gras, etc.) ;
- un *attribut* (à l'intérieur d'une balise) :
  - *contenant du texte* (par exemple texte à afficher lorsque le curseur de la souris passe au-dessus d'une image ou d'un lien) ;
  - *ne contenant pas de texte* ;
- un *texte* : tout ce qui est à l'extérieur des balises.

Ensuite, on parcourt la liste des tokens, et on traite chacun suivant sa nature :

- si c'est une balise non textuelle :
  - *ouvrante*, on incrémente un *compteur de contenu non textuel*, et on écrit le token dans le squelette ;
  - *fermante*, on décrémente le *compteur de contenu non textuel*, et on écrit le token dans le squelette ;
- si c'est autre chose :
  - si le *compteur de contenu non textuel* est supérieur à 0, on écrit le token dans le squelette ;
  - si le *compteur de contenu non textuel* est égal à 0,
    - si c'est une balise forte, ou un attribut ne contenant pas de texte, on écrit le token dans le squelette ;
    - dans les autres cas, on ajoute le contenu au *tampon de segment courant*, et non au squelette. Dès que l'on rencontre un token qui correspond à un autre

cas, on crée un segment à partir du tampon de segment courant, on remet à zéro ce tampon, et on reprend le traitement normal.

Lors de la création d'un segment :

- On enlève les balises faibles présentes au début et la fin du segment putatif, si elles sont équilibrées (par exemple, dans `<i>ceci est un <b>texte</b></i>`, on enlèvera `<i>` et `</i>`). Les balises enlevées sont ajoutées au squelette. Plus précisément, chacune d'elles reçoit un identificateur unique et un contenu qui est formé du nom d'élément original, de sa position dans le flux d'entrée, et des valeurs des attributs éventuels, comme le fait MosesWeb (ex : `[$$_htm_23 2345 i att="val"]` pour `<i att="val">`). Ici, on a supposé que la balise commençait à pa position 2345 dans le fichier d'entrée. Cela permet de réinsérer les balises originales en les faisant « glisser » en suivant l'alignement créé après traduction (via Giza++) d'un segment entre l'original et sa traduction.
- On vérifie que le segment contient effectivement du texte. S'il ne contient que des balises faibles, ou des blancs, on les écrit dans le squelette et aucun segment n'est créé.
- Si il reste du texte pour créer un segment, on ajoute dans le squelette :
    1. un marqueur de début de segment ;
    2. l'identifiant du segment (que l'on demande à SECTra) ;
    3. un marqueur de fin de segment.

Après la création du squelette et des segments, on met à jour les contextes des segments.

2.7.2 API Java

L'API Java est basée sur une seule classe SegDoc, qui expose les méthodes présentées dans la spécification.

Constructeur :

- Segdoc(String corpus) : permet la création d'un nouveau SegDoc pour un corpus donné.

2.7.3 API REST

L'API REST est basée sur l'API Java. Elle expose deux services, correspondant au deux fonctionnalités présentées dans la spécification.

2.8 Implémentation

2.8.1 État de l'implémentation : SegDoc 1.0

Toutes les fonctionnalités sont implémentées, hormis le traitement des attributs des balises. Il reste des bogues dans la segmentation et l'habillage, sur certaines pages « exotiques » quant à leur manière de concevoir les pages HTML : CSS avancé, contenu généré à la volée, etc. Toutefois, ce n'est pas bloquant pour le développement et le test de SECTra-v3 et du logiciel iMAG.

2.8.2 État de l'implémentation : SegDoc 1.1

La segmentation en phrases n'est pas encore opérationnelle. On se dirige vers la réutilisation de règles existantes au format SRX[2], qui sont utilisées pour effectuer la segmentation en phrases, notamment dans des logiciels d'aide aux traducteurs (par exemple OmegaT[3]). Le développement de la version 1.1 s'annonce donc peu coûteux.

[2] Segmentation Rules eXchange

[3] http://en.wikipedia.org/wiki/OmegaT

Comme XLIFF et TMX, qui sont supportés par SECTra-v3, SRX fait partie du standard OAXAL[4] [6], supporté par l'OASIS.

