Conférence TALN 1999, Cargèse, 12-17 juillet 1999

Une bibliothèque d'opérateurs linguistiques pour la consultation de base de données en langue naturelle

Béatrice Bouchou, Denis Maurel

LI/E3i, Université François Rabelais
64, avenue Jean Portalis,
37200 Tours
(bouchou+maurel)@univ-tours.fr

Résumé

L'interrogation de bases de données en langue naturelle est une application directe du traitement automatique des langues naturelles. Son utilité va en s'accroissant avec le développement d'outils d'information accessibles au grand public à travers la Toile Internet. L'approche que nous proposons s'appuie d'une part sur les fondations linguistiques établies par la théorie de Z. S. Harris (dans l'élaboration du dictionnaire, et surtout dans la définition des opérateurs linguistiques), et d'autre part sur un outil informatique précis (les transducteurs). Elle représente une alternative aux traitements syntaxico-sémantiques habituellement développés dans des formalismes logiques. Elle s'appuie sur la constitution d'une bibliothèque d'opérateurs linguistiques pour les domaines d'application.

Mots clés

interrogation de base de données, langage naturel, opérateurs linguistiques, transducteur (automate à nombre fini d'états)

1. Introduction

Utiliser la langue naturelle pour exploiter les ordinateurs est souhaitable pour des raisons essentielles de confort d'utilisation d'une part, et d'ouverture générale vers un public peu formé à l'informatique d'autre part. L'accès à l'information stockée dans les bases de données est justement l'un des principaux axes grand public de l'informatique, c'est pourquoi le défi de la langue naturelle est important dans ce domaine. Les nouvelles possibilités offertes par la Toile Internet renforcent encore cette importance, car pour le commun des mortels la langue naturelle présente bien des avantages par rapport aux actuels systèmes d'interrogation par mots-clés, même sophistiqués.

Le but global est de traduire une question en langue naturelle vers une requête codée dans un langage informatique d'interrogation de base de donnée, comme SQL par exemple. La traduction doit associer aux mots de la question d'une part des éléments de la base considérée, tables, attributs ou instances, d'autre part des opérations à réaliser sur ces éléments, à coder dans les mots clés du langage d'interrogation.

Les premiers travaux sur la consultation de base de données utilisent tout naturellement un lexique pour détecter des mots clés dans la question. Ainsi dans le système BASEBALL [Green et al., 1960], l'information contenue dans la question est représentée par une liste de

paires (attribut - valeur). Un certain nombre de fonctions sont appliquées sur ces éléments. La connaissance nécessaire à la traduction est codée en partie dans la valeur liée à l'attribut, mais l'essentiel se trouve dans les fonctions. La limite évidente de cette technique est qu'elle est forcément étroitement liée au contexte. Dans ces conditions, on n'obtient de bons résultats que pour des produits « ad hoc », conçus dans un but trop particulier [Sabah 1997]. Cependant la limite est-elle due à l'approche par mots-clés en soi, ou bien au défaut de représentation adéquate de la connaissance nécessaire ? Il est à noter que dans BASEBALL, le traitement syntaxique est effectué selon la théorie syntaxique de Z. S. Harris que nous utilisons aussi pour définir nos opérateurs linguistiques.

Dans le but de généraliser l'outil d'interrogation (c'est-à-dire d'améliorer sa portabilité), plusieurs travaux s'attachent à exploiter automatiquement la structure de la base et son contenu pour alimenter le lexique [Kaplan 1984, Grosz et al., 1987, Clifford 88]. Dans son article, Kaplan présente un système, CO-OP, qui est portable dans la mesure où ses sources de connaissances sont uniquement un lexique et la base de données elle-même. De même, le système TEAM développé par Grosz et ses collègues a pour objectif principal la portabilité : il est conçu pour interagir avec deux types d'utilisateurs, un expert de la base de donnée d'une part, l'utilisateur final d'autre part. Le système construit sa connaissance par interaction avec l'expert de la base de données. L'essentiel de l'interprétation dépend des directives de cet expert. Nous adoptons le même principe.

