Quel lexique pour un traitement automatique de la référence ?

J.-M. Pierrel, B. Gaiffe
& L. Romary
CRIN-CNRS & INRIA Lorraine

Les systèmes de dialogues oraux homme-machine, finalité des recherches que nous menons au sein de notre équipe, nécessitent d'aborder à la fois des problèmes de reconnaissance automatique, de compréhension et d'interprétation d'énoncés. Dans ce cadre, lexique et référence constituent deux des points fondamentaux de nos études. Le lexique, en tant que système dans lequel sont représentées toutes les informations concernant les mots du langage, est une composante aussi incontournable que peuvent l'être les mots dans un processus de reconnaissance de phrases (Segui 1989; Pierrel 1989). Il en est de même de la référence car s'il est vrai qu'une approche finalisée nous permet souvent de restreindre, voire de contourner certains problèmes linguistiques, une telle approche nécessite pour le moins d'être capable de faire la liaison entre des expressions de langue et les objets ou fonctions de la tâche gérée.

Nous nous proposons donc dans cet article, après une rapide introduction, de dresser un bilan critique des représentations lexicales dans notre domaine, de présenter notre approche de la référence et de discuter quelques propositions tendant à répondre à cette question choisie pour titre de notre contribution "Quel lexique pour le traitement automatique de la référence ?" ou "Comment concilier les aspects linguistiques structuraux nécessaires à la reconnaissance de phrases et les aspects pragmatiques prépondérants dans toute utilisation finalisée du langage ?".

1. Importance du problème en compréhension automatique de dialogues oraux homme-machine

Un rapide parcours des diverses approches dominantes dans notre domaine (Haton 1980; Pérennou 1987; Calliope 1989; Haton 1991) montre que le lexique en reconnaissance de la parole occupe une position clé lui assurant un

statut privilégié comme passage obligé à la fois lorsque l'on construit des mots candidats, à partir des données acoustico-phonétiques et phonologiques (processus ascendant) et lorsque l'on émet des mots en fonction de données linguistiques et contextuelles (processus descendant) (Haton 1985).

De plus, que cela soit au niveau infra-lexical, mais surtout au niveau supra-lexical (syntaxique et sémantique), la plupart des systèmes existants, fondés sur un processus général d'hypothèse et test, utilisent des lexiques hiérarchisés et/ou des lexiques à base de traits sémantiques pour permettre aisément une prédiction lexicale en fonction du contexte, réduit le plus souvent au seul contexte linguistique déjà reconnu ou traité. Cela induit donc une stratégie de reconnaissance et privilégie nettement les approches de type focalisation lexicale (Romary 1989b).

Ce type d'approche se caractérise en fait par :

(a) une séparation trop nette entre la reconnaissance et l'interprétation contextuelle

Dans la plupart des systèmes, on voit ainsi apparaître une structure par trop séquentielle entre deux grandes fonctionnalités :

- la reconnaissance de phrases et de leurs structures
- l'interprétation pragmatique

(b) une dichotomie trop importante entre le lexique syntaxico-sémantique et la structure d'objets (de référents ?) de la tâche

Les lexiques demeurent essentiellement syntaxiques et sémantiques[1] et ni la liaison entre les lexèmes et les objets de la tâche ni la prise en compte du contexte ne sont facilement modélisables. Qui plus est, la limitation du lexique aux seuls mots porteurs de sens ne permet pas de traitements fins des modalités et de la référence. Cela induit en particulier pour les groupes nominaux des traitements équivalents pour des énoncés pourtant nettement différenciés tels que[2] :

(1) "ouvre *la* fenêtre"
(2) "ouvre *cette* fenêtre"
(3) "ouvre *une* fenêtre"

---

[1] On doit alors comprendre le terme de sémantique comme un ensemble de contraintes permettant de restreindre les hypothèses fournies par la reconnaissance.

[2] Dans le paradigme désormais classique d'un environnement informatique à base de menus, fenêtres, icones etc.

