Un système de dialogue fondé sur l'étude d'interactions humaines.

Alain Loisel, Guillaume Dubuisson Duplessis, Nathalie Chaignaud, Jean-Philippe Kotowicz

Un système de dialogue fondé sur l'étude d'interactions humaines

A. Loisel⋆ G. Dubuisson Duplessis⋆ N. Chaignaud⋆ J.-Ph. Kotowicz⋆

⋆INSA Rouen LITIS - EA 4108, BP08, 76801 Saint-Etienne du Rouvray
[name]@insa-rouen.fr

Résumé :
Cet article décrit un module de dialogue Homme-Machine pour la recherche d'information. Notre démarche interdisciplinaire vise une modélisation computationnelle en procédant à un recueil de corpus de dialogues entre un utilisateur jouant le rôle d'un patient désirant une information médicale et un expert CISMeF affinant cette demande pour construire la requête. La structure des dialogues ainsi obtenus a été analysée en relevant un certain nombre d'indices discursifs. À partir de cette analyse, nous avons construit et implémenté un modèle d'agent artificiel capable d'aider l'utilisateur dans sa tâche de recherche d'information.

Mots-clés : Interaction, dialogue Homme-Machine, modélisation, analyse de corpus

Abstract:
This study strives to improve medical information search by including a conversational agent to interact with the user in natural language. To study the processes involved during the information search, a bottom-up methodology was adopted. Experimentation has been set up to obtain human dialogues between a user (playing the role of patient) dealing with medical information search and a CISMeF expert refining the request. The analysis of these dialogues underlined the use of discursive evidence. A model of artificial agent has been implemented. It assists the user in its information search by proposing to him examples and choices.

Keywords: Interaction, human-computer dialogue, modeling, corpus analysis

1 Introduction

Pour accomplir une tache de recherche d'informations spécialisées en interagissant avec l'utilisateur en langue naturelle, la machine doit tenter d'analyser son objectif et lui proposer des solutions. Présenter des exemples, des aides, des corrections ou des clarifications sous forme de choix permet d'accompagner l'interlocuteur jusqu'à une solution, si nécessaire en élargissant son but initial. Un tel système examine avec l'utilisateur les mots-clés à employer. Il teste ainsi plusieurs requêtes en utilisant des critères pour orienter la recherche dans la direction souhaitée par l'utilisateur. Dès lors, le dialogue permet d'établir un *terrain partagé*.

Notre étude interdisciplinaire s'inscrit dans le cadre du projet Cogni-CISMeF (financé par le Programme Interdisciplinaire TCAN du CNRS) qui vise à intégrer un module de dialogue H-M dans le système d'indexation de connaissances médicales CISMeF (Catalogue et Index des Sites Médicaux Francophones, www.cismef.org) [8]. Le système CISMeF propose une interface graphique et un langage de requête permettant de décomposer une requête en mots-clés issus d'un lexique contrôlé (le MeSH, Medical Subject Headings). La terminologie CISMeF comporte des *mots-clés*, des *qualificatifs* (symptômes, traitements, etc.), des *méta-termes* (spécialités médicales) et des *types de ressources*. Le système s'est étoffé en offrant la possibilité d'effectuer des requêtes étendues. Or, l'utilisateur est souvent peu enclin à l'utilisation de requêtes complexes. D'où notre proposition de recourir au dialogue en langue naturelle pour faciliter la recherche de documents.

Notre approche a pour but de concevoir un système apte à s'adapter à un interlocuteur humain dans un domaine spécialisé. Cela nécessite non seulement d'extraire explicitement ses connaissances pour les traduire dans un langage informatique, mais également de comprendre les mécanismes sous-jacents. Pour que les échanges H-M puissent s'effectuer dans de bonnes conditions, l'existence de moyens de communication qui prennent en charge les points de vue du demandeur via des interfaces efficaces et conviviales paraît essentielle. L'homme est doté de processus robustes de communication et de raisonnement qui lui sont propres. Notre hypothèse est que c'est en prenant en compte les contextes, les stratégies discursives et les variations langagières des locuteurs que les machines seront mises au service de l'homme, et non l'inverse. De ce point de vue, les communications H-M ne sauraient être conçues que grâce à l'étude des interactions Homme-Homme (H-H). Nous sommes conscients que la nature même du dialogue H-M est différente de celle du dialogue H-H, mais l'analyse d'interactions H-H devrait toutefois permettre d'obtenir des indices indispensables à la conception d'un dialogue H-M.

Pour atteindre un tel objectif, nous avons procédé à un recueil de corpus de dialogues entre

un utilisateur jouant le rôle d'un patient désirant une information médicale et un expert CISMeF affinant cette demande pour construire la requête. L'analyse de la structure des interactions ainsi obtenues et l'étude d'un certain nombre d'indices discursifs nous ont permis de dégager les principales caractéristiques de ces dialogues. C'est à partir de l'étude de ce corpus que notre modèle de dialogue H-M a été conçu. Nous avons montré que le système de dialogue GoDIS [13] répond en partie à nos besoins. Ainsi, nous essayons de contribuer, modestement, à réduire le fossé entre les systèmes de dialogue H-M et la communication entre humains.

Cet article suit un plan linéaire : la section 2 expose un bref état de l'art sur les systèmes de dialogue ; la section 3 décrit le cadre expérimental permettant le recueil de notre corpus et son analyse ; la section 4 présente la modélisation de l'agent dialogique découlant de notre analyse avec son implémentation et un exemple de dialogue ; enfin, la section 6 conclut cet article et donne quelques perspectives.

2 Les systèmes de dialogue

Nous nous intéressons particulièrement aux modèles conventionnels du dialogue qui utilisent la notion de tableau conversationnel [16] pour modéliser l'interaction dans le dialogue. Pour préserver la cohérence du dialogue, le tableau conversationnel doit être adapté à l'énoncé courant. De plus, au cours des échanges, des informations vont être partagées pour constituer le *terrain commun* [22], c'est-à-dire l'intersection des savoirs des interlocuteurs.

Plus spécifiquement la théorie QUD (Question Under Discussion) [9] a débouché sur une modélisation informatique GoDIS [13] englobant certains principes issus de la planification du dialogue [1, 23] comme les plans partagés [11]. [11] décrit la théorie DSP (Discourse Segment Purpose) qui propose deux types de relations entre les segments du dialogue : la dominance et la satisfaction-précédence.

