Synergies Tunisie n° 4 - 2015 p. 101-111

Un langage pivot entre langue naturelle et langage informatique

Laurent Tromeur
Ontomantics S.A.S. France
laurent.tromeur@ontomantics.com

Reçu le 17-03-2015 / Évalué le 24-04-2015 / Accepté le 15-06-2015

Résumé

Nous abordons dans cet article la notion d'espace dans le cadre de la traduction automatique d'une langue naturelle vers un langage informatique. Cette problématique inclut des notions d'espace linguistique et d'espace informatique et fait appel à des notions qui appartiennent au domaine de l'ingénierie linguistique et plus précisément, à celui du Traitement Automatique des Langues (TAL). Nous analysons ici les relations entre ces deux notions et la façon dont elles interagissent, via l'intermédiaire d'un langage pivot.

Mots-clés : traduction automatique, langage pivot, représentations métalinguistiques, dictionnaires électroniques, transducteurs

A pivot language between natural language and computer language

Abstract

We discuss in this article the concept of space as part of the automatic translation of natural language into computer language. This working hypothesis includes the concepts of linguistic space and computer space and lies upon notions originating from the field of language engineering and more specifically, from Natural Language Processing (NLP). We analyze the relationship between these two concepts and how they interact, via the intermediary of a pivot language.

Keywords : automatic translation, pivot langage, metalinguistic representations, electronic dictionnaries, transducer

Introduction

Le sujet qui nous intéresse ici est la question du langage pivot entre une langue naturelle et le langage informatique. La problématique du passage de l'une à l'autre constitue l'un des thèmes centraux de la Traduction Automatique. Nous allons commencer par expliciter les notions d'espace linguistique et d'espace informatique.

À l'écrit et d'un point de vue purement graphique, un texte est un ensemble de mots et de signes concaténés. C'est donc un espace qui se lit de façon linéaire, mais qui ne s'interprète pas forcément linéairement (références, procédés de reprise anaphorique, etc.). De plus, cette linéarité est à modérer dans le sens où un lecteur expérimenté ne lit pas les mots les uns à la suite des autres mais de façon globale (par blocs). Notons également que la portée du message écrit peut être très variée : il peut s'agir d'un message à caractère informatif, déictique, d'œuvres littéraires, de discours oraux retranscrits, etc.

En informatique, les *mots* au sens linguistique n'existent pas (on parle de *chaînes de* caractères). Si on veut pouvoir les prendre en compte, il faut pouvoir les spécifier en indiquant par exemple au programme ou au langage de programmation qu'un mot est une chaîne de caractères bornée à gauche et à droite par un espace ou une marque de ponctuation. Le but d'un langage informatique est de faire fonctionner un programme : le langage est interprété par l'ordinateur et traduit en langage machine bas niveau afin d'exécuter toutes sortes d'opérations logiques et mathématiques. D'une façon analogue au langage humain, le langage informatique obéit à une syntaxe stricte ainsi qu'à un lexique (qui diffèrent tous deux selon le langage) dont le non-respect empêche l'interprétation par la machine. Certains langages sont lus par la machine de façon linéaire, comme le lecteur humain parcourt un texte écrit : ce sont les langages impératifs. Il existe deux autres paradigmes de programmation dont les langages ne sont pas interprétés linéairement : la programmation logique (qui fonctionne par inférence) et la programmation fonctionnelle.

Dans le cadre de travaux sur la traduction automatique entre une langue naturelle et un langage informatique, il faut parvenir à faire le lien entre ces deux types de textes très différents. Quelles sont les méthodes à employer pour traduire dans un langage informatique des instructions exprimées en langue naturelle ?