Le système TEAM est une excellente référence dans l'approche qui consiste en une analyse syntaxique complète, suivie d'analyses sémantiques sur les arbres obtenus. Toutes ces opérations sont réalisées dans un formalisme du type logique du 1er ordre. Or, une fois isolés les constituants sémantiques pertinents dans la question, il apparaît que les trois quarts des nœuds de l'arbre syntaxique ne représentent que la «glu syntaxique» (op. cit., p. 203) de la phrase originelle. Ces nœuds sont pourtant pris en compte dans toute la première phase d'analyse, ce qui nuit à l'efficacité. Un autre exemple axé sur la logique est décrit dans [Péquegnat et al., 1988] : pas moins de cinq bases de connaissances y sont constituées, représentations logiques des données linguistiques, conceptuelles (la base de données), sémantiques. Un système complet de réécriture est nécessaire pour chaque passage d'une représentation à une autre… Les traitements en sont alourdis d'autant. Nous ne passons pas par une formalisation logique, bien que la formulation des opérateurs linguistiques s'en rapproche quelque peu.

A la fin des années 80, les développements liés à l'interrogation de bases de données en langue naturelle se sont, nous semble-t-il, scindés en deux : d'une part, l'approfondissement d'outils d'analyse du langage naturel (pour répondre en particulier au problème de la portabilité) et, d'autre part, l'étude du dialogue homme - machine. Nous revenons peut-être sur des idées anciennes, mais c'est pour les aborder avec des outils différents, linguistiques et informatiques.

Outils linguistiques

En ce qui concerne le traitement linguistique, le système est caractérisé par :

- La construction interactive, avec l'administrateur de la base, d'un dictionnaire électronique de mots clés, qui exploite de façon automatique son schéma et son contenu. Celui-ci contiendra donc, entre autre, toutes les instances.
- La définition et l'utilisation d'opérateurs linguistiques, plutôt que des opérateurs de la logique : c'est là une application de la théorie proposée par Z. S. Harris [Harris, 1968, 1976] et décrite, pour le français, par de nombreux travaux [Gross, 1975], [Boons et al., 1976]. Nous intégrons au lexique une description exhaustive des opérateurs de la langue pour le domaine considéré, en associant aux arguments des

prédicats une information qui se rapproche du concept de classe d'objet [Le Pesant, Mathieu-Colas, 1998].
- Une interprétation de la question par analyse pré-syntaxique : pour cela, on extrait de la question les informations linguistiques pertinentes, en s'aidant du dictionnaire [Maurel, 1991, Maurel et Mohri, 1995].

Ces trois points correspondent à un système de mots clés complet, enrichi par le codage et l'exploitation des opérateurs linguistiques du domaine.

Outils informatiques

Les avantages liés à la représentation des données par un transducteur, déjà recommandée par [Gross et Perrin, 1989], ne sont plus une nouveauté dans le cadre du traitement du langage naturel [Roche et Schabes, 1997]. C'est grâce aux progrès algorithmiques de la théorie des automates à nombre fini d'états que nous pouvons utiliser un dictionnaire aussi complet, et donc aussi volumineux. Notre dictionnaire sera, en fait, un transducteur minimal. Celui-ci sera construit directement par l'algorithme de [Stoyan, 1999], adapté aux transducteurs, en suivant le principe de placement des sorties défini par [Mohri, 1994] ; puis il sera compacté sous la forme d'un tableau de taille à peu près égale au nombre de transitions [Liang, 1983].

En ce qui nous concerne, nous mettons en œuvre une « cascade de transducteurs » :
- le premier transducteur est notre dictionnaire : il extrait de la question les informations pertinentes et les remplacent par ses propres codes
- un deuxième transducteur, qui part donc du résultat du premier, traite les opérateurs linguistiques reconnus,
- puis des transducteurs « SQL » génèrent finalement une requête à partir des informations produites par les deux premiers.

En conséquence, plutôt qu'une description formelle de la phrase à partir de règles syntaxiques, puis sémantiques, nous associons aux mots l'information issue de la base de données, avec les traitements afférents. Plus précisément, ces traitements sont associés aux opérateurs linguistiques par l'administrateur de la base de données lors de l'installation du système sur la base.

Dans la section 2 de cette communication, nous revenons sur le principe général du système, ainsi que sur les transducteurs utilisés ; dans la section 3, nous précisons le contenu du dictionnaire, et, en particulier, le codage des opérateurs linguistiques. Nous concluons en section 4. Afin d'unifier la présentation, tous les exemples ont trait à la même base de données, brièvement décrite en Annexe.

2. Le système : une cascade de transducteurs

2.1. Présentation générale

Nous donnons Figure 1 un aperçu de l'application, laquelle se divise en deux classes d'outils : ceux qui « installent » le module d'interrogation sur la base de donnée (partie a de la Figure 1) et ceux qui réalisent l'interrogation (partie b).