En faisant ainsi l'impasse complète sur l'analyse des déterminants, on définit des systèmes produisant pour ces trois énoncés une interprétation unique du type :

OUVRIR (FENETRE)

qui se caractérise par une absence de traitement de la référence.

Or, nous sommes convaincus qu'une avancée notable dans le domaine de la compréhension et la gestion de dialogues finalisés en langue naturelle passe par la mise en oeuvre d'un traitement précis et efficace de la référence. En effet, vu du côté de l'informaticien, l'aspect finalisé se traduit *in fine* par la mise en oeuvre de fonctions sur les objets de la tâche traitée et nécessite donc de résoudre les problèmes de référence entre les expressions langagières et les objets et fonctions de la tâche.

Pourtant, et ce n'est pas le moindre des paradoxes, la stratégie sous-jacente à ces systèmes découle assez naturellement des spécifications de départ de nos études : réaliser des systèmes finalisés adaptés à des experts du domaine de l'application. Dans ce cadre, il était naturel de penser que les formulations langagières spontanées de l'expert suivraient assez rigoureusement la structure, que l'on qualifierait maladroitement de séquentielle, de la tâche. La composante pragmatique se trouvait donc limitée exclusivement à la prise en compte des contraintes de la tâche et n'avait pas à traiter des aspects liés aux intentions de l'utilisateur qui, du moins c'était l'hypothèse sous-jacente, n'exprimait que des énoncés exécutables par une commande élémentaire de la tâche.

Ainsi, une expression du type

(4) "crée une fenêtre verte"

ne pouvait intervenir que s'il existait une commande de la tâche rigoureusement équivalente (i.e. "création d'une fenêtre verte...").

Il n'y avait pas lieu dans ce cas de se poser des problèmes de planification et donc de mettre en oeuvre de véritables raisonnements pragmatiques. Comme nous le montre le système PARTNER (Morin 1987), cela a permis d'aboutir à de premières réalisations où l'on pouvait projeter l'ensemble des contraintes pragmatiques dans la structure du langage et donc réduire la phase d'interprétation à une simple mise en correspondance ou "matching" entre les constituants de l'énoncé et les fonctions et objets de la tâche.

Mais très vite, nous nous sommes rendus compte qu'il y avait là de multiples biais; le premier et le plus grave étant sans aucun doute de présupposer une adéquation parfaite entre les énoncés langagiers et les structures de la tâche. L'intérêt d'une interface en langage naturel n'est-il pas précisément de permettre à l'utilisateur de se focaliser sur ses objectifs (intentions) et de s'abstraire de la structure souvent trop contraignante de la tâche ? Ainsi l'énoncé (4) devrait plutôt être interprété comme la spécification de l'objet (le référent) final recherché (une fenêtre verte) sans se soucier des procédures à mettre en oeuvre pour y parvenir qui dans ce cas peuvent correspondre à la mise en oeuvre de deux fonctions élémentaires

CREATION (FENETRE)
CHANGEMENT-COULEUR (FENETRE, VERT)

Sans insister sur les aspects de planification qui sortent du champ de cette contribution, on voit là l'importance du traitement de la référence et du raisonnement pragmatique nécessaires à l'interprétation d'un énoncé fort simple au premier abord.

2. Approches traditionnelles concernant le lexique

2.1 Les lexiques dans des micro-domaines

Dans l'état actuel des connaissances, les systèmes de dialogue qui fonctionnent ne portent que sur des univers référentiels, et donc sémantiques, réduits. C'est ce que nous entendons par micro-domaines. En pratique, les applications correspondantes sont de type commande. Autrement dit, les énoncés de l'utilisateur se ramènent à une interprétation de la forme :

[commande à effectuer, arguments de cette commande].

Par ailleurs, on présuppose que l'utilisateur connaît les bases de fonctionnement de l'application et que, par conséquent, il ne produira que des énoncés correspondant à des actions faisables dans le contexte de celle-ci.