2.1 La théorie QUD

QUD [9] est une théorie conventionnelle du dialogue fondée sur une sémantique formelle. L'originalité de ce modèle par rapport à ceux traitant des *questions* comme Groenendijik et Stokhof [10] est de proposer une version structurée d'un tableau de conversations dirigé par la

$$
\begin{bmatrix}
\text{PRIVATE} & \begin{bmatrix} \text{AGENDA} & \text{OpenQueue(Action)} \\ \text{PLAN} & \text{OpenStack(PlanConstruct)} \\ \text{BEL} & \text{Set(Proposition)} \\ \text{NIM} & \text{OpenQueue(Pair(DP,Move))} \end{bmatrix} \\
\text{SHARED} & \begin{bmatrix} \text{COM} & \text{Set(Proposition)} \\ \text{ISSUES} & \text{OpenStack(Question)} \\ \text{QUD} & \text{OpenStack(Question)} \\ \text{ACTIONS} & \text{OpenStack(Action)} \\ \text{PM} & \text{OpenQueue(Move)} \\ \text{LU} & \begin{bmatrix} \text{SPEAKER} & \text{Participant} \\ \text{MOVES} & \text{Set(Move)} \end{bmatrix} \end{bmatrix}
\end{bmatrix}
$$

FIGURE 1 – État d'information de GoDIS

sémantique des questions en contexte. Le but de QUD est de déterminer très précisément les propriétés des couples *Questions-Réponses* (Q-R) (« Issues » en anglais) et de montrer comment les questions et assertions en discussion enrichissent le tableau de conversation. En reprenant l'approche par jeux de dialogue [15, 19], Ginzburg conçoit le dialogue en termes de coups dans un jeu. Cette approche nous semble doublement prometteuse. Tout d'abord, le jeu appelle une conception dynamique du dialogue, non prédéfinie préalablement comme pourrait l'être un scénario pouvant être rejoué à l'identique ; l'issue d'un jeu ne saurait être prédéterminée. D'autre part, le jeu nécessite la participation entière du sujet engagé dans l'interaction.

2.2 Les dialogues fondés sur les *issues*

Fondé sur QUD, GoDIS [13] est un modèle de dialogue computationnel ne gardant qu'une sémantique simplifiée des questions. Il est possible de représenter trois types de question : les questions polaires (?P), questions ouvertes ($?\lambda$x.P(x)) et les questions alternatives (?set(P_1(x), P_2(y), ..., P_n(z))). Les Q-R sont intégrées dans une structure de plans de dialogue qui représentent à la fois la tâche et le dialogue.

Les plans de dialogue, appelés *dialoguePlans* sont des séquences d'actions abstraites. De plus, GoDIS utilise un *état d'information* (IS, « information state » en anglais) proche du tableau conversationnel. Cette structure comprend une partie privée et une partie partagée (voir figure 1).

La partie privée correspond à l'état interne de l'agent (un des interlocuteurs) et la partie partagée définit le tableau conversationnel, qui mémorise l'information partagée par les deux interlocuteurs (i.e. le terrain commun). Des règles de mise à jour et de sélection permettent de contrô-

ler l'IS. Des relations sémantiques sont utilisées pour déterminer les réponses possibles aux questions. Compte-tenu du coup dialogique initial, d'autres coups sont générés en réponse.

2.3 Les stratégies de dialogue

Les stratégies de dialogue [6] gèrent les tours de parole entre l'utilisateur et le système dans le but de conduire efficacement le dialogue. Elles permettent de décider, par exemple, quand le système doit guider l'utilisateur ou pas.

Selon Caelen, le dialogue est une activité conjointe et est dirigé par les buts conversationnels du dialogue. Le dialogue est alors vu comme un processus conduisant à un objectif final. Chaque échange linguistique est une série de tours essayant de remplir des sous-objectifs. Caelen distingue cinq stratégies de dialogue :

- La *stratégie directive* est utilisée pour diriger complètement l'utilisateur.
- La *stratégie réactive* est utilisée quand le système exécute les ordres de l'utilisateur.
- La *stratégie constructive* introduit des digressions pour laisser de côté temporairement l'objectif courant. Cette stratégie est adoptée pour présenter des exemples ou une expérience précédente utile à la situation courante.
- La *stratégie coopérative* ajuste l'objectif courant à celui de l'utilisateur. Le système essaye de s'adapter au but de l'utilisateur en lui suggérant de manière opportuniste des nouvelles informations ou en lui offrant des choix, tout en restant dans le même sujet.
- Finalement, la *stratégie argumentative* apparaît quand l'utilisateur n'est pas d'accord avec l'objectif courant du système. Cette stratégie est principalement présente dans les dialogues argumentatifs. Notre application n'est pas concernée par ce type de stratégie.

2.4 Discussion

Le concept de Q-R peut être considéré comme une spécialisation des jeux de dialogues [12]. Comme le souligne [4], on peut définir des relations intentionnelles entre jeux de dialogues ainsi que des relations sémantiques. Or GoDIS ne définit que la seule relation de dominance entre Q-R et cette relation n'est pas récursive. D'autre part, aucune relation sémantique n'est définie et les différentes Q-R se succèdent par une relation de séquence implicite. Puisqu'aucune contrainte de satisfaction-précédence n'existe, GoDIS permet d'accommoder tout plan de dialogue à tout moment. Il ne permet de traiter que des dialogues simples où chaque plan est défini comme une séquence simple d'actions ou de questions. De plus, sauf pour les phénomènes interactionnels de clarification ou d'établissement du terrain commun, le dialogue ne peut s'éloigner de la tâche pour introduire une digression, une explication ou pour proposer des suggestions à l'utilisateur.

[6] propose la notion de stratégie de dialogue pour choisir la meilleure direction d'ajustement des buts à un moment donné. Deux types de stratégies nous intéressent particulièrement : les *stratégies coopératives* et *constructives*.

Le principal avantage de GoDIS est la notion d'*accommodation* [14] qui apporte de la flexibilité au dialogue. À l'instar de QUD, GoDIS utilise les règles d'accommodation, qui permettent à l'utilisateur de ne pas répondre à la Q-R courante, de corriger une réponse précédemment fournie, de répondre à une question qui n'a pas encore été posée ou de passer à un autre plan de dialogue. Comme le stipule la théorie de l'action conjointe [7], le dialogue n'est pas une activité planifiée, mais laisse la place à l'opportunisme.

3 Recueil de corpus et analyse

3.1 Recueil de corpus

Une expérimentation a été mise en place pour obtenir des dialogues H-H entre des experts et des utilisateurs. Lors de cette expérimentation, un « expert » formé au système CISMeF se retrouve en tête à tête avec un interlocuteur recruté sur la base du volontariat. Ces interviewés sont des membres du laboratoire LITIS (secrétaires, administrateur réseau, doctorants, enseignants-chercheurs). Pour éviter toute connivence entre personnes, les membres du projet les moins familiers avec le laboratoire ont réalisé cette expérimentation en jouant le rôle d'expert. L'intervention de ces deux expérimentateurs a permis de contraster les démarches. Préalablement à l'entretien, chaque volontaire (vingt et un au total) avait été invité à préparer une question d'ordre médical à poser à CISMeF. Celui qui occupe la fonction d'expert dispose d'un accès à CISMeF. C'est lui qui manipule la machine en fonction de ce que lui dit l'interviewé. L'expert doit dans le même temps verbaliser tout ce qu'il est en train de faire. Les dialogues sont enregistrés afin d'être retranscrits et les logs des

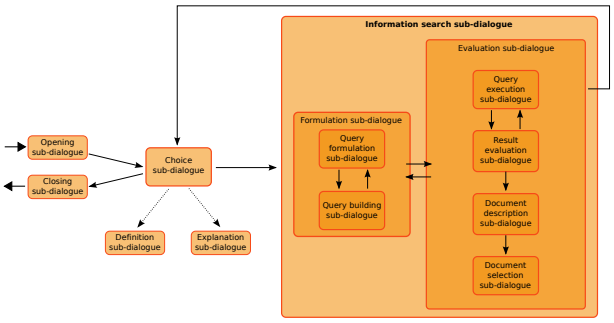

FIGURE 2 – Liens entre les sous-dialogues

requêtes sont récupérés par le système. L'entretien se clôt lorsque la réponse satisfait le demandeur, ou qu'il semble qu'aucune réponse ne puisse être trouvée.