2.8.3 EXEMPLES

2.8.3.1 Segmentation avec l'API Java

Commande :

```
String skel = new Segdoc("test").seg(
                    new URI("http://localhost/test"),
                    Sectra.SEG_TYPE_HTML,
                    "<html> Hors du div, <i>point</i> de salut!<br/><div><b>Un div normal.</b></div>
                            <script>Et dans un script ?</script><div><!--Commentaire--></div>
                            Une liste<ul><li>Premier item</li><li>Deuxième item</li></ul></html>",
                    "fra"
) ;
System.out.println(skel);
```

Affichage :

```
<html>  SEG:2595  <div><b>  SEG:2596  </b></div>
<script>Et dans un script ?</script><div><!--Commentaire--></div>
  SEG:2597  <ul><li>  SEG:2598  </li><li>  SEG:2599  </li></ul></html>
```

Par défaut (c'est paramétrable), les caractères et marquent les limites de segment.

Habillage I Java

Commande :

```
String newDoc = new Segdoc("test").wear(
            new URI("http://localhost/test"),
            Sectra.SEG_TYPE_HTML,
            skel,
            "eng"
);
System.out.println(skel);
```

Affichage :

```
<html> Out of the div, <i>point</i> of hello! <br/><div><b> A normal div.</b></div>
<script>Et dans un script ?</script><div><!--Commentaire--></div>
A list<ul><li> First item</li><li> Second item</li></ul></html>
```

2.8.3.2 Segmentation d'une page Web réelle

Les figures 1 à 3 présentent un document HTML réel (en anglais), le résultat de sa segmentation, puis son habillage avec des segments traduits en anglais.

2.8.4 PERFORMANCES

Nous avons commencé une évaluation dans le cadre de la traduction de pages HTML avec SECTra-v3. Il s'agit donc d'une évaluation conjointe SegDoc + SECTra-v3.

SegDoc a été évalué sur un Pentium double cœur E5200 (2,5 GHz).

En segmentation, SegDoc traite 1 Ko en 38 ms.

En habillage, c'est très variable, et cela dépend du temps nécessaire pour la traduction. Si tous les segments sont déjà traduits dans SECTra, 1Ko de squelette est traité en 17 ms.

[4] Open Architecture for XML Authoring and Localization

3. Conclusion et perspectives

Nous prévoyons dans un premier temps d'améliorer la couverture de SegDoc 1.0. La segmentation en phrases sera introduite dans SegDoc 1.1. SECTra-v3 disposera alors d'un segmenteur équivalent à celui de SECTra-v1.

Nous ajouterons ensuite les fonctionnalités suivantes :

- version 1.2, SegDoc monolingue avancé :
  - détection des fichiers satellites (images, Flash, etc.), et possibilité d'habiller les documents avec des versions localisées de ces fichiers, lorsqu'elles sont disponibles ;
  - segmentation récursive, pour mieux supporter les niveaux et les ambiguïtés de segmentation des documents arborescents (cas des documents HTML) ;
  - traitements d'autres types de documents XML (par exemple ODT, DOCX) ;
  - intégration de SANDOH pour la détection du codage et de la langue dans les documents monolingues.
- version 1.3, SegDoc multilingue avancé :
  - segmentation en réseau (plus complexe que la segmentation récursive), pour un support complet des ambiguïtés de segmentation de tous documents, ce qui permet notamment d'utiliser en parallèle plusieurs niveaux de segmentation *(voir point suivant) ;*
  - intégration de SANDOH pour ajouter un niveau de segmentation en tokens monolingues pour les documents multilingues.

Références bibliographiques

[1] **HUYNH, Cong-Phap (2010)** *Des suites de test pour la TA à un système d'exploitation de corpus alignés de documents et métadocuments multilingues, multiannotés et multimédia.* Thèse informatique, Grenoble :Univérsité Joseph Fourier, 2010, 228 p.

[2] **VO-TRUNG, Hung. SANDOH (2004)** *Système pour l'ANlyse des DOcuments Hétérogènes*, actes de la conférence internationale JADT 2004, volume 2, pp. 1177-1184, mars 2004, Louvain-la-Neuve, Belgique.

[3] Philipp Koehn, Philipp Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, Evan Herbst (2007) *Moses: Open Source Toolkit for Statistical Machine Translation*, Annual Meeting of the Association for Computational Linguistics (ACL), demonstration session, Prague, Czech Republic, June 2007.