Présentation de nos travaux

La thèse se prépare en collaboration avec une Jeune Entreprise Innovante du génie logiciel, la SSII Ontomantics. L'entreprise a développé une solution d'automatisation Web 2.0 en rupture avec les outils existants ; le système Ontomantics est une plateforme permettant de mettre en place des applications rapidement et de façon simplifiée, puisqu'il propose une interface graphique de développement et ne nécessite la connaissance d'aucun langage de programmation. Son objectif est d'autoriser tout un chacun à pouvoir mettre en œuvre les applications qu'il désire en seulement quelques heures, là où un développeur mettrait plusieurs jours pour parvenir au même résultat. Le développement ainsi que l'exécution de l'application se font toutes deux via un

navigateur Internet et ne nécessitent aucune installation sur le poste client puisque tout est joué directement depuis un serveur. La technologie Ontomantics permet des gains de temps et d'argent puisqu'il n'est pas nécessaire de faire appel à des ressources externes pour modéliser son besoin applicatif (achat de logiciel tiers, location des services d'un expert, etc.).

Le système se divise en plusieurs modules : une partie permet de mettre en place la ou les bases de données qui seront utilisées dans l'application ; un second module permet de créer les écrans nécessaires à la navigation avec un système intuitif de *Drag & Drop* (*Glisser - Déposer*) et un dernier module correspond à la mise en place des règles régissant le comportement de l'application.

La philosophie de l'entreprise est de rendre accessible le développement d'applications à des personnes qui n'ont pas un profil d'informaticien. C'est dans ce cadre que s'inscrit notre projet. La version actuelle du logiciel ne permet pas d'atteindre complètement ce but, car des connaissances en informatique restent nécessaires à l'élaboration d'applications. Pour faciliter la formulation des requêtes par les utilisateurs, nous voulons mettre en place une interface en langue naturelle qui fasse appel à des ressources linguistiques et à diverses techniques du traitement automatique des langues.

La chaîne de traitement de l'information

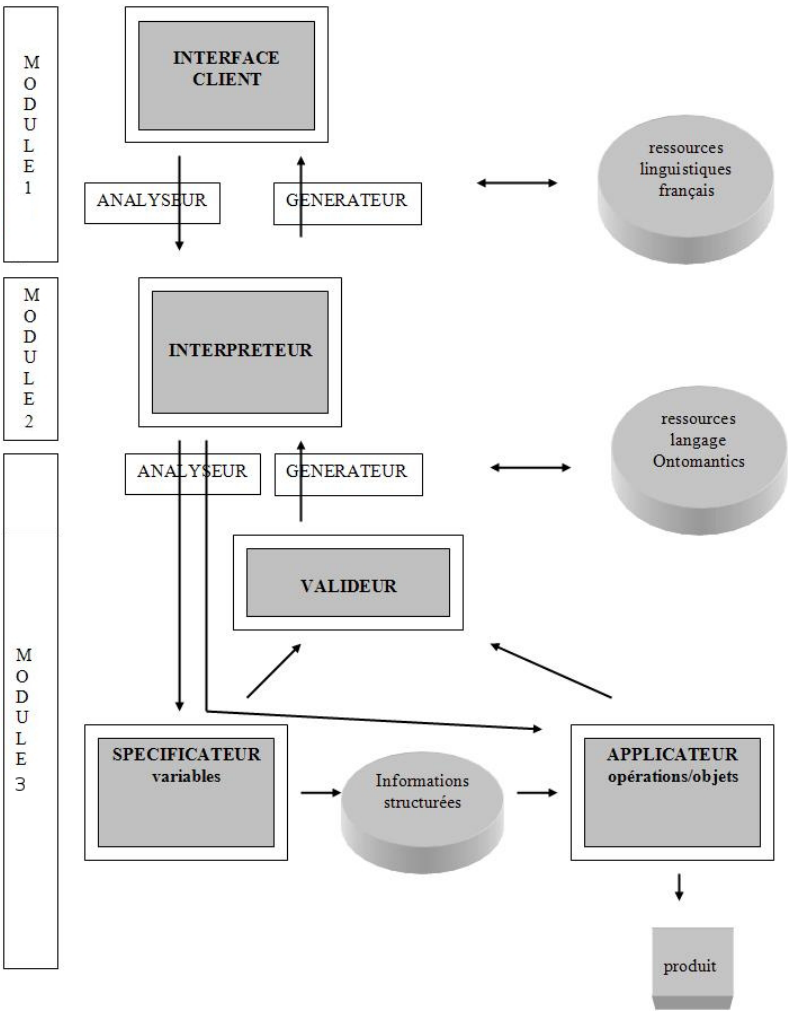
Voici l'architecture que nous souhaitons mettre en place : le *designer* d'Ontomantics (le composant qui permet de créer son application, par opposition au *player* qui permet d'exécuter cette dernière une fois réalisée ou en cours de réalisation pour débogage et test) comporte les trois modules séparés suivants:

- la partie permettant de créer sa (ses) base(s) de données,
- la partie permettant de créer les écrans qui permettront d'interagir avec l'application pendant son exécution,
- le moteur de règles qui permet de modeler le comportement de l'application.

Ces éléments constituent ce que nous appelons le module Ontomantics, et qui se trouve à la fin de la chaîne de traitement. Le système se compose de deux autres modules qui comportent chacun des fonctions et des variables spécifiques : le module linguistique et le module interface. Le premier niveau de la chaîne de traitement est le niveau linguistique ; comme son nom l'indique, c'est la couche qui contient toutes les données linguistiques de l'architecture, notamment des listes de mots pour chaque métier ou domaine classés selon leur fonction (prédicats, arguments, actualisateurs). Dans ce module, les instructions en langue naturelle (en l'occurrence le français)

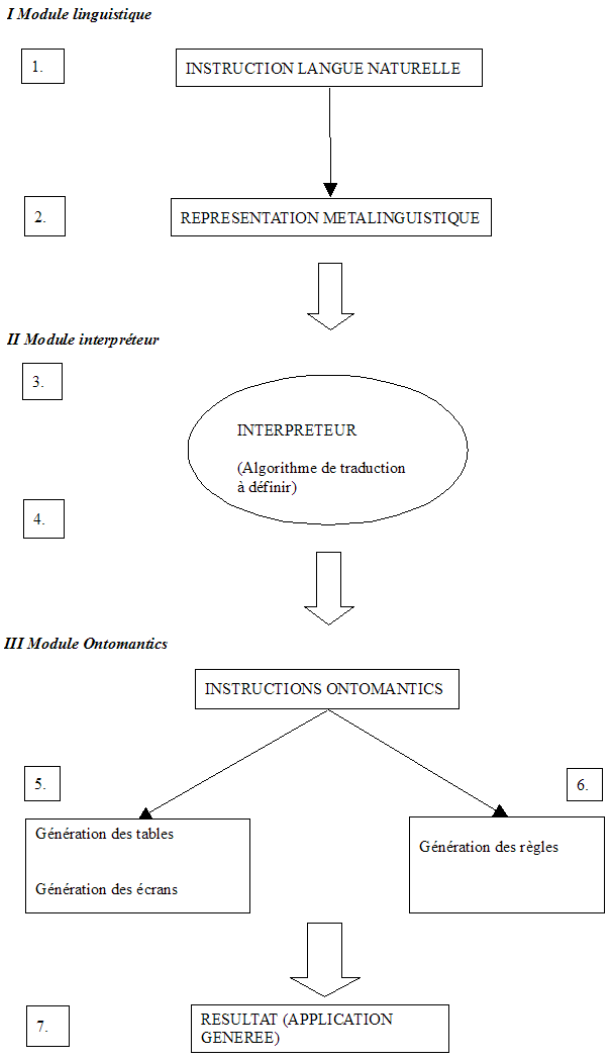
seront analysées de telle sorte que les phrases en rapport avec ces instructions soient associées à des représentations métalinguistiques. La méthodologie et la technologie sous-jacentes à l'analyse sont proches de celles d'un module d'analyse en traduction automatique[1].

