Caractérisation de Requêtes d'Assistance à partir de corpus

François Bouchet* Jean-Paul Sansonnet*
bouchet@limsi.fr jps@limsi.fr

*LIMSI-CNRS
Université Paris-Sud XI
BP 133, F-91403 Orsay Cedex

Résumé :
La modélisation formelle de l'interaction entre les usagers grand public et les systèmes informatiques a un rôle crucial à jouer au niveau de la Fonction d'Assistance. Il s'agit préalablement de caractériser sémantiquement, en termes de couverture et de précision, des phénomènes liés à l'assistance pour proposer à terme un agent rationnel assistant générique capable de réactions pertinentes aux requêtes des usagers. Dans cet article, nous présentons notre approche de construction d'un langage pour des agents conversationnels assistants, basée sur l'étude préalable d'un corpus de requêtes recueillies dans des situations effectives d'assistance.

Mots-clés : Agents conversationnels assistants, corpus de requêtes d'assistance, langage de requêtes

Abstract:
Formal modeling of the interaction between ordinary users and computer-based systems has a major part to play in the Assistance Function. A first objective is to characterize semantically, both in coverage and precision, the phenomena associated with the Assistance Function to provide a generic assisting rational agent capable of pertinent reactions to users' requests. In this paper, we present our approach to the construction of a language for a class of assisting conversational agents, based on the study of a corpus of users' requests registered in actual assisting experimentations.

Keywords: Assisting conversational agents, assistance requests corpus, requests language

1 Cadre de l'étude

1.1 La Fonction d'Assistance

Le développement de l'informatique grand public a entraîné une forte augmentation du nombre d'usagers novices en informatique n'ayant ni le temps ni l'envie d'utiliser les manuels papiers ou les FAQ de logiciels de plus en plus complexes (en dépit des progrès ergonomiques), dont par ailleurs ils ne maîtrisent pas le vocabulaire spécifique. Des systèmes d'aide contextuelle (CHS – Contextual Help Systems [5]) ont été développés pour mieux s'adapter à leurs besoins, un ami expert est toujours préféré pour réaliser une tâche donnée dans une application [3]. Parallèlement, les agents conversationnels animés dotés de capacités de dialogue et de raisonnement [11] ont mis en évidence les avantages d'une présence virtuelle pour faciliter l'interaction homme-machine [8].
Pour répondre à ce besoin d'assistance, le projet DAFT mené au LIMSI-CNRS [13] se propose de développer des Agents Conversationnels Assistants (ACA), capables d'analyser des requêtes en langue naturelle écrite non contrainte d'*usagers novices* en *situation réelle* d'utilisation d'applications de complexité croissante (applets, pages web, sites actifs, traitements de texte…), pour fournir une assistance pertinente en contexte à la manière des CHS avec les bénéfices liés à la présence d'agents conversationnels animés.

1.2 Traitement des requêtes

Le système d'assistance DAFT vise à fournir un noyau d'agent rationnel. L'architecture du système d'assistance (cf. fig. 1) contient classiquement : le module d'analyse sémantique des requêtes usager (GRASP) construisant les requêtes formelles[1], le module de raisonnement sur le

[1] par opposition à la forme en langue naturelle. La question de la sémantique opérationnelle associée aux requêtes, définie par les réactions de l'agent rationnel, n'est pas traitée ici.

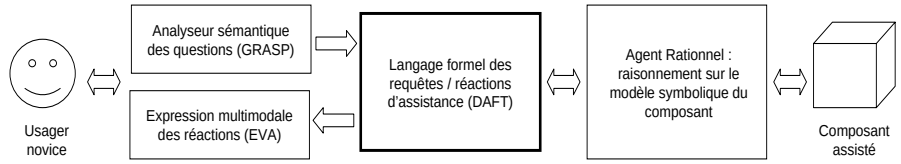


FIG. 1 – Architecture générale de la chaîne de traitement des requêtes d'assistance

modèle de l'application qui retourne une réponse formelle, et le module d'expression de cette réponse (EVA) selon trois modes, éventuellement conjoints :
1. Réplique de l'agent en langue naturelle : selon plusieurs modalités : bulle de texte, synthèse de parole, popup d'aide ;
2. Interaction avec l'application assistée : consultation et/ou modification des composants via un modèle de l'application ;
3. Animation de l'agent : le personnage animé peut exprimer certains états de l'agent (désaccord, incompréhension…) ou désigner les composants mentionnés.