Dans un tel cadre, et tenant compte des contraintes que nous venons d'exposer brièvement, il semble naturel d'utiliser les informations venant de la description de l'application pour contraindre la description linguistique. Ainsi, étant donné que l'on souhaite arriver en fin de compte à une fonction de la tâche, on peut utiliser les contraintes de type (au sens informatique du terme) que cette fonction impose à ces arguments. Par exemple, si le micro-domaine considéré consiste en des cubes et une pince de robot, la fonction PRENDRE-

DANS-LA-PINCE impose à son argument d'être du type CUBE. Au niveau "linguistique", cela signifie que si la partie prédicative d'un énoncé conduit à cette fonction, l'interprétation référentielle de son argument devra être de type CUBE[3]. On peut donc globalement projeter la description de la tâche sur les règles de grammaire et obtenir ce qu'on appelle dans le domaine du dialogue homme-machine une grammaire sémantique.

La position la plus extrême en matière de telles grammaires consiste à ne même plus avoir de catégories grammaticales mais seulement des types pour l'application.

Un exemple en serait :

```
commande-prendre   ->   verbe-prendre désignation-cube
verbe-prendre      ->   'prends'
                        'attrape'
désignation-cube   ->   Det 'cube' localisation-cube
                        Anaphore-cube
Anaphore-cube      ->   'le'
etc...
```

Dans la mesure où l'on projette les types de l'application sur la description du langage, la résolution de la référence est grandement facilitée : on filtre directement les objets de la tâche par les types récupérés au cours de l'analyse syntaxique des énoncés[4]. De plus, l'analyse syntaxique est fortement contrainte (le facteur de branchement sur les noms communs devient extrêmement faible) et permet par conséquent de guider efficacement la reconnaissance de parole.

Une telle approche n'est cependant envisageable que dans des micro-domaines. Dans un contexte plus vaste, la multiplication des prédicats conduit immédiatement à une multiplication des types d'objets. On ne tient donc absolument pas compte de la factorisation opérée par le langage en fonction de types similaires, c'est-à-dire de la notion de catégorie considérée alors indépendamment du prédicat.

---

[3] Et ceci quel que soit le nom effectif utilisé pour faire référence à l'objet en question.

[4] Aux anaphores près. Dans le cas des pronoms, par exemple, on filtre parmi les représentations des énoncés précédents un objet du type demandé par l'anaphorique. On a en effet virtuellement un pronom anaphorique par type d'objet.

2.2 Le lexique dans les approches casuelles

Comme nous l'avons vu précédemment, lorsque l'on s'intéresse à des domaines plus vastes, du type renseignements administratifs (Roussanaly 1992), la démarche consistant à projeter la description de la tâche sur le lexique et la grammaire ne peut être conservée. D'une part le domaine est beaucoup trop vaste, et d'autre part, ce type de système se voulant "grand public", on ne peut plus faire l'hypothèse que l'utilisateur est un expert de la tâche (ce qui serait d'ailleurs paradoxal dans un système d'aide).

Il est donc nécessaire de revenir à une description véritablement linguistique du sous-langage correspondant à la tâche. Autrement dit, au lieu de définir a priori ce sous-langage en fonction de ce que peut faire la tâche, on ne préjuge que de son existence et on espère, en étudiant des corpus, aboutir à une description linguistique d'un sous-ensemble simplifié de la langue naturelle qui se trouve en intersection avec lui. Dans le cas particulier des renseignements administratifs (l'équivalent des pages roses des annuaires) cette étude a été faite par Deville (1987, 1989) et s'est appuyée sur une description casuelle.

Nous allons dans un premier temps présenter brièvement cette description, puis montrer les problèmes qu'une telle approche nous pose lorsque l'on souhaite résoudre les références dans des énoncés, même dans le cas où ceux-ci présentent une structure générale simple.