Le corpus est constitué des retranscriptions des 21 dialogues enregistrés des deux experts et les 21 volontaires. Il contient 37 000 occurrences.

3.2 Analyse du corpus

Nous analysons le corpus sur 4 niveaux [18] :
– Une analyse des *sous-dialogues* fait apparaître la structure globale du dialogue comme succession d'étapes.
– Une analyse des *actes de dialogue* (actes de langage contextualisés au sens de [5]) : une liste de ces actes a été établie en fonction des buts illocutoires des énoncés du corpus.
– Une analyse des Q-R. Le formalisme de QUD et GoDIS permet d'analyser précisément notre corpus en termes de Q-R selon le sous-dialogue courant.
– Une analyse de la cohérence : nous sortons QUD du cadre de la théorie sémantique et GoDIS du modèle de Dialogue H-M simple, pour les utiliser dans le cadre plus riche du dialogue H-H.

Analyse des sous-dialogues. Il est possible de décrire la structure globale des dialogues par des enchaînements de différents sous-dialogues.

La figure 2 présente les différents sous-dialogues mis en évidence dans le corpus. Les flèches pleines entre sous-dialogues représentent les transitions naturellement induites par la tâche. Les flèches pointillées représentent une relation de décomposition des sous-dialogues.

Un dialogue débute par une phase d'« ouverture » qui permet d'accueillir l'utilisateur et de présenter CISMeF. Il s'ensuit une phase de « choix de sous-dialogues », qui dans la plupart des cas, conduit à une phase de « recherche d'informations ». Celle-ci est composée d'une ou plusieurs séquences de recherche qui aboutiront chacune à la résolution d'un problème posé par le demandeur. Enfin, le dialogue se termine par une phase de « clôture » généralement quand des documents pertinents sont trouvés.

La séquence de recherche se décompose en deux parties : un *sous-dialogue de formulation* de la requête qui enrichit une requête courante et un *sous-dialogue de test de la requête*. A chaque instant de cette phase, les interlocuteurs peuvent revenir sur la requête courante.

La formulation de la requête se décompose en deux sous-dialogues : un premier *sous-dialogue de formulation* dont le but est de cerner le type de la demande et un second *sous-dialogue de construction de la requête* où la requête est élaborée en coopération avec l'usager. Les termes constituant la requête sont discutés un à un en adéquation avec la terminologie CISMeF.

Suit une phase de test de la requête obtenue précédemment. Plusieurs requêtes sont exécutées lors du *sous-dialogue de lancement de la requête* et les résultats sont présentés au demandeur dans un *sous-dialogue d'évaluation* (quantitative) de la liste des documents, en fonction du nombre de documents trouvés. Cette liste est alors décrite qualitativement au demandeur, dans un sous-dialogue de description de la liste de sous-documents et un ou plusieurs documents peuvent être analysés en détail dans un dialogue de sélection de documents. A tout moment, cette phase de test de la requête peut être interrompue par des demandes de précision.

En plus des recherches de documents, le corpus fait apparaître des demandes de définition de termes et d'explications sur le système.

Les principales caractéristiques de ces sous-dialogues est qu'ils sont opportunistes et que certains sont optionnels. Chacun des sous-dialogues peut être annulé de manière explicite ou implicite. Dans ce cas, le dialogue reprend au sous-dialogue précédent ou revient au sous-dialogue de choix. Des sous-dialogues incidents peuvent apparaître n'importe quand pour assurer la mise à jour du *terrain commun* ou réaliser des digressions (dans ce cas, le dialogue régissant reprend à l'endroit où il a été laissé).

Analyse des actes de dialogue. La théorie des actes de langage [3, 21], a donné une base indis-

pensable à l'étude du dialogue à des fins d'implémentation. Mais elle est insuffisante car elle propose une étude des briques fondamentales sans proposer de modèle pour l'enchaînement de ces actes dans un langage, ni d'intégration du contexte dans la compréhension des énoncés et ceci même dans certains ajouts [24].

Les actes de dialogue [5] ont pour fonction première de modifier le contexte et sont interprétés en fonction du contexte courant. Cette notion nous semble plus adéquate et c'est elle que nous adoptons. Chaque énoncé de notre corpus peut être décomposé en segments associés à un acte de dialogue.

Nous avons établi une liste de ces actes selon une taxinomie issue de [17], elle-même reprise et adaptée de [25].

– Les assertifs initiatifs
  Inform : apporte une information pour laquelle on n'attend pas de réponse.
– Les directifs initiatifs
  RequestInfo : demande d'information.
  Offer : pour proposer quelque chose que l'allocuteur peut accepter ou refuser.
  RequestDirective : le locuteur attend une directive de l'allocuteur.
  Direct : permet de faire faire une action à l'interlocuteur.
  Suggest : regroupe des conseils ou des propositions. Contrairement aux offres, il n'est pas tenu d'être accepté ni refusé.
– Les assertifs réactifs
  Answer : réponse à une question.
  ShortAnswer : réponse à une question sous la forme d'une phrase nominale
  Accept : permet d'approuver l'énoncé précédent qui est réussi et satisfait.
  Refuse : l'énoncé précédent est réussi mais non satisfait.
  Acknowledge : manière de montrer à l'interlocuteur que son énoncé est réussi.
  WantsNothing : rien. Réponse à un *Request-Directif* non satisfait.
– Les directifs réactifs
  icm [1] *:sem*pos* : (perception sémantique positive) demande de confirmation des énoncés.
  *icm :sem*neg* : (perception sémantique négative) le locuteur n'a pas compris l'unité de discours précédente et la fait répéter.
  *icm :per*neg* : (perception négative) signale à l'interlocuteur que l'unité de discours n'a pas été entendue correctement et la fait répéter.
  *icm :per*pos* : (perception positive) signale que l'unité de discours a été entendue correctement et demande confirmation.
  *icm :und*neg* : (non-compréhension pragmatique) le locuteur signale que l'unité de discours est compréhensible mais n'a pu être reliée à un but du dialogue courant ou accommodable. Pour faire reformuler.
– Les déclaratifs
  Bye : pour clore le dialogue et fermer le canal de communication.
  Greet : pour initier le dialogue ou le continuer après une coupure.
  Thanks : permet de remercier l'interlocuteur.
– Les promissifs
  InformIntent : pour préciser à son interlocuteur ce qu'on va faire.