1.3 Caractérisation du langage formel

Pour caractériser le langage de requêtes, nous étions précédemment [10] partis de la modélisation interne des composants de l'application, mais des expérimentations avec des usagers effectifs ont révélé des phénomènes de dérive cognitive significatifs [7] montrant qu'il faut prendre en compte en priorité le point de vue de l'usager : sa perception cognitive de l'application mais aussi ses attitudes mentales, exprimées dans les groupes verbaux sous-tendant les actes de langage (cf. §3.1).
C'est pourquoi il nous faut un *corpus de requêtes d'assistance*, permettant de cerner le domaine de langue concerné par la Fonction d'Assistance et d'étudier la distribution des phénomènes associés, qui sera présenté dans la section 2. Nous présenterons ensuite les spécifications du langage de requêtes, puis une évaluation de son adaptation à notre domaine d'étude.

2 Présentation du corpus

2.1 Recueil du corpus

Peu de données publiques sont effectivement disponibles en dialogue *écrit* Homme/Machine, et nous nous situons dans le cadre assez différent des NLI (Natural Languages Interfaces [1]), traitant de *requêtes isolées* et non de *sessions dialogiques*. Enfin, il était nécessaire de contrôler précisément les conditions expérimentales de recueil des requêtes centrées sur la Fonction d'Assistance, aussi avons-nous été amenés à recueillir notre propre corpus.

Le corpus DAFT comprend 11 000 requêtes issues également de trois sources :
1. Sur une période de deux ans (juin 2004 - juin 2006), des sujets ont été placés devant divers types d'applications assistées par le système DAFT (1.0) : trois applets Java (modales et amodales i.e. contenant des threads), deux sites web (dont un éditable par les usagers) ;
2. Des requêtes construites manuellement à partir de deux thésaurus afin d'assurer une couverture linguistique élargie ;
3. Des FAQ de deux logiciels de traitement de texte très utilisés (Latex et Word).

Une étude comparative du corpus DAFT avec quatre autres corpus de dialogue existants (DAMSL/Switchboard [6], Map-Task [4] et Bugzilla [9]) valide la pertinence de notre démarche de recueil d'un corpus spécifique, et la section suivante en résumera certaines conclusions [2].

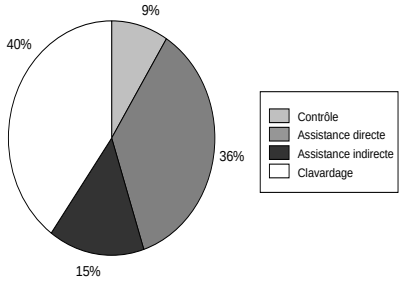


FIG. 2 – Répartition des requêtes du corpus DAFT par classes d'activités.

2.2 Catégorisation des activités conversationnelles du corpus

Lors de la phase de recueil du corpus, les sujets humains devaient réaliser certaines tâches, en faisant appel si nécessaire à un agent présent dans l'application pour les assister, mais pouvaient agir et s'exprimer de manière non contrainte. Quatre classes de comportements conversationnels ont pu être observées :

– **activité de contrôle** : corpus constitué de commandes, afin que l'agent agisse lui-même sur l'application ;
– **activité d'assistance directe** : regroupant des demandes d'aide explicitement formulées comme telles par l'utilisateur ;
– **activité d'assistance indirecte** : corpus de jugements de l'usager sur l'application constituant des demandes d'aide sous-entendues, perceptibles uniquement au niveau pragmatique ;
– **activité de clavardage** : réunissant le reste des interactions essentiellement centrées sur l'agent, ainsi que des expressions métalinguistiques, phatiques et de back-channeling.

La figure 2 donne la distribution de ces classes et la table 1 présente des exemples de phrases issues de chacune de ces classes. L'existence des sous-corpus de contrôle et de clavardage montre que l'utilisateur attend non seulement d'un ACA qu'il l'aide à utiliser l'application, mais aussi qu'il soit capable d'agir lui-même sur celle-ci, ainsi que de répondre à des commentaires annexes indépendants de la tâche à accomplir où il devient lui-même le centre d'attention (phénomène lié à la présence visuelle de l'agent).