L'utilitaire pour la construction du dictionnaire exploite la structure et le contenu de la base de données, ainsi que les bibliothèques d'opérateurs linguistiques ; il fait également appel à des dictionnaires de synonymes. Le transducteur généré est ensuite compacté.

En phase de fonctionnement, tant que la session de questions n'est pas terminée, il y a :
- lecture de la question,
- exécution de la cascade de transducteurs,
- interrogation de la base et présentation du résultat.

Les deux phases bien distinctes, le recours à un connaisseur de la base de données, tout cela se rapproche du système TEAM, cependant nos questions à l'administrateur de la base de données sont plus simples (plus techniques en ce qui concerne la base, mais moins linguistiques) et les traitements sur la question sont différents.

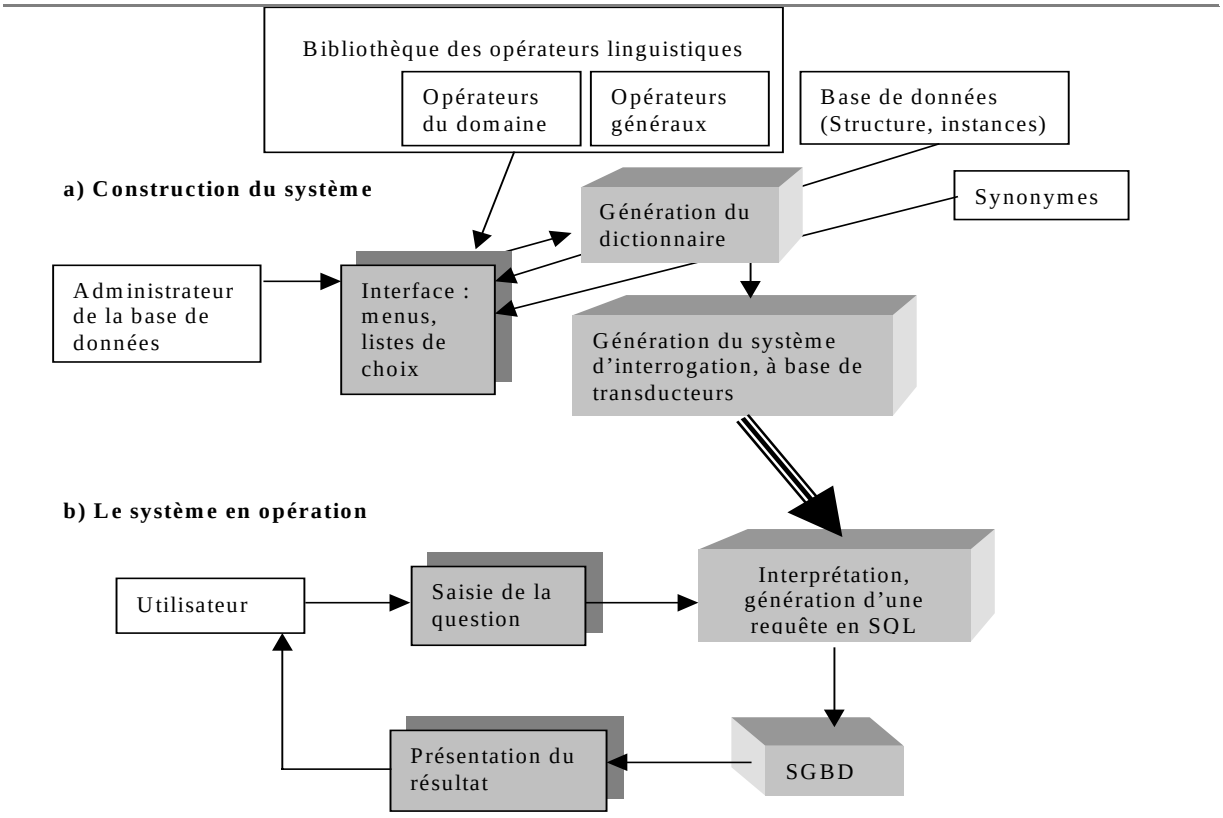

Figure 1 : Présentation générale

2.2. Les transducteurs

Comme nous l'avons dit en introduction, le premier transducteur, qui opère à partir de la question, est une représentation compacte de la base de données (les noms des tables et des attributs, ainsi que les instances, le tout accompagné de synonymes éventuels) et des opérateurs linguistiques que nous comptons mettre en œuvre.