Le module a également pour fonctionnalité de permettre à l'utilisateur de modeler son application. Le deuxième module, l'interface, permet de transposer les instructions en langue naturelle en instructions en langage Ontomantics. Le module est appelé '*interpréteur*' car il se charge d'analyser les représentations métalinguistiques et de les transformer en instructions Ontomantics pour pouvoir générer les tables, les écrans et les règles nécessaires au fonctionnement de l'application finale.



Pour assurer la continuité de l'information du début à la fin de la chaîne de traitement, il est nécessaire d'identifier les composantes de chaque module, de les classer et d'établir la nature des relations entre les composantes des différents niveaux. Nous avons dressé la typologie des instructions que peut exécuter la plateforme pour chaque niveau de la chaîne de traitement : c'est notamment ce travail qui permet de faire correspondre par exemple l'instruction en langue naturelle *Remplir la table d'utilisateurs X* du module linguistique, à la série d'instructions que nécessite son exécution au niveau informatique.

Voici un schéma qui représente la chaîne de traitement de l'information :



Voyons à présent un exemple de règle telle qu'elle est affichée dans la plateforme. C'est une simple règle de mise à jour d'éléments dans une base de données :

Nous voyons que cette règle comporte deux conditions et plusieurs actions. La difficulté à réduire une règle de ce type à une instruction ou une série d'instructions simples exprimables en langue naturelle dépend de la complexité de la règle.

Les représentations métalinguistiques

Rappelons ce que nous appelons représentations métalinguistiques. Les phrases sont représentées en termes de prédicat et d'argument. Ainsi:

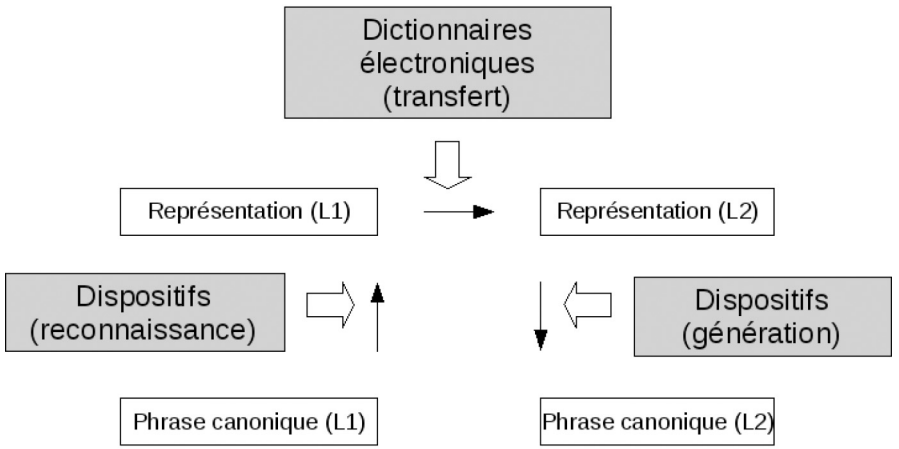
(1) L'employé stocke le matériel
donne lieu à la représentation métalinguistique suivante:
Pl ➛ stocker (employé, matériel)

Dans le cas où l'on a affaire à des phrases complexes, les représentations sont identiques si ce n'est que l'un des arguments est lui-même une représentation métalinguistique. Par exemple:

(2) Le chef demande à l'employé de stocker le matériel
aura comme représentation métalinguistique :
P2 ➛ demander (chef, stocker (employé, matériel))

Autrement dit, un discours donné, en tant qu'ensemble structuré de phrases, peut être représenté par un agencement de différentes représentations métalinguistiques qui rend compte de la complexité phrastique. Les unités fondamentales de discours sont les phrases en tant que prédicats saturés par des arguments.