3 Le langage de requêtes DAFT

3.1 Structure générale

L'étude menée à partir du corpus DAFT nous a permis de dégager une structure générale des requêtes (dans la suite nous parlerons de schémas de requêtes – en abrégé, de schémas), décomposables en trois niveaux imbriquables, donnés ci-dessous en partant de la couche externe :
– Les **Modalités** (notées M_i) : elles correspondent partiellement aux actes de langage au sens de Searle [14]. Elles donnent une valeur particulière, éventuellement liée à l'usager l'exprimant, à un contenu prédicatif (interrogation, doute, volonté…).
– Les **Prédicats** (notés P_i) : il s'agit essentiellement de verbes d'action (modifier, déplacer…) ou de descriptions d'états (faire partie de…).
– Les **Références** (notées R_i) : au sens de Références Extensionnelles Associatives [12] (toute mention ultérieure sera à prendre dans ce sens) ; elles correspondent généralement aux groupes nominaux.

3.2 Syntaxe et notations

Les modalités et prédicats sont tous dotés d'un schéma qui les lie à un ensemble d'unités sémantiques spécifiques appelées champs. On notera une *modalité* :

`SYMBOLE-MODALITE(`ch_1`, ..., `ch_n`)`

et un *prédicat* s'écrira :

`SymbolePrédicat(`ch_1`, ..., `ch_n`)`

où $ch_1, \ldots, ch_n$ représentent des champs. Un *champ* est lui-même constitué d'un couple *attribut-valeur* et s'écrit :

`attribut=valeur`

TAB. 1 – Quelques exemples de requêtes du corpus DAFT

N°	Corpus	Phrase → Transcription
1	Contrôle	*glisser le disque de droite à gauche* ⟶ `Move(obj="le disque", from="droite", to="gauche")`
2	Contrôle	*Donne moi la liste des projets* ⟶ `Show(obj="la liste des projets")`
3	Ass. dir.	*c quoi le GT ACA* ⟶ `ASK(INFOS(about="le GT ACA"))`
4	Ass. dir.	*comment faire pour gagner ?* ⟶ `ASK(WAY(goal=Win()))`
5	Ass. dir.	*d'après toi, y a t-il des fonctions d'annulation dans cette application ?*
⟶ `ASK(KNOWLEDGE(of="system", about=EXISTENCE(of=FUNCTION(doing=Cancel(), in="cette appli")))`		
6	Ass. dir.	*j'ai bien peur qu'il n'y ait pas moyen de changer la taille de la police qui est trop petite*
⟶ `FEAR(agent="user", fear=NEG(POSSIBILITY(todo=Modify(obj="la police...", property="taille"))))`		
7	Ass. dir.	*quelles sont les couleurs disponibles ?* ⟶ `ASK(OBJECT(`**`isa="couleur"`**`, ppt="disponible"))`
8	Ass. dir.	*ya un truc de couleur rouge ?* ⟶ `ASK(EXISTENCE(of=OBJECT(isa="truc",`**`ppt="couleur"`**`,val="red")))`
9	Ass. ind.	*le bouton "fermer" est semblable au bouton "quitter"*
⟶ `NEG(DIFFERENCE(between="le bouton 'fermer'", and="le bouton 'quitter'"))`		
10	Ass. ind.	*probablement que le bouton NEXT ne marche pas correctement*
⟶ `PROBABILITY(degree="2", of=PROBLEM(with="le bouton NEXT"))`		
11	Clav.	*Absolument* ⟶ `ACK()`
12	Clav.	*bonjour je suis Arthur* ⟶ `HELLO(),VALUE(object="user", property="nom", value="Arthur")`
13	Clav.	*sais-tu chanter ?* ⟶ `- non traitée -`

Les *valeurs* possibles des champs sont fortement liées au type de ces champs (cf. §3.3) et ont trois natures possibles :
– une référence (entre guillemets) issue de la requête en langue naturelle (ex : "le bouton").
– un marqueur symbolique (entre guillemets et en majuscule) dépendant du type du champ et indirectement issu de la requête de l'usager (ex : "TRUE" pour un booléen, "PAST" pour un temporel).
– une modalité ou un prédicat (uniquement lorsqu'il s'agit d'un champ de modalité).
La syntaxe générale d'une requête formelle est alors de la forme :