Prenons pour exemple la base des meubles, dont une partie est donnée Figure 3 (en annexe). La Figure 2 présente un petit extrait simplifié de dictionnaire. Les noms des tables sont remplacés par les attributs qui représentent les tables elles-mêmes. Ces attributs particuliers (soulignés, ici) seront indiqués au système par l'administrateur de cette base, lors de l'installation du logiciel. Nous associons donc à *salon* un attribut (*A.*) de code *A.Canapés.*<u>*Nom*</u>. De même, l'information associée à l'entrée *Bahia* se lit de la façon suivante : il s'agit d'une instance (*I.*) de l'attribut <u>*Nom*</u> de la table *Canapés* et sa valeur est *Bahia*.

La Figure 2 montre, dans sa partie inférieure, le transducteur correspondant : les ronds représentent les états et les lettres étiquettent les transitions. Un mot est reconnu lorsque son parcours correspond à un parcours des états du transducteur et se termine sur un état final (rond noir). Le code lié au mot est généré par le transducteur au cours de la reconnaissance (ce sont les sorties, en gras sur la figure).

Bahia →I.Canapés.Nom.Bahia

Baie →I.Canapés.Couleur.Baie

salon →A.Canapés.Nom

salons →A.Canapés.Nom

séjour →A.Canapés.Nom

séjours →A.Canapés.Nom

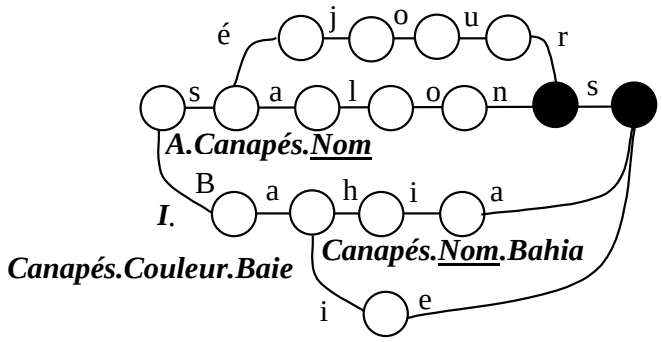

Figure 2 : Un exemple de représentation d'un dictionnaire par un transducteur

Prenons comme exemple la question : *Quels sont les salons de couleur Baie ?*. Elle contient un synonyme d'un nom de table, *salons*, un opérateur linguistique (voir § 3), la préposition *de*, un nom d'attribut présent dans plusieurs tables, *couleur* et, enfin, une instance, *Bahia*. Tous ces mots sont reconnus par le transducteur qui leur associe leur code :

salons → A.Canapés.Nom
de → DE
couleur → A.Canapés.Couleur | A.Lits.Couleur | A.Rideaux.Couleur | …
Baie → I.Canapés.Couleur.Baie | I.Rideaux.Couleur.Baie

Le premier transducteur génère un ensemble de codes à partir de la question. Chaque code est la « définition » d'un mot reconnu. Nous avons donc un système de reconnaissance de mots-clés, à ceci près que l'information associée à ces mots va permettre une analyse linguistique, grâce à la reconnaissance d'opérateurs.

Le deuxième transducteur associe aux opérateurs linguistiques ses arguments et transforme l'ensemble en un seul code (voir partie 3). Dans notre exemple, on obtiendrait seulement, au final :

DE(A.Canapés.Nom, I.Canapés.Couleur.Baie)

Le travail de codage des opérateurs linguistiques trouve ici son application et permet de générer des informations exploitables par les transducteurs de la dernière étape.

Les derniers transducteurs opèrent à partir du résultat du deuxième, générant chacun une partie de la requête SQL : en général, il y a une partie SELECT, une partie FROM et une partie WHERE [Bowman & al., 1996]. La partie SELECT comprend les attributs des tables identifiés dans la question (certains sont ajoutés systématiquement selon le contexte, en particulier le nom). La partie FROM comporte toutes les tables rencontrées dans la question. La composante WHERE est construite sur la base des opérateurs. Pour l'exemple des canapés de couleur Baie, on obtiendrait :

SELECT Produits.RéfProduit, Produits.Nom,
FROM Canapés INNER JOIN Produits ON Canapés.RéfProduit = Produits.RéfProduit
WHERE (((Canapés.Couleur) Like "Baie"))

3. La constitution du dictionnaire : les opérateurs linguistiques

La présentation rapide que nous venons de faire du système montre l'importance des codes associés aux mots dans le dictionnaire : là se situe toute l'information nécessaire à la génération d'une requête SQL pertinente. Nous indiquons dans un premier temps comment sont représentés et utilisés les opérateurs linguistiques, puis nous décrivons le processus de génération du dictionnaire.