La transformation des représentations métalinguistiques en phrases canoniques fait appel à un dispositif qui procède obligatoirement en deux étapes : la linéarisation (i.e. la spécification du rôle syntaxique des arguments), d'une part, l'actualisation (i.e., notamment, la spécification des différentes informations relatives au temps et à la détermination des unités nominales), d'autre part. L'efficacité du dispositif est améliorée si l'on subdivise l'actualisation en deux sous-étapes qui tiennent compte de la possibilité d'assimiler les substantifs soit à des prédicats soit à des arguments. Inversement, après l'identification des prédicats et des arguments, la transformation des phrases canoniques en représentation métalinguistique consiste à spécifier la nature des autres éléments de la phrase du point de vue de l'actualisation afin de les exclure en tant que tels et de leur associer, s'il y a lieu, un code relatif à leur fonction dans la phrase.



Le dispositif s'apparente à un système de traduction automatique tel que ceux qui ont été expérimentés au LDI[2]. La différence ici est qu'il est question de passer, non pas d'une langue naturelle source vers une langue naturelle cible, mais d'une langue naturelle vers un langage informatique. Dans un tel système de traduction, les trois modules (celui de reconnaissance, celui de transfert et celui de génération) font appel aux dictionnaires électroniques spécifiques à chacune des langues mises en correspondance puisqu'une grande partie des informations qui y sont enregistrées sont utilisées dans les dispositifs qui, dans une langue donnée, permettent le passage d'une phrase canonique à sa représentation métalinguistique (tâche prise en charge par le module de reconnaissance) et, inversement, le passage d'une représentation métalinguistique à l'une des phrases canoniques qui lui sont associés (tâche qui incombe au module de génération). Cependant, ces dictionnaires jouent un rôle crucial en ce qui concerne le passage d'une représentation métalinguistique d'une phrase canonique en Langue source à sa représentation métalinguistique équivalente en Langue cible ; c'est pourquoi ils ont la place centrale dans le schéma ci-dessus.

Pour le traitement des données linguistiques, le module de transfert se présente sous la forme de dictionnaires monolingues coordonnés contrairement aux systèmes qui font intervenir à ce niveau des dictionnaires bilingues. Cela implique notamment que chaque dictionnaire décrivant une langue donnée est doté d'un champ qui présente des équivalents de traduction possibles du lemme et à partir duquel il est possible d'accéder aux dictionnaires de la langue mise en correspondance. Le module de transfert fonctionne donc comme un pointeur qui utilise les dictionnaires électroniques de la langue source pour identifier les données lexicales majeures apparaissant dans la représentation métalinguistique de la phrase canonique en Langue 1 et, via le champ traduction, donner

leur équivalent dans la représentation métalinguistique correspondante en Langue 2 de telle sorte que le module de génération puisse utiliser les diverses informations rattachées aux unités lexicales majeures de la représentation métalinguistique en Langue 2 afin que le dispositif qu'il comporte puisse aboutir à une phrase canonique en Langue 2 qui sera donc la traduction de la phrase de départ.

L'avantage de notre système par rapport à un système de traduction plus classique est qu'il ne nécessite pas de prétraitement : les transducteurs de prétraitement servent à mettre en évidence les phrases canoniques des textes qui sont le point de départ des représentations métalinguistiques ; or, les instructions qu'entrera l'utilisateur seront déjà des phrases canoniques.

Intérêt de la notion de classes d'objets

Rappelons tout d'abord qu'une classe d'objets correspond à un ensemble d'items homogènes (à la fois morphologiquement et sémantiquement) qui sont caractérisés par leurs propriétés syntaxiques ; les classes sont distinguées selon qu'elles correspondent à des classes d'arguments (ex : les noms de *voie*), ou bien à des classes de prédicats (ex: les noms de *maladies physiques*).

L'intérêt de la notion de classes d'objets est de deux ordres ; d'une part, linguistique, d'autre part, informatique. Pour ce qui est du premier, mentionnons, entre autres, la possibilité de rendre compte des contraintes sur les déterminants des noms prédicatifs, des valeurs non standard des déterminants, des verbes supports et des adverbes. Du point de vue informatique, les classes d'objets étant assimilables à des grammaires locales, les phrases canoniques associées aux différents éléments qu'elles comportent peuvent donner lieu à la production de transducteurs les représentant et contribuant à une simplification de certains étiquetages métalinguistiques.