Implémenter sur machine une grammaire à composante casuelle suppose de définir la structure attendue par chacun des prédicats du langage et ce sous deux points de vue. La première description est d'ordre "structurel" : pour chacun des cas d'un prédicat donné, on a un ensemble de prépositions possibles. Une partie de la grammaire apparaît donc au niveau lexical même. L'autre information nécessaire est d'ordre "sémantique" : un terme construit par la syntaxe ne peut correspondre à un cas donné que s'il accepte de porter un trait sémantique fixé par le prédicat. En pratique, on doit donc représenter pour chaque prédicat l'ensemble des traits sémantiques qu'il impose à chacun de ses cas, et pour chaque nom commun susceptible d'être la tête d'un groupe nominal l'ensemble des traits qu'il accepte. En général, pour des raisons d'ordre pratique[5], on considère des traits à valeur ternaire (+, - ou 0), permettant de tenir compte des cas où aucune contrainte n'est imposée pour un prédicat ou un cas donné.

---

[5] Cela évite de lister in extenso l'ensemble des traits qu'accepte un nom donné : on se contente de détecter les conflits entre + et -.

L'étude faite par Guy Deville a permis de définir des classes de prédicats du point de vue de leur comportement casuel ce qui factorise d'autant les informations à représenter dans le lexique. Voici tout d'abord les classes de prédicats obtenues :

CTR (Control) : le prédicat fait référence à une action ou à un état impliquant un contrôle explicite;
DYN (Dynamic) : le prédicat entraîne un changement de configuration du monde de l'application;
ANI (Animate) : le prédicat implique la présence d'une entité animée;
MVT (Movement) : le prédicat implique qu'une entité est en déplacement;
TRS (Transitive) : le prédicat nécessite au moins deux arguments;
PRO (Production) : le prédicat exprime le transfert d'une entité de celui qui contrôle l'action vers une autre entité;
SPA (Space) : la primitive nécessite l'adjonction d'une expression spatiale.

Verb Type	Verb Primitive	CTR	DYN	ANI	MVT	TRS	PRO	SPA	Example
ACT	MVT1	+	+	+	+	-			to go
	MVT2	+	+	+	+	+			to bring
	ACTANIME	+	+	+	-	+			to vaccinate
	ACTOBJET	+	+	+	-	+			to sign
	ECHPROD	+	+	+	-	+	+		to give
	ECHOBT	+	+	+	-	+	-		to obtain
	ATRANS	+	+	+	-	-			to vote
STATE	LOCATION1	-	-		-	-		+	to find o.s.
	LOCATION2	+	-	+	-	+		+	to keep
	EXTENSION	+	-	+	-	+		-	to know
	STATUT1	-	-	+	-	-		-	to be french
	STATUT2	-	-	-	-	-			to be valid
	MESURE1	-	-	+	-	-		-	to be old
	MESURE2	-	-	-	-				to cost
PROCESS	PROCESS1	-	+	+	-	+			to change
	PROCESS2	-	+	-	-				to be destroyed

Table 1.

Les cas primitifs

Les principaux cas impliqués dans la définition des primitives sont décrits à l'aide des restrictions sémantiques que doivent vérifier les termes issus d'une analyse syntaxique afin d'être des candidats plausibles pour ce cas (cf table 2) :

AGT (Agent) : en général, l'entité qui contrôle un prédicat,
restriction sémantique : +Animate;

PAT (Patient) : l'entité affectée par un prédicat, restriction sémantique : +Animate;
OBJ (Object) : l'entité affectée par un prédicat, restriction sémantique : -Animate;
BEN (Beneficiary) : entité vers laquelle est effectuée un transfert ou à qui est appliqué l'action, restriction sémantique : +Animate;
SRC (Source) : entité à partie de laquelle s'effectue le mouvement d'une autre entité, restriction sémantique : +Location.