Certains actes de dialogue de notre corpus sont des actes d'établissement [23] : *Accept, Acknowledge, WantsNothing, Confirm, Refuse*.

Analyse des Q-R. À chaque sous-dialogue correspond un ensemble de Q-R qui font progresser la tâche de construction de la requête. Une Q-R peut être composée de plusieurs actes de dialogue, continus ou non.

La sémantique de toutes ces questions est très complexe comme la Segmented Discourse Representation Theory (SDRT) [2]. Mais nous supposons que pour l'expert et pour le système, ces élaborations ne sont qu'un moyen d'obtenir des termes CISMeF sur lesquels travailler.

Dans cette sous-section, nous présentons seulement les principales Q-R issues de notre corpus. La liste n'est pas exhaustive mais une présentation complète peut être trouvée dans [17]. Elles sont décrites en logique du premier ordre.

– Dans un *sous-dialogue de choix*
  Issue(x) : (x est une Q-R) pour initialiser une Q-R.
– Dans un *sous-dialogue de formulation de la requête*
  InitialQuery(x) : (x est une requête) pour formuler une requête initiale qui peut utiliser des termes CISMeF ou pas.
  Precisions(x) : (x est un terme) pour obtenir des précisions à propos d'un terme de la requête initiale
– Dans un *sous-dialogue de construction de la requête*
  AddKeyword(x) : (x est un mot-clé) pour ajouter ou proposer un mot-clé.
  Quand une requête n'est pas satisfaisante,

1. GoDIS utilise des actes de dialogue appelés icm (Interactive Communication Management)

l'utilisation d'hyperonymes ou d'hyponymes permet d'élargir ou de préciser la requête.
AddHypernym(x) : (x est un mot-clé) pour ajouter un hyperonyme.
AddSynonym(x) : (x est un mot-clé) pour ajouter un synonyme.
Quelquefois la suppression d'un terme est nécessaire pour avoir plus de résultats.
DeleteTerm(x) : (x est un terme) pour supprimer un terme.
Comme pour les mots clés, il peut être nécessaire d'ajouter des metatermes ou des qualificatifs avec, respectivement, *AddMetaTerm(x)* et *AddSubheading(x)*.

– Dans un *sous-dialogue d'évaluation des résultats*
NbDocuments(x) : (x est un entier) pour savoir le nombre de résultats.
TooManyDocuments : il y a trop de documents et la requête doit être précisée.
NotEnoughDocuments : il n'y a pas assez de documents et la requête doit être élargie.
– Dans un *sous-dialogue de sélection de document*
Les Q-R concernant la description des documents sont :
DocumentTitle(x,y) : (x est le numéro du document et y est son titre) pour le titre du document.
DocumentKeyword(x,y) : (x est le numéro du document et y est un mot-clé) pour les mots-clés du document.
DocumentTheme(x,y) : (x est le numéro du document et y est son thème) pour le thème principal du document.
DocumentResourceType(x,y) : (x est le numéro du document et y est son type de ressource) pour le type de ressource du document.
InterestingDocument(x) : (x est le numéro du document) pour dire que le document x est intéressant.
– Dans un *sous-dialogue de clôture*
Souvent, les dialogues finissent avec un résumé de l'expert, synthétisant la recherche.
SearchAbstract(x) : (x est un texte).

3.3 Analyse de la cohérence des Q-R

Après avoir présenté les différentes Q-R du corpus, nous définissons leurs relations de cohérence.

Dans la théorie QUD, une question est ajoutée dans le champ *issue* de l'IS une fois qu'elle a été posée. Dès que la réponse est reconnue, la question est alors supprimée de la pile. GoDIS a ajouté l'opportunité de garder la question dans la pile de manière à ce que l'interlocuteur puisse changer sa réponse grâce à la réaccommodation. Cependant, si l'interlocuteur refuse de répondre, la question est supprimée de la pile.

Cependant, certaines questions acceptent plusieurs réponses. La première réponse est donc résolvante mais ne clôt pas *de facto* la question sous-jacente.

[*Expert : Pouvez-vous être plus précis ?*

[*Demandeur : Pour savoir la procédure pour devenir donneur d'organes*

[*D : Et aussi, y-a-t'il des tests à faire ?*

Dans cet exemple, les deux réponses sont satisfaisantes à la question de l'expert. L'utilisation de mots comme « et », « aussi », etc. permet à l'interlocuteur de raffiner sa réponse. GoDIS simplifie ce problème en considérant qu'une seule réponse est acceptable si elle est potentiellement résolvante. Clairement ici, les deux réponses sont acceptables et résolvantes. D'ailleurs, la théorie des questions en discussion précise bien qu'il doit y avoir relativisation pragmatique : c'est toujours en rapport aux buts des interlocuteurs qu'une réponse peut réellement être résolvante. Cependant, d'autres questions n'admettent qu'une seule réponse.

[*D : Rhumatologue /.../ rhumatologie*

[*E : OK, nous allons essayer avec rhumatologie*

Dans l'exemple ci-dessus, le demandeur propose une réponse à la question qui est acceptée, mais qui ne correspond pas à un metaterme CISMeF. Le demandeur répond alors par un nouveau metaterme « rhumatologie » sans que la question ne lui soit reposée. Ici, la première réponse est résolvante, mais l'utilisateur peut toujours proposer une seconde réponse, un autre metaterme. Donner une deuxième réponse à la question doit alors être interprété comme une correction de la première réponse.

Dans certains cas, la réponse à une question est donnée par la personne qui la pose. Dans l'exemple qui suit, l'expert répond à sa propre question. Cela peut être interprété comme l'adoption d'une stratégie coopérative afin d'aider le demandeur. Celui-ci a la liberté de contester la réponse et d'en proposer une autre (phénomène de ré-accommodation).

[*E : Comment traduire cela ?*

[*E : Ce serait finalement des thérapeutiques*

Nous pouvons identifier deux mécanismes d'accommodation qui permettent d'interpréter certains actes de dialogue indirects.

– Accommodation de questions polaires vers des questions ouvertes.
  Cette accommodation se fait lorsqu'une question à réponse polaire est mise en discussion. Dans ce cas, une question ouverte implicite est également mise en discussion. Cela permet d'expliquer le fait que l'interlocuteur peut répondre par une réponse typée.

  [*E : Est-ce qu'il y a une autre question pour étendre la recherche ?*

  [*D : peut-être tendinite*
– Actes de dialogue indirects entre questions et actions
  Comment interpréter qu'une réponse n'est pas un acte de dialogue *Answer* ?

  [*D : Donc je crois qu'on va en rester là non ?*
  Poser la question *?Close* propose une action équivalente de clôture résolue par l'acte de dialogue d'acceptation. Ainsi, si une question polaire est ajoutée et qu'il existe une action associée, alors cette action est aussi ajoutée.