3.1. L'exemple de la préposition de

Dans l'exemple ci-dessus (*Quels sont les salons de couleur Baie ?*), la préposition de correspond à l'opérateur que nous avons appelé *DE*. Cet opérateur *DE* possède deux arguments, qui sont les attributs (éventuellement instanciés) d'une même table[i]. L'un de ces deux attributs doit correspondre à l'attribut représentant la table, ce qui est le cas ici.

Un seul de ces attributs soulignés étant présent ici, le deuxième transducteur choisit la table *Canapés* et non les autres tables qui ne sont présentes que d'un seul côté de l'opérateur. On obtient donc :

salons de couleur → DE(A.Canapés.Nom, A.Canapés.Couleur)
Baie → I.Canapés.Couleur.Baie | I.Rideaux.Couleur.Baie

Puis ce même transducteur reprend le résultat obtenu (dans la "cascade", le transducteur 2 est utilisé plusieurs fois) et instancie la *couleur* à *Baie*, tout en éliminant la référence à la table des *rideaux* :

salons de couleur Baie → DE(A.Canapés.Nom, I.Canapés.Couleur.Baie)

Lors de l'installation, l'administrateur a entré la formule SQL correspondant à ce genre de requête. C'est alors que les transducteurs SQL ont été construit. Ils vont pouvoir maintenant fournir le résultat présenté à la fin du § 2.

3.2. Les opérateurs linguistiques

Les connaissances linguistiques sont codées dans des bibliothèques qui seront réalisées en collaboration avec Barbara Kaltz, de l'équipe de recherche en linguistique de Tours, à la fois en français et en allemand, ceci pour éviter des réponses trop *ad hoc*… Quelques exemples détaillés sont donnés dans [Bouchou et al., 1999]. On distingue deux domaines :

- le domaine général : les groupes numériques, les groupes nominaux, les verbes "*être*", "*avoir*", les notions de comparaison ("*plus*", "*moins*"), …
- le domaine contextuel : ainsi pour la base de données sur le magasin de meubles, on aura besoin d'opérateurs comme "ACHAT", lié aux mots "achats", "acheter", "achetés", "acquisition", "acquisitions", "acquérir", "approvisionner", …

Les opérateurs représentent donc les différentes flexions des prédicats de la question (verbes, noms, adjectifs). Leur recensement permet, lors de l'installation du module sur la base, de poser à l'administrateur chargé de cette installation, qui connaît bien sa base de données, des questions très précises. Il lui est demandé *grosso modo* d'indiquer la partie de la base où se trouvent les informations associées aux opérandes, et leur méthode de calcul si nécessaire.

Par exemple, prenons l'opérateur *STOCK*, qui n'a qu'un opérande dans notre contexte (le stock courant). En français, il correspond à différents prédicats, des verbes :

Les salons de couleur Baie ont été stockés
Il reste des salons de couleur Baie

Ou des noms prédicatifs, associés à leurs verbes supports appropriés :

Les salons de couleur Baie sont en stock
Les salons de couleur Baie sont en réserve
Il reste en stock des salons de couleur Baie
Il reste en réserve des salons de couleur Baie

---

[i] Signalons que la préposition *de* peut aussi sélectionner un autre opérateur, que nous avons appelé *DANS* et qui est défini par des jointures entre tables, lors de l'installation. Par exemple ici, la question *Quelles sont les commandes de Dupont ?*. Voir à ce propos [Bouchou et al., 1999].

Son code dans la bibliothèque est le suivant :

STOCK(quoi?)

Dans le contexte de notre base de meubles, l'interaction avec l'administrateur de la base permet alors de construire l'association suivante (à l'intention du transducteur 2) :

quoi?≡ I.Produits.RéfProduits/Σ (+[UnitésReçues])- Σ (+[UnitésVendues])- Σ (+[PertesUnités])

De cette manière on sait que le stock concerne une référence de produit, et qu'il est calculé par la formule donnée.

Il peut arriver que la base ne permette pas de répondre à certaines des questions associées à l'opérateur en bibliothèque. Par exemple, s'il n'y a pas de notion de date dans la base, on ne peut pas interroger sur l'opérande *quand?* de l'opérateur *COMMANDE*. Dans ce cas l'administrateur n'indique aucun élément (ni attribut, ni formule de calcul), et si cette « dimension » est questionnée au cours de l'utilisation, le système répondra qu'aucune réponse n'est disponible.