$M_1(...M_n(a_1 = P_1(a'_1 = R_1, ..., a'_l = R_l), ..., a_m = P_m(...)) ...)$

Expression dans laquelle :
M_1 – M_n sont des *modalités*,
P_1 – P_m sont des *prédicats*,
a_1 – a_m, a'_1 – a'_l sont des *attributs*,
R_1 – R_l sont des *références*.
Plusieurs modalités peuvent s'imbriquer entre elles et contenir plusieurs prédicats coordonnés (i.e. au même niveau) contenant chacun une ou plusieurs références.

3.3 Les types de champs

Tous les champs de modalités et de prédicats sont dotés d'un type, et les attributs de ces champs ne peuvent être remplis que par une référence de type compatible[2]. Les 11 types définis au fur et à mesure de la transcription des phrases selon la nature des références sont définis dans la table 2[3]. Les schémas typés aident aussi à choisir le bon schéma lorsque plusieurs sont possibles, si les types des champs de l'un ne correspondent pas aux types des références de la requête langagière.
Pour gérer l'imbrication de schémas, on définit un type global pour les modalités pouvant être imbriquées (le type global des prédicats est toujours `[act]`). Ainsi la phrase *"y a-t-il un minimum pour la vitesse ?"* se transcrit :

[2]Cela suppose que l'analyse sémantique détermine le(s) type(s) des références ; actuellement, ceci se fait par l'attribution des types au niveau d'environ 1500 clés sémantiques (regroupant lemmes et locutions, équivalent de synsets Wordnet).

[3]En dépit des exemples, il n'y a pas d'équivalence lemme-type (voir "couleur" dans les phrases 7 et 8) ; la désambiguisation peut s'effectuer à l'aide d'informations grammaticales fournies par le module GRASP.

TAB. 2 – Liste des 11 types de champs du langage DAFT

Type	Signification	Exemples
[act]	Action	Prédicats d'action (cf. table 4)
[con]	Concept abstrait	*"les actions sur les tables"*, *"une nouvelle partie"*…
[obj]	Objet de l'application	*"une page"*, *"un disque"*, *"ça"*…
[ppt]	Propriété	*"couleur"*, *"le plus récent"*…
[val]	Valeur	*"bloqué"*, *"vrai"*, *"bleu"*…
[typ]	Type	*"page"*, *"lien"*, *"projet"*, *"membre"*…
[per]	Personne	utilisateur ou système uniquement
[man]	Manière	*"à l'envers"*, *"un tout petit peu"*, *"bêtement"*…
[pla]	Lieu ou zone de l'application	*"le site"*, *"au début"*, *"l'application"*…
[tim]	Instant ou durée	*"PAST"*, *"NOW"*, *"ALWAYS"*…
[boo]	Booléen	*"TRUE"* et *"FALSE"*

```
ASK(EXISTENCE(time="NOW", of=VALUE(
object="vitesse", asked="min")))
```

où `EXISTENCE` a un champ `of` attendant un élément de type `[obj]` : le type global de `VALUE` est donc `[obj]`.

3.4 Modalités, prédicats, références

L'annotation du corpus a permis de déterminer 39 **modalités**, regroupées selon leur *structure* ou leur *signification* en 8 catégories (cf. table 3), distinguables selon :
– Leur profondeur dans la requête DAFT : sans détailler les règles d'imbrication, les modalités de jugement ou d'expression de sentiments se situent plutôt dans les couches externes des requêtes (ex : phrase 6 de la table 1), tandis que les modalités d'informations sont plutôt sur les couches internes près des prédicats (ex : phrase 5).
– La complexité des schémas de requêtes : les marqueurs (cf. table 3) ne prennent qu'un argument unique, tandis que des schémas comme `OBLIGATION` en prennent 4 ou 5, même si tous les champs sont rarement remplis simultanément.
– La complexité d'identification du phénomène sémantique représenté : des modalités comme `PLACE` ou `MOMENT` sont parfois implicites et difficiles à éliciter, ainsi : *"je ne trouve pas X"*, sous-entend "l'endroit **où** est X", alors que dans *"stoppe le compteur à 1000"*, on devra éliciter : "**quand** il vaut 1000".
– Leur fréquence : comme souvent avec des phénomènes linguistiques, la distribution des modalités suit une loi de Zipf.
– Leur généricité : les modalités circonstancielles et de jugement sont plutôt classiques dans ce type d'approche [6]. Par contre, celles de la catégorie "assistance" sont très liées au domaine de langue étudié, ce qui démontre l'intérêt du corpus.