  [*E : Est-ce que tu veux préciser autre chose que ça ?*

  [*D : Non*

  [*E : Non ? donc a priori on n'a pas trouvé de chose précisément là-dessus / mais plus sur les troubles de sommeil en général*

Dans cet exemple, l'utilisateur refuse la proposition. Quand la question principale *RequestInfo(?Precisions)* est annulée, la question implicite *RequestInfo(?λx.Precisions(x))* est également annulée.

Les relations intentionnelles permettent de décrire des liens entre questions, entre la tâche et le dialogue. Nous nous sommes concentrés sur les relations intentionnelles et les mécanismes permettant de passer d'une Q-R à une autre en progressant dans la tâche.

Dans notre corpus, il y a de nombreuses relations de dominance entre les actions de haut niveau liées à un sous-dialogue et des questions. Par exemple, l'action de haut niveau de construction de requête *Action(ConstructionRequête)* domine la question d'ajout de mots-clés *?λm.AddKeyword(m)*.

D'autre part, la relation de satisfaction-précédence est indispensable pour modéliser notre corpus. Les différentes actions et questions déterminent les différents sous-dialogues.

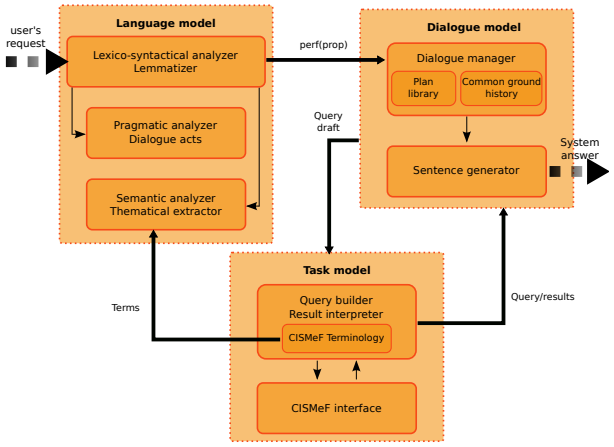

FIGURE 3 – Agent dialogique Cogni-CISMeF

Ainsi, il y a une relation de satisfaction-précédence entre *Action(QueryBuilding)* et *Action(QueryExecution)* dans la mesure où la construction d'une requête valide est indispensable à son action de lancement.

Finalement, lorsque deux questions sont dominées par une Q-R principale, mais qu'il n'y a pas de relation de satisfaction-précédence, elles sont séquencées dans un ordre plus ou moins arbitraire. Cet ordre est dirigé par l'interlocuteur guidant l'échange. Ainsi, dans notre corpus, un des experts utilise peu de stratégies coopératives : il pose une question pour ajouter des mots-clés, puis une autre pour ajouter des qualificatifs, puis une autre pour les metatermes. Cette séquence est révisable et si le demandeur choisit d'ajouter les termes dans une séquence autre que celle proposée, les réponses doivent être accommodées par accommodation globale.

4 Cogni-CISMeF

Pour intégrer les éléments de l'analyse du corpus dans un agent dialogique Cogni-CISMeF, nous proposons une architecture modulaire à trois composantes (voir figure 3) : le modèle de la langue, le modèle du dialogue et le modèle de la tâche. [17] présente une description complète de ces modèles.

4.1 Le modèle de la langue

Cogni-CISMeF reçoit de l'utilisateur un énoncé en langue naturelle et réalise principalement deux analyses : une sémantique et une pragmatique (cf. figure 4).

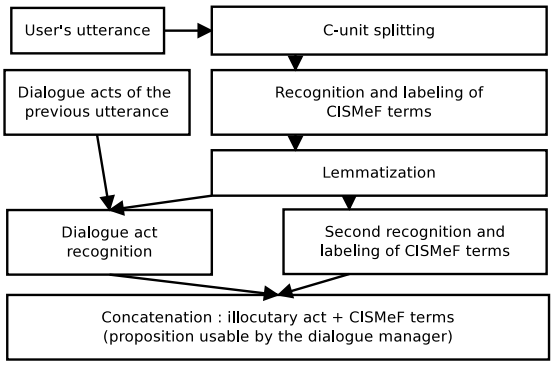

FIGURE 4 – Modèle de la langue

L'analyse sémantique débute par un découpage en *c-unit* [25], l'unité textuelle. Ensuite, les termes CISMeF sont recherchés et étiquetés. Cela est effectué en deux étapes : avant et après la lemmatisation de l'énoncé de l'utilisateur. Les termes CISMeF sont identifiés grâce à une base de données via le modèle de la tâche.

L'analyse pragmatique repose sur un interpréteur d'actes de dialogue. Les différentes règles sont fondées tout d'abord sur des marqueurs linguistiques, tels que le temps et le mode du verbe, des marqueurs d'interrogation, d'acceptation, etc., associés au cotexte immédiat du dialogue (l'énoncé précédent du dialogue). Les règles s'appliquent en trois passes, pouvant modifier le résultat obtenu précédemment. L'annotation est effectuée tout d'abord en utilisant les marqueurs relatifs aux verbes et au sujet grammatical du c-unit. Cette annotation est ensuite corrigée en tenant compte des performatifs découverts dans les c-units précédents.

Finalement, les résultats de ces analyses sont concaténés pour former une proposition utilisable par le gestionnaire de dialogue.

4.2 Le modèle de la tâche

Il encapsule directement :
– la terminologie CISMeF : elle permet d'accéder aux qualificatifs associés à un terme, à la définition d'un terme, aux hyperonymes et hyponymes d'un terme. Elle contient une liste des termes spécialisés à l'usage des patients pour détecter les mots-clés médicaux dans la requête du demandeur.
– un constructeur de requêtes à partir des termes reconnus de la demande.
– un interpréteur de résultats permettant d'affiner la requête, si nécessaire, et englobant diverses stratégies de recherche (par ex : si la requête fournit trop de documents, il faut la préciser en ajoutant des qualificatifs parmi une liste proposée, ajouter un metaterme, utiliser un hyponyme, utiliser des termes patients,...). Il est en relation avec le gestionnaire de dialogue pour ajouter les Q-R pertinentes à la restriction de la requête et proposer des suggestions.
– le moteur CISMeF qui lance la requête construite pour accéder à la base de documents médicaux.

4.3 Le modèle du dialogue

Il comprend un gestionnaire de dialogue et un générateur de phrases. L'approche de GoDIS est très adaptée au type de dialogue que nous voulons modéliser, celui-ci étant basé sur une tâche explicite et demandant de la part des interlocuteurs peu de raisonnement sur leurs intentions. Il utilise une liste d'actes de dialogue qui coïncide avec notre taxinomie. Cependant, ces actes de dialogue ne sont pas complets par rapport à cette dernière. En effet, il manque des actes tels que *Inform*, *Offer* et *Suggest*. Ces actes permettent à l'expert de proposer des informations pertinentes de manière opportuniste selon les résultats de la recherche à un instant du dialogue.