Dans le même ordre d'idées, l'opérateur *COMMANDE* représente un exemple d'ambiguïté dans le contexte. Lorsqu'on demande : *a-t-on commandé des canapés bleus ?*, il est possible que l'on veuille savoir si des canapés bleus sont en cours de commande et vont arriver. Il est également possible que l'on veuille un bilan des commandes de canapés bleus, réceptionnées ou non. De tels cas se repèrent dans la phase de constitution des bibliothèques, grâce aux lexiques linguistiques du domaine. Le système peut alors construire une réponse pour chaque interprétation signalée.

3.3. Création du dictionnaire

Le dictionnaire électronique pour l'interprétation des questions posées à la base de données est construit à partir de cinq sources :

- le contenu de la base : les enregistrements
- le schéma (structure) de la base : tables, attributs, etc.
- des dictionnaires de synonymes
- le codage précédemment décrit des opérateurs généraux et du domaine
- une connaissance pointue de la base de données et de son utilisation courante (connaissance qu'a l'administrateur)

Dans un premier temps, chaque enregistrement de la base est copié dans le dictionnaire, le code associé indiquant qu'il s'agit d'une instance de tel champ, de telle table (cf. "Bahia" et "Baie" dans le petit dictionnaire de la Figure 2). A la fin de cette phase, le dictionnaire renferme une copie de chaque donnée de la base.

Pour les autres sources, l'alimentation du dictionnaire passe par une interaction avec l'administrateur de la base de données. Par exemple, en ce qui concerne la structure de la base, nous développons un utilitaire qui extrait les données relatives à la structure de la base à partir du catalogue et interroge l'administrateur pour leur associer les informations utiles pour l'interprétation de la question. Cet utilitaire ajoute également des synonymes (après avis de l'administrateur).

Nous l'avons vu, les codes distinguent les tables, les attributs, les instances, les chiffres, les opérateurs, etc.

Par exemple, le fait qu'il existe une table désignée par *Canapés* dans la base sera traduit par une entrée canapé dans le dictionnaire, associée au code *A.Canapés.*<u>*Nom*</u>, lequel s'interprète de la façon suivante : « ce mot correspond au nom d'une table de la base de donnée, la table *Canapés* ; celle-ci a pour attribut principal l'attribut <u>*Nom*</u> ». Si

l'administrateur juge que le mot *séjour* est un synonyme de *canapé* dans le contexte d'utilisation de sa base, alors *séjour* sera associé au même code.

Le principe est le même pour les opérateurs linguistiques liés au contexte. Pour un opérateur donné (issu de la bibliothèque des opérateurs linguistiques), l'administrateur de la base devra donner, lorsqu'elles existent, les formules de calcul de chaque opérande.

Cette étape est cruciale pour la qualité de l'interprétation des questions. L'administrateur de la base de données doit utiliser sa connaissance de la base et de son contexte, la connaissance qu'il a des besoins des utilisateurs (c'est important) et la connaissance normale qu'un locuteur a de sa langue. Il n'a pas à être un expert des méthodes et outils de l'IA.

4. Conclusion

Pour se situer par rapport aux techniques logiques classiques, disons que le premier transducteur effectue une reconnaissance qui se situe au niveau d'une « présyntaxe ». Il détecte les éléments significatifs de la question. L'analyse porte alors sur cette sélection, et non sur un arbre syntaxique complet de la phrase. Cette analyse s'appuie sur une définition linguistique générale des « prédicats » de la question, définition qui a été particularisée, au moment de l'installation sur la base de données, pour correspondre exactement au contexte, et devenir ainsi complètement opératoire.

L'approche que nous développons évite une série d'opérations sur des arbres syntaxico-sémantiques. On peut voir la syntaxe comme une définition en compréhension du langage, alors que le transducteur offre une définition extensive des mots d'une question : tout ce qui est nécessaire à l'interprétation doit y être. C'est là l'apport essentiel des transducteurs à nombre fini d'états : représenter une masse importante d'information sous une forme suffisamment condensée pour être exploitable efficacement.

La souplesse, et en particulier la « portabilité » du système (c'est-à-dire son adaptabilité à toute base de données) est satisfaisante dans la mesure où tout se passe à la construction du dictionnaire. A l'intérieur d'un même domaine, le passage d'une base de données à l'autre ne pose pas de difficulté, puisque la structure et le contenu de la nouvelle base permettent d'alimenter la partie à modifier dans le dictionnaire. De même, les évolutions de la base de données sont prises en compte par reconstruction de la partie du dictionnaire concernée, puis par régénération des transducteurs.