Nous avons aussi défini les schémas de 46 **prédicats** (cf. table 4[4]), dont la fréquence suit également une loi de Zipf, constituant une base assez complète d'actions pour les sous-corpus de contrôle et d'assistance dans le cadre des applications testées.

Les **références** ont été généralement gardées sous forme brute, mais la modalité `OBJET` permet de les encapsuler pour aider l'identification de l'entité référencée dans l'application (pour la phrase 2, on pourrait écrire `objet = OBJET(`**`isa`**`="liste",` **`property`**`="projets")`).
La principale difficulté (outre la résolution de la référence) est d'automatiser cette décomposition, et elle est liée au problème d'identification des types (cf. § 3.3).

[4] Les types suffisent bien souvent à la compréhension puisqu'ils sont liés à la structure actancielle classique du verbe – en cas d'ambiguïté, nous précisons un nom ou un exemple de référence possible (entre guillemets).

TAB. 3 – Liste des 39 modalités M_i de DAFT

Informations structurelles et fonctionnelles d'une entité	
INFOS	(Avoir) des informations/plus d'informations sur un sujet.
MEANING	Le sens de quelque chose, d'après quelqu'un.
TYPE	Le type d'un objet est tel type.
VALUE	La valeur (de la propriété d'un objet / d'un objet) est valeur.
PROPERTY	La propriété 'propriété' d'un objet vaut une valeur.
FUNCTION	La fonction faisant une action déterminée pour des objets donnés dans un cadre particulier, et ayant certaines propriétés.
ROLE	Le rôle d'un objet / concept / personne par rapport à un référentiel.
LIMIT	Limites dans un objet / des propriétés d'un objet.
FUNCTIONING	Le fonctionnement d'un objet.
Identification d'entités	
EXISTENCE	Existence de quelque chose dans un endroit.
OTHER	Un objet / Une action autre que celui / celle considéré(e).
OBJECT	Un objet ayant un type, des propriétés et situé en un lieu donné. Peut servir à effectuer une action ou être objet de celle-ci.
Relations inter-entités	
DIFFERENCE	Différences entre une entité et une autre selon un critère.
ORDER	Réalisation d'un ensemble d'actions étape par étape, la dernière étant considérée comme le but de la procédure.
Capacités, droits et devoirs d'une entité	
POSSIBILITY	Possibilités offertes à quelqu'un par une autorité (de réaliser quelque chose / au sujet d'un objet) pour quelqu'un.
KNOWLEDGE	(Connaissance/Savoir/Compréhension/Capacité) de quelqu'un/quelque chose (pour faire une action / au sujet d'un objet).
OBLIGATION	Obligation d'une personne par une autre à faire quelque chose / d'un objet.
WILL	Volonté de quelqu'un qu'une action soit réalisée par une personne / d'un objet.
PROBABILITY	Probabilité d'une action / d'un objet à un certain degré.
Circonstancielles	
WAY	Moyen d'atteindre un but.
EFFECT	L'effet d'une action / d'un objet est une autre action.
REASON	La raison/cause de quelque chose.
PLACE	Emplacement de quelque chose / où faire quelque chose.
INSTANT	Instant où une action se fait / où un événement a lieu.
NUMBER	Nombre d'éléments dans un espace donné.
Sentiments	
LIKE	Un agent aime un objet aimé (quelqu'un ou quelque chose).
FEAR	Un agent a peur d'une crainte.
BOTHER	Le gêné est incommodé par une gêne.
DOUBT	Un doute d'une personne au sujet de quelque chose (absence de certitude).
SURPRISE	Un agent est surpris par quelque chose.
REGRET	Un agent a des regrets au sujet de quelque chose.
HAPPY	Un agent est heureux au sujet de quelque chose.
Assistance	
PROBLEM	Situation insatisfaisante, résultat inattendu. Problème d'une personne avec un objet ou pour faire une action.
MISTAKE	Acte involontaire/inadapté. Un responsable commet une erreur sur une action ou un objet.
HELP	Un demandeur réclame l'aide d'un assistant, (pour réaliser une action / au sujet d'une entité), en faisant quelque chose.
TELL	Un locuteur s'adresse à un interlocuteur pour lui parler d'un sujet ou lui exposer une action à réaliser.
Marqueurs	
CHECK	Marqueur de demande de vérification d'une information.
ASK	Marqueur d'interrogation, pour toute interrogation qui n'est pas un CHECK.
NEG	Marqueur de négation précédant une action ou modalité (sa position est alors à prendre en compte pour retrouver le sens).