Verb Primitive	AGT	PAT	OBJ	BEN	SRC	DES
MVT1	+	-	-	-	+	+
MVT2	+	-	+	-	+	+
ACTANIME	+	-	-	+	-	-
ACTOBJET	+	-	+	o	-	-
ECHPROD	+	-	+	+	-	-
ECHOBT	o	-	+	+	-	-
ATRANS	+	-	-	o	-	-
LOCATION1	-	+/-	-/+	-	-	-
LOCATION2	-	+	+	o	-	-
EXTENSION	-	+	+	-	-	-
STATUT1	-	+	-	-	-	-
STATUT2	-	-	+	-	-	-
MESURE1	-	+	-	-	-	-
MESURE2	-	-	+	o	-	-
PROCESS1	-	+	o	-	-	-
PROCESS2	-	-	+	-	-	-

Table 2.

On notera l'ambiguïté apparaissant dans cette description entre ce qui concerne le lexique et ce qui concerne les référents. Ainsi, lorsqu'un cas donné doit être "animé", il semblerait naturel que, d'une part, l'élément lexical considéré porte ce trait, mais également que le référent possède la caractéristique correspondante. De façon plus générale, si on considère l'analyse casuelle comme centrée sur un prédicat au sens logique du terme, les arguments de ce prédicat doivent parcourir le modèle (au sens logique du terme) sous-jacent. Autrement dit, par souci de cohérence, la recherche des référents devra utiliser les propriétés sémantiques imposées par le prédicat[6].

[6] De manière plus générale, toute description componentielle du lexique pose le difficile problème du lien théorique (et pratique : Cavazza 1992) entre les traits ainsi définis et les

On imposera alors qu'un terme soit référentiel pour qu'il puisse correspondre à un cas. Cela semble confirmé par des exemples tels que :

Jean vole un cadre doré à l'ancienne

où "l'ancienne" ne peut être le bénéficiaire du verbe voler[7] que si dans le contexte linguistique ou extra-linguistique on est capable d'identifier un référent qui puisse être appelé "l'ancienne".

La question qui se pose alors immédiatement est de savoir si tous les termes référentiels de l'énoncé doivent correspondre à des cas de prédicats. Imaginons donc un contexte simple de manipulation d'objets sur un écran (une interface de type Macintosh devant laquelle on pourrait parler). Dans un tel contexte, un énoncé tel que :

"crée un texte en deux colonnes"

semblerait parfaitement admissible.

Si on souhaite employer une grammaire casuelle dans un tel système, on peut envisager pour cet énoncé deux types de résultats d'analyse :

créer (texte, 2 colonnes)

dans lequel "(en) 2 colonnes" correspondrait à un cas (probablement optionnel) de créer ou :

créer (texte)
format (texte, 2 colonnes).

La seconde représentation rend compte du fait que "en 2 colonnes" n'a de sens que relativement au fait qu'il s'agisse d'un texte. En revanche, en terme de fonction de la tâche, c'est bien une fonction du type : CREER-TEXTE-2-COLONNES sans argument qu'il faudra appeler, et cette fonction renverra un objet de type "texte". Pour se donner les moyens d'identifier cette fonction, il faudra bel et bien prévoir sur l'entrée lexicale "créer" la liaison possible avec "texte" et avec "2 colonnes", c'est-à-dire, indirectement, envisager l'énoncé complet ou presque au niveau du prédicat[8].

---

contraintes sur les référents. On ne peut se permettre de songer à une implémentation sans avoir effectué une réflexion profonde sur ce point.

[7] Et par conséquent l'énoncé devient ambigü.

[8] Du point de vue de la programmation, il est probable qu'on écrirait des règles séparées du lexique, mais dans la mesure où ces règles utiliseront directement des informations lexicales,

Ce problème provient, selon nous, de ce qu'on a considéré le prédicat comme étant l'élément central de la détermination de l'action à effectuer, et ce d'une façon trop indépendante de ses arguments. Dans la section suivante, nous envisagerons donc la démarche inverse qui consiste à s'appuyer sur les objets (les référents en cause dans l'action) et à considérer les prédicats comme des propriétés de ces objets.