L'application que nous proposons reste néanmoins relative à un domaine (par exemple le tourisme, la vente par correspondance, la gestion de stocks, …). Le passage d'un domaine à l'autre nécessite la constitution de la bibliothèque d'opérateurs linguistiques liés au domaine, lorsqu'elle n'existe pas encore. Mais une fois construite, cette bibliothèque sert pour toutes les bases de données du domaine considéré.

Le premier développement lié aux idées présentées ici avait un but spécifique : l'interrogation en langue naturelle d'une base de données spatiales sur des informations touristiques Tourangelles [Thomas et al. 1997]. Depuis, une partie des résultats a été reprise et généralisée pour créer un prototype dont nous testons actuellement la portabilité sur plusieurs exemples de bases de données.

Les développements actuels sont réalisés en Java. Nos efforts portent principalement sur l'enrichissement de la bibliothèque des opérateurs linguistiques (travail linguistique), l'amélioration de l'interface de construction du dictionnaire, l'optimisation des transducteurs.

Références

Boons J. P., Guillet A., Leclère C. (1976a), *La structure des phrases simples en français. I. Constructions intransitives*, Genève, Droz.

Boons J. P., Guillet A., Leclère C. (1976b), *La structure des phrases simples en français. II. Constructions transitives*, Paris, Rapport de recherche du LADL n°6.

Bouchou B., Maurel D., Kaltz B. (1999), Prédicats logiques / prédicats linguistiques pour la consultation de base de données en langue naturelle, *Revue Informatique et Statistique dans les Sciences Humaines*, à paraître.

Bowman J., Emerson S., Darnowsky M., *The practical Sql handbook : using standard query language*, Addison Wesley, 1996.

Clifford J. (1988), Natural Language Querying of Historical Databases, *Computational Linguistics*, Vol. 14, n° 4, pp. 10-34.

Green B.F., Wolf A.K., Chomsky C., Laughery K. (1960), Baseball : an automatic question-answerer, *Computers and tought*, Mc Graw Hil, New-York, pp. 207-216.

Gross G. (1998), Pour une véritable fonction synonymie dans un traitement de texte, *Langages*, 1998, n°131.

Gross M., Perrin D. (1989), Electronic Dictionaries and Automata in Computational Linguistics, *Lecture Notes in Computer Science*, 377, Springer.

Gross M. (1975), *Méthodes en syntaxe*, Hermann, Paris.

Grosz B., Appelt D., Martin P., Peirera F.(1987), TEAM : an experiment in the design of transportable natural language interfaces, *Artificial Intelligence 32*, pp 173-243.

Harris Z. S. (1968), *Mathematical Structures of Language*, Interscience Publishers.

Harris Z.S. (1976), *Notes du cours de syntaxe*, Paris, Le Seuil.

Kaplan S. J. (1984), Designing a Portable Natural Language Database Query System, *ACM Transactions on Database Systems*, vol. 9, n°1, march 1984, pp. 1-19.

Le Pesant D., Mathieu-Colas M. (1998), Introduction aux classes d'objets, *Langages*, 1998, n°131.

Liang F. M. (1983), *Word Hyphenation by Computer*, PhD Thesis, Computer Science Departement, Standford University, Research Report STAN-CS-83-977.

Maurel D., Mohri M. (1995), Computation of French Temporal Expressions to query database, *First International Workshop on Applications of Natural Language to Data Bases (NLDB'95)* (Actes p. 71-80).

Maurel D. (1991), Préanalyse des adverbes de date du français, *TA information*, volume 32, n°2, p. 5-17.

Mohri M. (1994), Minimization of Sequential Transducers, *Theorical Computer Science*.

Péquegnat C., Aguirre J.L. (1988) , Interface en langue naturelle pour l'interrogation d'une base de données relationnelle, *Systèmes experts et applications*, pp 441-460.

Roche E., Schabes Y. ed. (1997), *Finite state language Processing*, Cambridge, Mass./London, England: MIT Press.

Sabah G. (1997), Le sens dans le traitement automatique des langues, *T.A.L.*, 1997, vol. 38, n°2, pp. 91-133.

Stoyan M. (1999), Direct Construction of Minimal Acyclic Finite States Automata, *Annuaire de l'Université de Sofia St. Kl. Ohridski*, Faculté de Mathématiques et Informatique, volume 92, livre 2.

Thomas N., Maurel D., Trépied C. (1997), *Interrogation en langage naturel d'une base de données spatiales*, Rapport interne du Laboratoire d'Informatique de l'Université de Tours, n°190, 10 p.