Signalons également que les classes d'objets présentées sous forme de grammaires locales sont également d'un grand intérêt pour le module de génération puisque après l'identification de l'équivalent dans la langue cible du prédicat et des arguments en langue source, il est possible à l'aide des dictionnaires électroniques de spécifier de quelles classes ils relèvent et de les associer ainsi à des grammaires locales qui permettent de générer les phrases canoniques souhaitées. Autrement dit, le champ classe dans les dictionnaires utilisés dans le module de transfert est une clef à partir de laquelle le système va pouvoir accéder au module de génération.

Voici un exemple d'instruction que l'utilisateur pourra entrer :

Fais un planning incluant 10 personnes et 3 activités pendant 2 mois
➛ *Faire Planning (10 personnes, 2 mois, 3 activités)*

À partir de cette instruction, le système devra générer la base de données correspondante et les écrans nécessaires à la navigation, mais il générera également les règles d'exécution de la requête.

Conclusion

Nos travaux récents nous ont permis de mettre en évidence plusieurs pistes de recherche. Dans l'immédiat, nous allons commencer par créer des applications plus complexes que celles réalisées jusqu'alors avec la plateforme, elles utiliseront notamment des bases de données plus importantes et feront appel à des règles plus avancées. Cela nous permettra de maitriser complètement l'outil et cela servira de base à la suite de nos travaux; nous étudierons les différents niveaux du module interface afin de faire le lien avec les niveaux des autres modules (linguistique et Ontomantics). Une fois que nous aurons établi tous les liens entre les différents modules, nous maîtriserons toute la chaîne de traitement de l'information et nous serons en mesure de travailler sur la traduction de la langue naturelle vers le langage Ontomantics grâce aux représentations métalinguistiques. L'autre grand chantier sera de mettre au point un système d'acquisition automatique de vocabulaire métier : l'idée étant de constituer des classes d'arguments élémentaires et de prédicats automatiquement à partir de corpus larges, qui constitueront ainsi les ressources linguistiques nécessaires au projet. De plus, un tel outil pourrait être utilisé pour de nombreux travaux futurs et intégré dans de nombreux systèmes de traduction automatique.

Bibliographie

Antoniadis, G., Ponton, C. 1996. Un système noyau de génération automatique. In : *Informatique et Langue Naturelle*, Actes du colloque de Nantes, 9-10 octobre 1996, p. 37-48.

Blanco, X., Buvet, P.-A. 1999. «À propos de la traduction automatique des déterminants de l'espagnol et du français », *Meta 44 :4*, Montréal, Les Presses de l'Université de Montréal, p.525-545.

Buvet, P.-A,. Blanco, X 2000. « De l'analyse syntactico-sémantique du lexique à la traduction automatique », *BULAG* 25, p. 69-87.

Blanco, X., Buvet, P.-A. 2004. Verbes supports et significations grammaticales. Implications pour la traduction espagnol-français. In : *Lingvisticae Investigatione* 27 (2), p.327-342.

Buvet, P.-A., Tromeur, L. 2010. « **Le dialogue homme-machine : un système de traduction automatique spécifique** » *Meta*, 2010. 1 (55), p.58-70.

Le Pesant, D., Mathieu-Colas, M. 1998, (éd.), *Les Classes d'objets*, *Langages* 131.

Tromeur, L. 2011. *Mise en place d'une interface en langue naturelle pour la plateforme Ontomantics*, Thèse de doctorat en sciences du langage.

Van Der Lans, R.-L. 2006. Introduction to SQL: Mastering the Relational Database Language, éd. Addison-Wesley.

Weizenbaum, J. 1966. « Eliza : a computer program for the study of natural language communication between man and machine », CACM, 9, p. 26-45.

Notes

1. Buvet P.-A. et X. Blanco.
2. Laboratoire Lexiques, Dictionnaires, *Informatique* (Université de Paris 13).

© Revue du Gerflint. Éléments sous droits d'auteur.