3. La référence

Si nous revenons à notre problème de départ, à savoir la gestion de dialogues de commande, nous avons préjugé du fait que l'utilisateur souhaitait déclencher une action donnée, et toute notre analyse du langage (ou au moins la partie concernant les arguments de la commande) s'est appuyée sur les caractéristiques de cette action.

En dehors même des problèmes mentionnés dans la première section relativement à cette approche, on peut noter que cela suppose un utilisateur non seulement expert vis-à-vis de la tâche, mais connaissant également les fonctions élémentaires de cette tâche. Ainsi, si dans la tâche, on peut créer une fenêtre (dont la couleur par défaut sera bleue) et changer la couleur d'une fenêtre mais qu'on n'a aucune commande permettant en une seule étape de créer une fenêtre d'une couleur donnée, on a présupposé que notre utilisateur expert ne dira jamais quelque chose comme :

"crée une fenêtre rouge"

parce que atteindre son but dans un tel cas imposerait d'effectuer deux actions successivement.

Inversement, dans l'approche fondée sur les grammaires de cas, on acceptera un tel énoncé, mais on aura bien du mal à trouver l'action à effectuer, si de la même façon on préjuge d'un lien direct entre un verbe et une action unique.

Trouver la suite d'actions à effectuer, cette suite pouvant se réduire à une seule action, suppose donc de modéliser l'énoncé comme un but à atteindre. Autrement dit, avant de se poser la question des actions de la tâche à effectuer,

---

la séparation entre le lexique et ces règles n'est qu'une question technique de l'ordre du génie logiciel.

il faut avoir traduit l'énoncé sous forme de l'état attendu des référents auxquels il fait mention.

Ce faisant, on arrive à expliquer qu'un même énoncé puisse aboutir, selon les référents mentionnés, à des interprétations différentes en terme d'action de la tâche. Ainsi, si on suppose un système de type Macintosh, l'énoncé :

"Ouvre le document de gauche"

va pouvoir donner lieu au lancement de l'application MacDraw ou Word en fonction de caractéristiques du référent, non mentionnées dans l'énoncé.

On aura donc un typage des objets de la tâche reposant sur les actions qui peuvent leur être appliquées, ce typage étant indépendant du lexique au sens où les types correspondant ne recouvrent pas nécessairement des catégories lexicales.

Cette partie du problème étant en partie évacuée, il nous reste à expliquer comment on peut faire référence à des objets. En effet, si on admettait une sémantique pour laquelle "document" renverrait à l'ensemble des documents dans la tâche, on reviendrait directement au problème précédent. Il nous faut donc une représentation permettant, en contexte, de référer aux objets mentionnés. Autrement dit, l'ensemble des informations liées au niveau lexical à un mot donné ne sera pas nécessairement entièrement utilisé pour identifier un référent donné. Bien plus, on peut envisager que l'ensemble des informations lexicales utilisées pour identifier le référent dans un énoncé donné soit disjoint de celui utilisé pour identifier un référent de même nom dans un autre énoncé.

Afin d'illustrer notre propos, nous considérerons le cas de la référence définie, toujours sur l'exemple :

"affiche le document"

Nous restreindrons notre analyse aux références non-génériques. Dans ce cadre, la résolution du groupe nominal défini nous impose d'identifier un ensemble d'objets à l'intérieur duquel devra se trouver le référent "document". Si l'ensemble contient plus d'un élément (et c'est forcément le cas dans une situation donnée), le référent du groupe nominal défini étant vu comme unique en tant que document, il faut nécessairement que les autres éléments de l'ensemble ne soient pas des documents. C'est ce qu'illustre le schéma suivant, ou N représente les propriétés associées à la tête nominale du GN :

Dans ce schéma, N représente un ensemble d'informations permettant de donner le nom de document au référent relativement à l'ensemble considéré, c'est à dire vis-à-vis des autres éléments de cet ensemble qui sont vus comme ne méritant pas ce nom.