TAB. 4 – Liste des 46 actions du langage DAFT (ordonnée par le nombre de champs)

Bip()	**Start**([obj])	**MakeAMove**([pla]*from*,[pla]*to*)
End() – sens contextuel	**Swap**([obj])	**Sort**([obj],[man]*"en sens inverse"*)
Scroll()	**Use**([obj])	**Update**([obj],[per]*by*)
Activate([obj])	**Win**([man]*"facilement"*)	**Add**([obj],[pla]*in*,[man]*"sans risque"*)
Cheat([man]*"un peu"*)	**Belong**([obj],[obj]*to*)	**Click**([per],[obj],[man])
Check([obj])	**Cancel**([act],[obj])	**Exceed**([ppt],[obj]*of*,[val])
Contact([per])	**Close**([obj],[tim]*when*)	**Give**([per]*from*,[obj],[per]*to*)
Count([man]*"rebours"*)	**Control**([obj],[ppt])	**Go**([pla]*to*,[tim]*at*,[man])
Create([obj])	**Delete**([obj],[man])	**Operate**([per]*operator*,[obj],[man])
Download([obj])	**Handle**([per],[act])	**Modify**([obj],[ppt],[val],[man])
Happen([tim]when)	**Join**([per],[obj])	**Stop**([act],[obj],[tim]*when*,[tim]*during*)
Hide([obj])	**Restart**([obj],[per])	**Play**([per],[per]for,[obj],[tim]*"next turn"*,[man])
Quit([obj])	**Restore**([obj],[val])	**Show**([per]*shower*,[per]*to*,[obj],[man],[pla]*in*)
Recommend([obj])	**Revert**([act],[con])	**Move**([obj],[per],[pla]*in*,[pla]*from*,[pla]*to*,[man])
Repeat([con]*"partie"*)	**Save**([obj],[pla]*in*)	
Replay([con]*"jeu"*)	**See**([obj],[per])	

TAB. 5 – Taux de couverture des 4 sous-corpus par le langage DAFT

Sous-corpus	Ctrl	Ass. dir.	Ass. ind.	Clav.*
Couverture	92,3%	96,7%	70,2%	82,4%

*(hors dialogue)

4 Discussion et évaluation

4.1 Adéquation et robustesse

Une fois une première version du corpus DAFT recueillie (environ 5000 requêtes), le traitement a été effectué manuellement à partir de deux sous-ensembles au 1/10^{e} (1075 phrases). Parmi elles, 698 ont été transcrites en requêtes DAFT respectant la syntaxe définie en 3.2, offrant une couverture (rapport nombre de phrases transcrites / nombre de phrases totales) globalement correcte mais inégale (cf. table 5)[5].

L'ajout de nouvelles requêtes d'assistance dans le corpus ne doit entraîner ni remise en question de la structure adoptée, ni ajout de nombreux prédicats. Nous avons donc étudié une nouvelle tranche de 1/10^{e} du corpus retranscrite en DAFT (uniquement pour les sous-corpus de contrôle et d'assistance) et avons ainsi pu estimer que seules 2 modalités de quantification (`TOO`, `ENOUGH`) et 4 prédicats d'action (`Bloquer`, `Communiquer`, `Choisir`, `Agir`) supplémentaires seraient nécessaires, soit un ajout de l'ordre de 5% pour une augmentation de 50% de la taille du corpus. De plus, la structure même de la syntaxe proposée n'a pas été mise en défaut, confirmant que la méthodologie employée (sous-ensembles au 1/10^{e}) n'a pas biaisé le langage de requêtes.