[4] **LARIBI, Nahla (2009)** *Documentation et visualisation des étapes de la segmentation de MosesWeb et participation à la réalisation de programmes d'échange entre les formats de segmentation de MosesWeb et de SECTra_w*. Stage RICM3, Grenoble : Université Joseph Fourier, Polytech'Grenoble, 2009.

[5] **VO-TRUNG, Hung (2009)** *Réutilisation de traducteurs gratuits pour développer des systèmes multilingues.* In Proceedings of the Conférence Rencontre des Étudiants, 2004.

[6] **Zydroń Andrzej, Saldana Derek (2009)** *Reference Model for Open Architecture for XML Authoring and Localization Version 1.0. Organization for the Advancement of Structured Information Standards.*
En ligne : http://docs.oasis-open.org/oaxal/V1.0/oaxal-v1.0.html

4. ANNEXES : IMAGES D'ÉCRANS

4.1 ÉTATS D'UNE PAGE HTML TRAITÉE PAR SEGDOC

Figure 1: page Web d'origine (en français)

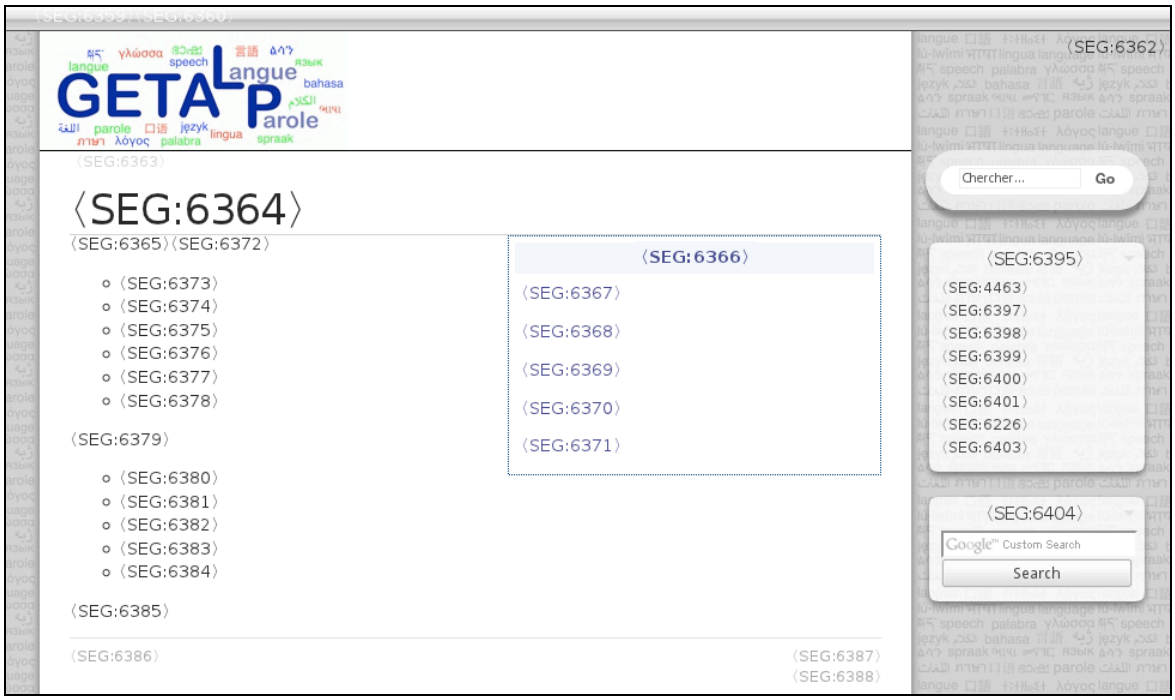


Figure 2: squelette de page Web généré par SegDoc-1

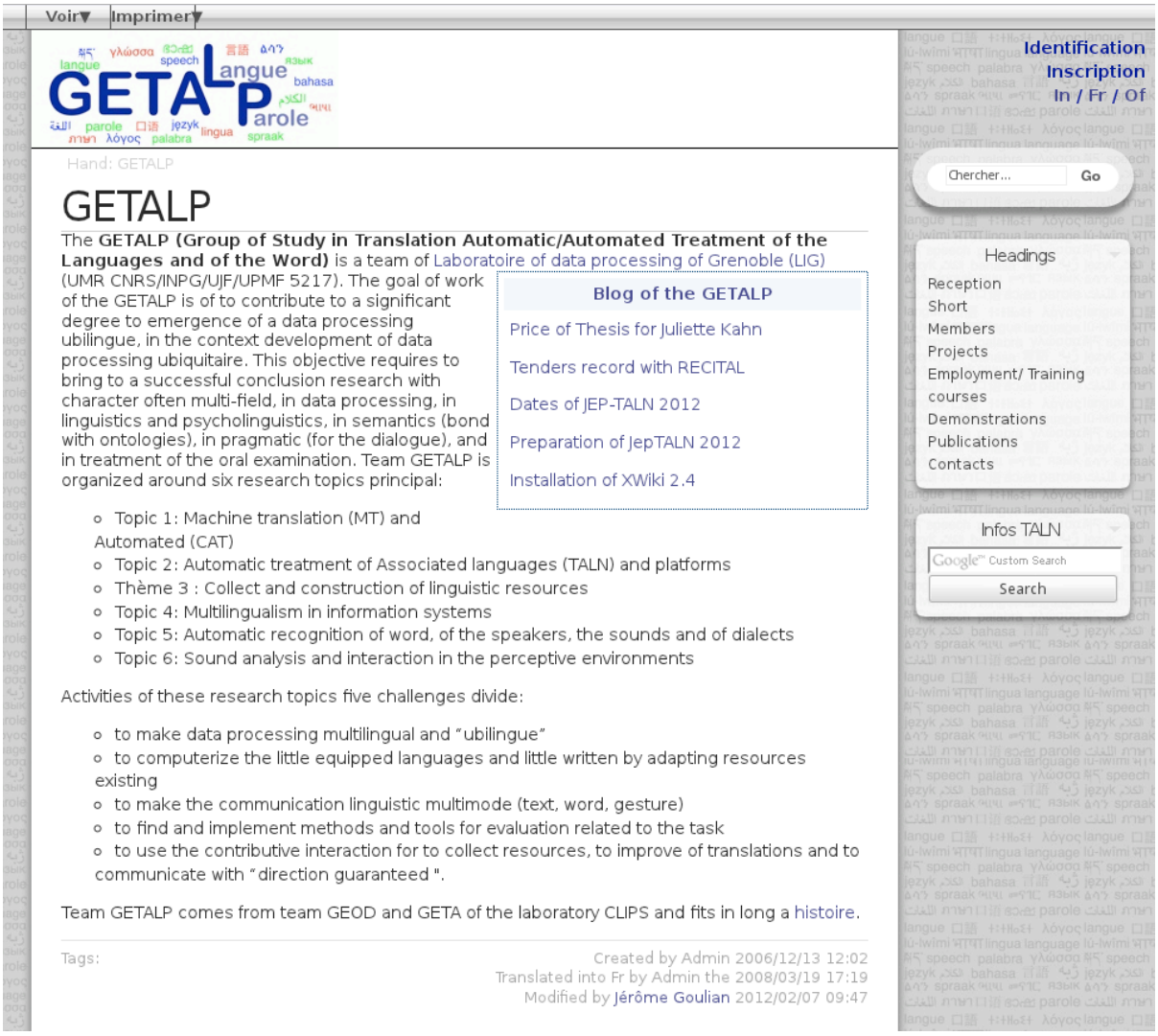


Figure 3: page Web habillée par SegDoc avec des segments traduits par Google en anglais