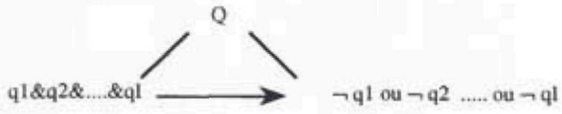
Supposons alors qu'on dispose d'un lexique dans lequel, à l'entrée "document", on dispose d'un ensemble de caractéristiques {p1, ..., pk} typiques des documents et indépendantes de tout contexte. Nous supposerons, pour simplifier l'exposé que ces caractéristiques sont directement des propriétés des objets qui peuvent être appelés "document"[9]. On a alors, dans le lexique, une description du document prototypique contenant éventuellement des alternatives (certaines propriétés peuvent se contredire entre elles, un choix étant alors nécessaire). A ces éventuelles propriétés incompatibles entre elles près, si on devait créer un nouveau référent qui soit un document, on pourrait créer un nouvel objet ayant toutes les propriétés {p1, ... pk} non niées par le reste du discours. Cela correspondrait à un Gn indéfini introduisant un nouveau référent.

Cette description lexicale étant posée, identifier le "document" dans un ensemble donné suppose de comprendre pour quelles raisons, ou pour rester dans le formalisme proposé, grâce à quelles propriétés, l'un des objets de l'ensemble est "le document" et les autres pas. Il nous faut donc un objet qui ait des propriétés en commun avec le prototype telles que l'une au moins de ces propriétés lui appartienne en propre vis-à-vis de chacun des autres éléments de l'ensemble.

Si donc on note {q1, ..., ql} les propriétés communes au référent et au prototype, chacun des autres éléments de l'ensemble devra ne pas posséder q1 ou q2 ou .... ql.

On arrive donc au schéma :

[9] Et ce par opposition à des propriétés plus "sémantiques" sur lequelles il faudrait encore effectuer des calculs pour arriver à des propriétés des référents.



Si on pose de plus que, à égalité de différenciation, le meilleur candidat à être le référent est celui qui possède le plus de propriétés communes avec le prototype, il est facile de se convaincre que la proposition se retourne et que le meilleur candidat est celui qui possède le plus de propriétés communes avec le prototype et à égalité, celui qui se différencie des autres éléments de l'ensemble (à condition que les éléments ayant le plus de propriétés n'aient pas exactement les mêmes).

Nous avons conscience d'avoir beaucoup simplifié les données du problème. En particulier, une telle approche suppose de pouvoir lister exhaustivement les propriétés des objets. Cette démarche ne serait par conséquent probablement pas valide hors de tout cadre. Dans le cas du dialogue homme-machine finalisé, il ne s'agit pourtant pas d'une hypothèse supplémentaire à celles que nous faisons déjà : de fait, la représentation informatique nous impose de toute façon de lister ces propriétés.

Par ailleurs, cette explication, pour simplifiée qu'elle soit, illustre selon nous la possibilité d'utiliser des prototypes sans pour autant imposer qu'un objet ne puisse être appelé N (N étant le nom associé au prototype) que s'il a toutes les propriétés imposées par ce prototype. Autrement dit, on sort d'une modélisation logique dans laquelle à un mot donné serait associé une dénotation indépendante du contexte.

Enfin, nous avons laissé deux points en suspens : le choix du bon ensemble dans lequel résoudre la référence définie et le fait que le nom employé par le locuteur soit approprié pour le référent désigné (dans les systèmes homme-machine, il est hors de question que la machine juge de la qualité du langage employé par le locuteur. Le système se borne à essayer d'interpréter. Les seuls cas d'échecs sont donc des échecs dans l'interprétation).