Le générateur de phrase est assez simple puisqu'il se contente de traduire une proposition logique venant du gestionnaire de dialogue en une phrase à trous instanciée par les variables spécifiées. Cependant, le système utilise des mécanismes permettant d'obtenir des productions plus naturelles. Par exemple, un système de génération aléatoire a été mis en place pour varier les énoncés les plus courants : demande de formulation, salutations, acquiescement, etc.

Contrairement au générateur de phrases, le gestionnaire de dialogue est compliqué. Il modélise les sous-dialogues observés dans le corpus sous la forme d'une bibliothèque de plans et gère un état d'information ainsi qu'une représentation du terrain commun. Il repose sur GoDIS auquel nous avons ajouté la modélisation des questions à réponses multiples, la modélisation de l'accommodation de questions et d'actions, des relations d'accommodation pour la prise en compte des relations intentionnelles et enfin, des stratégies dialogiques donnant de la souplesse au dialogue.

Dans la suite, nous ne présentons qu'en partie la librairie de plans et les stratégies dialogiques.

```
PlanA(DocumentSearch,
   (IfThen(not AddKeyword(m))
          (AssumeAction(QueryFormulation),
           AssumeAction(QueryBuilding),
           AssumeAction(QueryDisplay),
           Findout(QuerySatisfaction),
           IfThen(not QuerySatisfaction)
                  (Forget(Done(QueryBuilding)),
                   Forget(Done(AddQuery)),
                   Forget(Resolved(QuerySatisfaction)),
                   Forget(Resolved(?λx2.AddKeyword(x2))),
                   Forget(Resolved(?λx3.AddSubheading(x3))),
                   Forget(not QuerySatisfaction),
                   AssumeAction(QueryBuilding)),
           AssumeIssue(?λx.Documents(x)),
           AssumeAction(EvaluationListeDocuments),
           Findout(?NewSearch),
           IfThen(NewSearch)
                  (ForgetAll, AssumeAction(DocumentSearch))
           Confirm(DocumentSearch)))
```

FIGURE 5 – Plan de l'action *DocumentSearch*

La librairie de plans. Notre modèle utilise deux types de plans (en logique du premier ordre) utilisant « ? » pour représenter les questions.

– les plans de question (au sens QUD) dont le but est de répondre à une question et de renvoyer une donnée (plans appelés *planQ*)
– les plans d'actions qui exécutent une séquence d'actions (*planA*).
– L'ensemble des plans de dialogue utilise une liste d'actions provenant en partie de GoDIS. Par exemple, *Findout(Q)* permet de traiter une question Q en générant un acte de dialogue *Ask()*. Le système pose cette question de manière répétitive jusqu'à sa résolution ou son abandon. *AssumeAction(A)* permet d'ajouter dans l'IS une action. *Forget* et *ForgetAll* permettent de retirer des connaissances de l'IS.

La figure 5 décrit le plan d'action *DocumentSearch* permettant de lancer les trois premières étapes d'une recherche : la formulation libre, la construction et l'affichage de la requête courante. Une question-réponse polaire est ensuite posée (*Findout*) pour que l'utilisateur valide la requête. Si la réponse est positive, le dialogue continue avec le lancement de la requête et l'évaluation des documents retournés. Si la réponse est négative, le système oublie les résultats des étapes précédentes et réinitialise le plan avec l'action *QueryBuilding* pour affiner la requête. Sans sortie explicite de l'utilisateur ou du système, ou une sortie prévue par le plan courant, le système poursuit la séquence de recherche. Toutes ces étapes sont liées par des relations de satisfaction-précédence.

Les stratégies dialogiques. Les *stratégies coopératives* modifient les plans dans le but de proposer des suggestions ou des aides. Le dialogue reste alors dans le même contexte. Le plan courant est toujours actif et le système propose à l'utilisateur de nouvelles informations obtenues durant la recherche. De manière concrète, de nouvelles Q-R (questions, actions, ou propositions) sont construites dynamiquement et ajoutées au plan courant (dans *shared/plan*) tout en gardant les autres champs de l'IS inchangés.

RULE : **ExecCooperativeAction**
CLASS : *PlanExecution*
PRE : { empty(/private/plan), top(/private/plan, CooperativeAction(Q)),
EFF : { P1=content(/shared/com) P2=content(/private/bel) Concatenate(P1, P2, Ps) FindCooperativePlanAction(Q,Ps,PlanAction) Add(/private/plan, PlanAction)

FIGURE 6 – Rule *ExecCooperativeAction*

Une nouvelle action *CooperativeAction* est ajoutée dans le système sous trois formes :

– *CooperativeAction* (?x) pour choisir une nouvelle question à demander à l'utilisateur,
– *CooperativeAction*(action) pour ajouter une nouvelle *taskAction*,
– *CooperativeAction*(p(x)) pour proposer une réponse ou une suggestion à une question en discussion. Le système peut également répondre à une des questions qu'il a soulevée en consultant sa base de données. Ce prédicat peut être ajouté à n'importe quel endroit d'un plan où il est pertinent de proposer des suggestions à l'utilisateur.
– Quand une action est trouvée dans le plan courant, une règle appelée *ExecCooperativeAction* est exécutée (voir Figure 6). Elle exécute la procédure *FindCooperativePlanAction* qui trouve de nouvelles informations à présenter à l'utilisateur. Finalement, la règle ajoute un nouveau plan d'action en haut de la pile *private/plan*.

Par exemple, un des principaux problèmes pour un moteur de recherche est de déterminer le nombre adéquat de documents à présenter à l'utilisateur. Ce problème concerne le plan *ResultEvaluation* (voir Figure 7) qui étiquette le nombre de documents trouvés avec *TooManyDocuments* ou *NotEnoughDocuments*. Cette étiquette est ajoutée à l'IS via l'action de plan *Inform*. Si le nombre est trop grand, le plan appelle le plan *CooperativeAction* pour affiner la requête en proposant des suggestions dynamiquement ajoutées dans *private/bel* et *private/plan*. Symétriquement, un plan est exécuté quand aucun ou peu de documents sont trouvés. La recherche reprend alors en prenant en compte les

réponses de l'utilisateur aux suggestions. Par accommodation de plan, l'utilisateur peut ajouter de nouveaux termes ou soulever de nouvelles Q-R. Les plans *ForgetAction* et *ForgetIssue* ont pour rôle de nettoyer l'IS. Enfin, le plan *AssumeAction* est effectué pour lancer une nouvelle requête. Finalement, si le nombre de documents est correct, il est possible de raffiner les résultats en classant les documents par leur pertinence.

Les *stratégies constructives* ont pour objectif d'apporter des séquences d'explications durant le dialogue. Elles détournent le système de son plan courant : de nouvelles informations ont été acquises qui permettent au système de proposer des digressions, des conseils ou des exemples. Le système tente de lancer de tels plans de digressions à chaque mise à jour de l'IS. Cela est fait grâce à une base de données de règles (dépendantes du domaine) qui analyse proactivement l'IS pour exécuter de nouveaux plans. Si une règle correspond à la situation du dialogue, une nouvelle action ou question (but du plan correspondant) est ajoutée dans *shared/issues* ou *shared/action* pour être exécutée. Le gestionnaire de dialogue trouve alors le plan correspondant et produit les coups dialogiques appropriés. Quand la séquence se termine, la Q-R de la digression est supprimée de *shared/issues*. Le contexte pré-digression est alors rétabli.