4.2 ÉCRANS 2 — COPIE D'ÉCRAN DE RSTTOOL

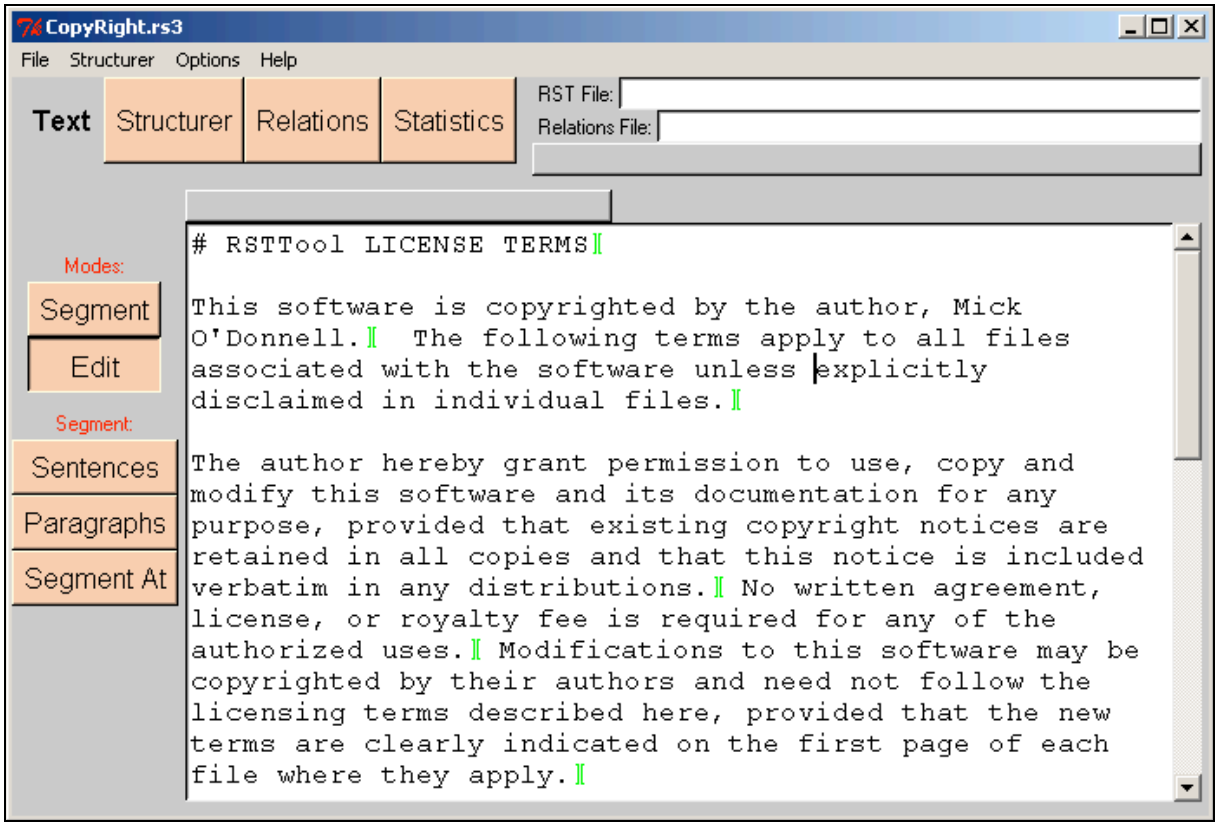


Figure 4 : Interface de segmentation de RSTTool : les "][" délimitent les segments