4.2 Perspectives

Il reste à savoir s'il est computationnellement possible de produire des réactions *correctes*, ce qui nécessite de relier les éléments des requêtes aux éléments de l'application assistée ; mais surtout *pertinentes*[6]. La conception d'un système intégrant le langage de requêtes proposé pour

[5] Une partie du sous-corpus de clavardage de dialogue avec l'agent (202 phrases) n'a pas été prise en compte : la structure de ces requêtes est proche des requêtes d'assistance, mais il faudrait étendre le vocabulaire des prédicats.

[6] En particulier, en cas d'échec du système de raisonnement sur la structure du composant, il est toujours possible de sim-

une large classe de composants assistés constitue la suite directe de cette étude, et permettra une évaluation plus approfondie des choix effectués.

Références

[1] Ion Androutsopoulos and Maria Aretoulaki. *The Oxford Handbook of Computational Linguistics*, chapter Natural Language Interfaces, pages 629–649. Oxford University Press, March 2003.

[2] François Bouchet and Jean-Paul Sansonnet. Étude d'un corpus de requêtes en langue naturelle pour des agents assistants. In *Proc. of WACA 2006*, October 2006.

[3] Antonio Capobianco and Noëlle Carbonell. Conception d'aides en ligne pour le grand public : défis et propositions. In J.-M. Robert A. Drouin, G. Eude, editor, *Proc. ERGO-IA'2002*, pages 309–335, October 2002.

[4] Jean Carletta, Amy Isard, Stephen Isard, Jacqueline Kowtko, Gwyneth Doherty-Sneddon, and Anne Anderson. Hcrc dialogue structure coding manual. Technical report, HCRC, University of Edinburgh, June 1996.

[5] Bernard J. Jansen. Seeking and implementing automated assistance during the search process. *Information Processing and Management*, 41(4) :909–928, July 2005.

[6] Daniel Jurafsky, Rebecca Bates, Noah Coccaro, Rachel Martin, Marie Meteer, Klaus Ries, Elizabeth Shriberg, Andreas Stolcke, Paul Taylor, and Carol Van Ess-Dykema. Switchboard discourse language modeling project final report. Technical report, Center for Speech and Language Processing, Johns Hopkins University, 1998.

[7] David Leray and Jean-Paul Sansonnet. Ordinary user oriented model construction for assisting conversational agents. In *CHAA'06 at IEEE-WIC-ACM Conference on Intelligent Agent Technology*, 2006.

[8] James C. Lester, Sharolyn A. Converse, Susan H. Kahler, Steven Todd Barlow, Brian A. Stone, and Ravinder S. Bhogal. The Persona Effect : Affective impact of animated pedagogical agents. In *CHI '97 : Proceedings of the SIGCHI conf. on Human factors in comp. syst.*, pages 359–366, New York, NY, USA, March 1997. ACM Press.

[9] Gabriel Ripoche. *Sur les traces de Bugzilla*. PhD thesis, Univ. Paris XI, June 2006.

[10] Nicolas Sabouret and Jean-Paul Sansonnet. Un modèle de requêtes sur le fonctionnement de composants actifs. In *Proc. MFI 01*, volume 3, pages 419–436, May 2001.

[11] David Sadek, Philippe Bretier, and E. Panaget. Artimis : Natural dialogue meets rational agency. In *IJCAI (2)*, pages 1030–1035, August 1997.

[12] Susanne Salmon-Alt. *Référence et dialogue finalisé : de la linguistique à un modèle opérationnel*. PhD thesis, Univ. H. Poincaré, Nancy 1, May 2001.

[13] Jean-Paul Sansonnet, Karl Le Guern, and Jean-Claude Martin. Une architecture médiateur pour des agents conversationnels animés. In *Proc. WACA'01*, pages 31–39, June 2005.

[14] John Rogers Searle. *Speech Acts : An essay in the Philosophy of language*. Cambridge University Press, new edition, January 1969.

plifier les requêtes afin de parvenir à une réponse vide comme "*Désolé, je ne peux donner d'explication au fait que le compteur s'est arrêté.*", correcte mais pas réellement pertinente.