4. Perspectives

Dans le cadre particulier du dialogue finalisé homme-machine, sur des applications simples de type commande de processus, l'approche courante consiste à utiliser, au niveau même de la définition du langage, les informations

de typage des objets provenant de la description de la tâche. Cette approche pose deux types de problèmes : d'une part elle présuppose un utilisateur connaissant parfaitement l'application en terme de ses fonctions élémentaires, et d'autre part, elle aboutit à distribuer sur le lexique le typage de la tâche rendant l'ensemble ingérable dès que l'on dépasse quelques fonctions.

Il nous semble donc nécessaire de séparer la description du langage et la description de l'application. Dans le cas particulier des dialogues de commande, cette séparation nous conduit à considérer les énoncés de l'utilisateur comme spécifiant des buts à atteindre par le système. La référence aux actions devient alors un problème de planification à résoudre dans le modèle de la tâche, indépendamment du contenu lexical de l'énoncé puisque portant directement sur des référents.

Résoudre les références aux objets mentionnés dans l'énoncé devient alors crucial et ne peut se faire en spécifiant pour chaque nom N un ensemble de conditions stables et valables en tout contexte pour être un grand N. Une telle approche ramènerait en effet inévitablement au problème de départ (on pourrait spécifier les actions possibles sur les "N" et les prévoir dans la grammaire). Le contenu lexical des noms doit donc être plus vaste que ce qui sera employé pour une référence donnée, à charge pour le mécanisme de calcul de la référence de retenir les propriétés discriminantes du référent dans le contexte.

Références bibliographiques

CALLIOPE (collectif) (1989), *La parole et son traitement automatique*, Masson.

CAVAZZA, M. (1992), *Analyse Sémantique du Langage Naturel par Construction de Modèles*, Thèse de Doctorat de l'Université de Paris VII.

DEVILLE, G., PAULUSSEN H. & PIERREL J.-M. (1987), "Une grammaire de cas comme modèle de représentation sémantique d'énoncés de dialogues oraux homme-machine finalisés", *Actes du 6ème Congrès AFCET-RFIA*, Antibes, 157-174.

DEVILLE, G. (1989), *Modelization of task-oriented utterances in a man-machine dialogue system*, Thesis of Doctor of Linguitics, University Instelligen, Antwerpen.

HATON, J-P. (1980), "The representation and the use of a Lexicon in Automatic Speech Recognition and Understanding", in SIMON, J. C. (ed.), *Spoken Language Generation and Understanding*, Dordrecht, Reidel, 331-335.

HATON, J-P. (1985), "Intelligence artificielle en compréhension automatique de la parole : état des recherches et comparaison avec la vision par ordinateurs", *TSI* 4-3, 265-287.

HATON, J-P., PIERREL J.-M., PÉRENNOU G., CAELEN J. & GAUVAIN J.-L. (1991), *Reconnaissance automatique de la parole*, Paris, Dunod.

MORIN, P. & PIERREL J.-M. (1987), "PARTNER : un système de dialogue oral homme-machine", in *Actes Cognitiva-87*.

PÉRENNOU, G.(1987), "On the role and the structure of the lexicon in Language Understanding" in HATON J.-P. (ed.), *Fundamentals in Computer Understanding*, Cambridge, Cambridge University Press, 217-248.

PIERREL, J-M. (1987), *Dialogue oral homme-machine*, Paris, Hermès.

PIERREL, J-M. (1989), "Lexique et compréhension automatique de la parole", *Lexique* 8, 137-166.

ROMARY, L. (1989), *Vers la définition d'un modèle cognitif pour la représentation du temps dans un dialogue oral homme-machine*, Thèse de l'Université de Nancy I.

ROMARY, L. & PIERREL J.-M. (1989), "The use of Dempster-Shafer rule in the lexical component of a man-machine oral dialogue system", *Speech Communication*.

ROUSSANALY, A. & PIERREL J.-M. (1992), "Dialogue oral homme-machine en langage naturel : le projet DIAL", *Techniques et Sciences Informatiques* 11 (2), 45-91.

SEGUI J. (1989), "L'accès au lexique, données expérimentales et modèles", in Calliope (1989).