Annexe

La base de données dont sont issus les exemples présentés au fil de cette communication a pour objet la gestion d'un magasin de meubles. Les principales tables, leurs attributs et leurs jointures sont dans la Figure 3 ci-dessous.

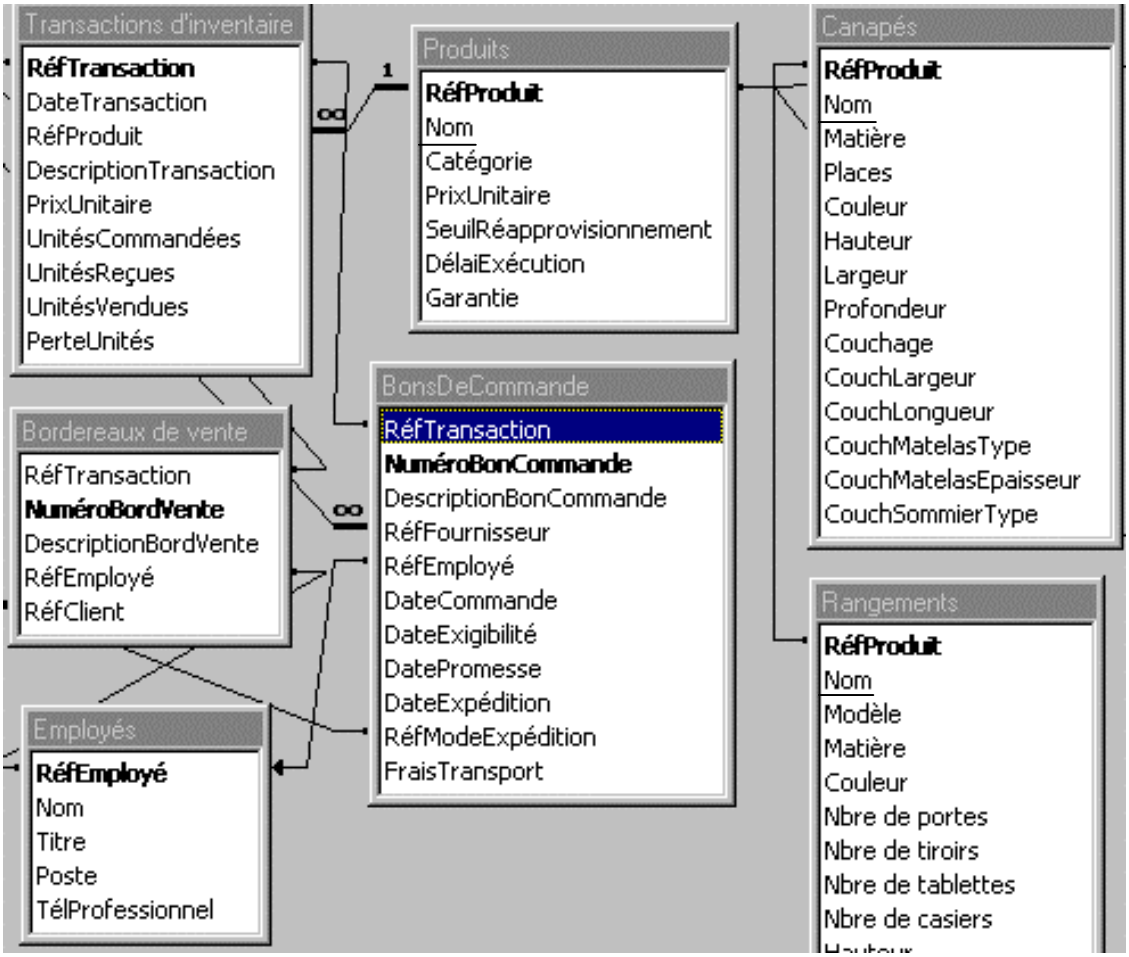

Figure 3 : extrait des tables et jointures de la base des meubles

Les questions typiquement posées à cette base concernent les caractéristiques des meubles (questions dites « de catalogue ») et le stock ou les commandes en cours. Elles sont émises par les vendeurs pour répondre aux clients du magasin. Par exemple,

"*qu'avons-nous comme canapés bleus* ?",

"*qui a commandé le produit de référence XXX* ?", etc.

Plus épisodiquement, des opérations de marketing (table des clients), des calculs de primes salariales (au pourcentage des ventes), diverses analyses et autres bilans peuvent justifier d'autres types d'interrogation, également pris en compte dans le système d'interface en langue naturelle.