Les règles constructives sont donc un type de règle d'accommodation. Contrairement à celle de GoDIS, elles permettent au système d'accommoder ses propres plans plutôt que ceux de l'utilisateur. Ce mécanisme permet de laisser de côté, temporairement, le plan courant.

Dans notre application, le système est capable de proposer à l'utilisateur un certain type de documents même s'il ne l'a pas explicitement demandé. Une règle est alors ajoutée pour proposer certains types de document selon l'identité de l'utilisateur avant d'exécuter la requête. Cette règle est initiée par le plan *CooperativeAction* et ajoutée dans le plan *DocumentSearch*.

```
PlanA(ResultEvaluation,
 (InformIntent(ResultEvaluation),
   AssumeIssue(?λn.NbDocuments(n)),
   IfThen(n isTooHigh)
     (InformIntent(ImproveQuery),
     Inform(TooManyDocuments),
     CooperativeAction(?λx.Issue(x) TooManyDocuments),
     ForgetIssue(),
     ForgetAction(),
     AssumeAction(DocumentSearch)),
   IfThen(n isTooLow)
     (InformIntent(ImproveQuery),
     Inform(NotEnoughDocuments)),
     CooperativeAction(?λx.Issue(x) NotEnoughDocuments),
     ForgetIssue(),
     ForgetAction(),
     AssumeAction(DocumentSearch)),
   Assume(CorrectNbDocuments),
   Confirm(ResultEvaluation)))
```

FIGURE 7 – Plan de l'action *ResultEvaluation*

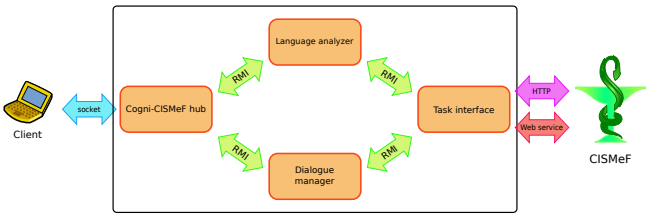

FIGURE 8 – Architecture de Cogni-CISMeF

```
If (exists User(patient) in /private/bel)
And (exists DocumentSearch in /private/issues)
And (neg  exists  RessourceType(x) in  /private/bel)
 Then Launch(Suggest(Plan(Question(RessourceType(x)))))
```

5 Implémentation

La figure 8 présente l'architecture de Cogni-CISMeF composée de quatre parties : le *hub Cogni-CISMeF*, le *gestionnaire de dialogue*, l'*analyse de la langue* et la *gestion de la tâche*. Le *hub Cogni-CISMeF* joue un rôle central. Il gère la connexion avec les clients utilisateurs. Afin de produire la réponse du système, ce module envoie l'énoncé utilisateur à l'*analyse de la langue* qui produit une proposition compréhensible pour le *gestionnaire de dialogue*. Ce dernier génère alors la réponse du système.

Le *hub Cogni-CISMeF* gère une grande variété de clients allant des « lourds » (par exemple en Java) au client web. Notre client utilisateur est une interface web qui comprend un champ texte permettant de saisir et d'envoyer un énoncé, et une zone de texte qui affiche l'historique du dialogue. Cette interface permet également de cliquer sur des propositions de l'expert.

L'objectif du module de *gestion de la tâche* est de maintenir la connexion avec CISMeF. Chaque module peut fonctionner sur son propre serveur. L'architecture de Cogni-CISMeF est en effet distribuée. La communication entre les serveurs peut utiliser plusieurs moyens comme CORBA, socket, etc. Nous utilisons la technologie Java Remote Method Invocation (RMI).

L'objectif du module d'*analyse de la langue* est de regrouper les outils de traitement automatique des langues. Il y a des outils de lemmatisation, de découpage en c-unit, d'analyse

```
CC : Bienvenue sur Cognicismef. Que puis-je faire pour vous ?
% Dialogue d'ouverture + Dialogue de choix
U : donne-moi une définition du paludisme
% Accommodation du dialogue d'ouverture + Dialogue de définition
CC : le terme a pour définition : Encore appelé malaria (...). Maintenant, que puis-je faire pour vous ?
% Reponse a la demande de définition + Retour au dialogue de choix
U : rechercher des documents la-dessus
% Dialogue de recherche de documents
CC : Entendu. Alors ajouter le mot clé paludisme ?
% Etablissement de la question + Dialogue de construction de requête (relation de dependance)
U : oui
CC : OK. Est-ce que vous voulez ajouter autre chose ?
U : y a-t-il des thérapeutiques ?
CC : Très bien. Voulez-vous ajouter le metaterme thérapeutique ou le qualificatif thérapeutique ? Voulez vous une
     définition de 'metaterme' ou de 'qualificatif' ?
% Toujours dialogue de construction de requête + Proposition de dialogue de définition
U : Qu'est-ce qu'un qualificatif ?
% Dialogue de définition (dialogue incident)
CC : Les qualificatifs sont des concepts généraux, qui peuvent être affiliés à un mot clé pour en préciser le sens.
U : ajouter le qualificatif thérapeutique
% Retour au dialogue de construction de requête
CC : OK. Est-ce que vous voulez ajouter autre chose ?
U : non
CC : Entendu. C'est noté. La requête a été ajoutée. Alors voici la requête actuelle : motclé(paludisme),
     qualificatif(thérapeutique). Pour l'instant, la requête vous convient-elle ?
% Dialogue de lancement de requête + Mise en terrain commun
U : oui
CC : Très bien. J'ai lancé la requête. J'ai mémorisé la requête, j'ai trouvé des résultats. Il y a dans cette liste
      11 documents. Il y a trop de documents. On peut essayer des termes plus spécifiques.
% Dialogue d'évaluation de requête + Proposition d'expansion de requête + Succession de stratégies collaboratives
% + proposition de dialogue de construction de requête
CC : Alors êtes-vous patient ou médecin ?
% Accommodation de question
U : patient
CC : On peut essayer d'ajouter des documents spécifiques pour les patients.
% Dialogue de construction de requête
U : oui
% Action associée a la question accommodée : type_ressource(patient)
CC : OK. Alors voici la requête actuelle : motclé(paludisme,thérapeutique), qualificatif(traitement médicamenteux),
      type_ressource(patient) Pour l'instant, la requête vous convient-elle ?
% Mise en terrain commun
U : oui
CC : OK. J'ai lancé la requête. J'ai mémorisé la requête, j'ai trouvé des résultats. Il y a dans cette liste 1
     document. Voici les titres : 1. Modèle de chapitre pour les manuels PCIME - la prise en charge intégrée des
     maladies de l'enfant. Je viens de lancer l'évaluation. Quel numéro de document souhaitez-vous ?
% Etablissement + Dialogue de selection de document
U : le premier
CC : le document sélectionné est : Modèle de chapitre pour les manuels PCIME - la prise en charge intégrée des
     maladies de l'enfant. Alors est-ce qu'on recherche autre chose ?
U : non
CC : Entendu. Cette séquence est terminée. Quelle question vous intéresse ?
% Dialogue de clôture + Retour au dialogue de choix
```

FIGURE 9 – Exemple de dialogue avec Cogni-CISMeF (CC) et l'utilisateur (U).

d'actes de dialogues, etc. Nous utilisons TreeTagger [20] pour la lemmatisation.