4.3 ÉCRANS 3 — COPIE D'ÉCRAN D'OMEGAT

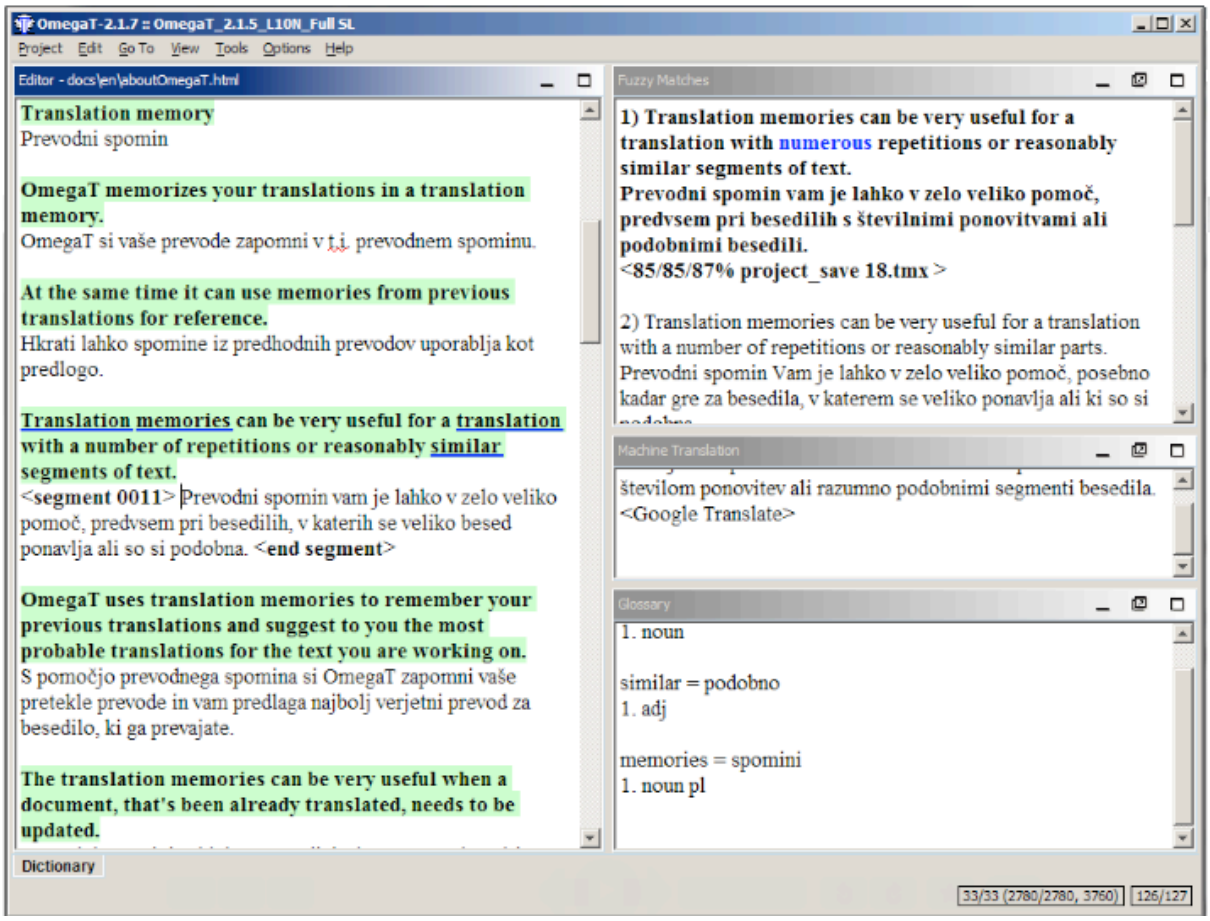

OmegaT-2.1.7 = OmegaT_2.1.5_L10N_Full SL
Project Edit Go To View Tools Options Help
Editor - docs\en\aboutOmegaT.html
Translation memory
Prevodni spomin
OmegaT memorizes your translations in a translation memory.
OmegaT si vaše prevode zapomni v t.i. prevodnem spominu.
At the same time it can use memories from previous translations for reference.
Hkrati lahko spomine iz predhodnih prevodov uporablja kot predlogo.
Translation memories can be very useful for a translation with a number of repetitions or reasonably similar segments of text.
<segment 0011> Prevodni spomin vam je lahko v zelo veliko pomoč, predvsem pri besedilih, v katerih se veliko besed ponavlja ali so si podobna. <end segment>
OmegaT uses translation memories to remember your previous translations and suggest to you the most probable translations for the text you are working on.
S pomočjo prevodnega spomina si OmegaT zapomni vaše pretekle prevode in vam predlaga najbolj verjetni prevod za besedilo, ki ga prevajate.
The translation memories can be very useful when a document, that's been already translated, needs to be updated.
Fuzzy Matches
1) Translation memories can be very useful for a translation with numerous repetitions or reasonably similar segments of text.
Prevodni spomin vam je lahko v zelo veliko pomoč, predvsem pri besedilih s številnimi ponovitvami ali podobnimi besedili.
<85/85/87% project_save 18.tmx >
2) Translation memories can be very useful for a translation with a number of repetitions or reasonably similar parts.
Prevodni spomin Vam je lahko v zelo veliko pomoč, posebno kadar gre za besedila, v katerem se veliko ponavlja ali ki so si
Machine Translation
številom ponovitev ali razumno podobnimi segmenti besedila.
<Google Translate>
Glossary
1. noun
similar = podobno
1. adj
memories = spomini
1. noun pl
Dictionary
33/33 (2780/2780, 3760)
126/127

Figure 5 : Interface de travail d'OmegaT : les segments sont surlignés en vert