Le *gestionnaire de dialogue* est fondé sur Trindikit (version 3.2.2) et GoDIS et utilise le langage SICStus Prolog (version 3.12.2). Il est composé d'un ensemble de règles de mise à jour et de sélection, de règles de sémantique des questions-réponses, d'une ontologie du domaine, des plans liés aux différents sous-dialogues, de la base de règles stratégiques, et enfin du gestionnaire de requêtes. Actuellement, nous avons une version minimale qui nous permet de tester certains phénomènes dialogiques. Une refonte complète de ce module est en cours.

Chacun de ces modules demande encore à être amélioré, mais il est possible de mener un dialogue de bout en bout (figure 9).

6 Conclusion et perspectives

Cet article présente une étude complète de la conception d'un système de dialogue H-M, du recueil de corpus à une implémentation. Nous avons appliqué QUD très peu traitée dans la communauté francophone et intégré l'application GoDIS. Le corpus de dialogues humains a été analysé sur différents niveaux, ce qui a permis de décomposer des dialogues en sous-dialogues liés à la tâche. L'analyse des dialogues en termes de Q-R décrit des relations de cohérence entre sous-dialogues. Les Q-R suivent la tâche tout en offrant la souplesse des dialogues humains. En partant de QUD et de

GoDIS, nous avons montré que ces théories sont des modèles implémentables. Nous les avons enrichies par une sémantique des Q-R et l'utilisation de stratégies dialogiques.

Pour finir notre étude, nous devons valider l'apport effectif de notre système de dialogue H-M. Son évaluation se fera selon le calcul de la plus-value apportée au système CISMeF. Pour les mêmes questions médicales émanant de notre corpus, il s'agira de comparer les requêtes construites par les demandeurs utilisant CISMeF, celles proposées par le conservateur de la bibliothèque médicale du CHU de Rouen et celles construites par Cogni-CISMeF suite à un dialogue avec les demandeurs. Nous mesurerons la plus-value (précision et rappel) respective de Cogni-CISMeF et du conservateur par rapport à la requête seule de l'utilisateur dans CISMeF. Puis, nous étudierons les possibles différences en termes de précision et de rappel entre Cogni-CISMeF et le conservateur. Actuellement, le système n'est pas encore assez abouti pour une telle expérience. Il nécessite, au minimum, d'améliorer l'analyse sémantique en incorporant un lexique conséquent.

Notre volonté de départ de prendre en compte le point de vue de l'utilisateur par la capture et l'analyse d'un dialogue en cours d'action, nous conduit à mesurer la complexité des phénomènes interactionnels inhérents à tout dialogue H-H. Le travail qui reste à accomplir est immense pour permettre, à l'instar d'une navigation sur une plate-forme de jeu, un réel couplage opérationnel entre l'utilisateur et une plate-forme informatique. Mais les conclusions de l'analyse linguistique du corpus, tout comme celles d'un système informatique de dialogue basé en premier lieu sur la question, ne sauraient rester vaines. Elles sont au contraire, pour notre équipe, les marqueurs indispensables de recherches pluridisciplinaires à venir.

Références

[1] J. Allen and C. Perrault. Analyzing intention in utterances. *Artificial Intelligence*, 15(3) :143–178, 1980.

[2] N. Asher. *Reference to Abstract Objects in Discourse*. Dordrecht : Kluwer, 1993.

[3] J.L. Austin. *How to Do Things with Words*. Oxford University Press, Oxford, 1962.

[4] M. Beveridge and D. Milward. Ontologies and the structure of dialogue. In *Workshop on the Semantics and Pragmatics of Dialogue*, Barcelona, 2000.

[5] H. Bunt. Dynamic interpretation and dialogue theory. *Dialogue and instruction*, 1996.

[6] J. Caelen. Strategies of dialogue. In *Speech Technology and Human-Computer Dialogue Conference*, pages 27–42, 2003.

[7] H. Clark. *Using Language*. Cambridge University Press, 1996.

[8] S. Darmoni, JP. Leroy, F. Baudic, M. Douyere, J. Piot, and B. Thirion. Cismef : a structured health resource guide. *Methods of Information in Medicine*, 39 :30–35, 2000.

[9] J. Ginzburg. Interrogatives : Questions, facts, and dialogue. *The Handbook of Contemporary Semantic Theory*, pages 385–422, 1996.

[10] J. Groenendijik and M. Stokhof. *Studies on the semantics of questions and the pragmatics of answers*. PhD thesis, 1984.

[11] B. Grosz and C. Sidner. Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3) :175–204, 1986.

[12] J. Hulstijn. Structured information states : Raising and resolving issues. In *Mundial 97, Formal Semantics and Pragmatics of Dialogue*, University of Munich, 1997.

[13] S. Larsson. *Issue-based dialogue management*. PhD thesis, Department of Linguistics, Göteborg University, 2002.

[14] S. Larsson, P. Ljunglof, R. Cooper, E. Engdahl, and S. Ericsson. Godis - an accommodating dialogue system. In *ANLP/NAACL-2000 Workshop on Conversational Systems*, 2000.

[15] J. A. Levin and J. A. Moore. Dialogue games : Metacommunication structures for natural language interaction. *Cognitive Science*, 1(4) :395–420, 1977.

[16] D. K. Lewis. Scorekeeping in a language game. *Journal of philosophical logic*, 8 :339–359, 1979.

[17] A. Loisel. *Modélisation du dialogue Homme-Machine pour la recherche d'informations : approche questions-réponse*. PhD thesis, INSA Rouen, 2008.

[18] A. Loisel, JPh. Kotowicz, and N. Chaignaud. An issue-based approach to information search modelling : Analysis of a human dialogue corpus. In *International Conference on Text, Speech and Dialogue*, pages 609–616, 2008.

[19] N. Maudet and B. Chaib-draa. Commitment-based and dialogue-game based protocols : new trends in agent communication languages. *The Knowledge Engineering Review*, 17(2) :157–179, 2002.

[20] H. Schmid. TreeTagger - a language independent part-of-speech tagger. *Institut fur Maschinelle Sprachverarbeitung, Universitat Stuttgart*, 1995.

[21] J.R. Searle. *Speech Acts – An Essay in the Philosophy of Language*. Cambridge University, 1969.

[22] R. Stalnaker. Assertion. *Syntax and semantics*, 9 :315–332, 1978.

[23] D.R. Traum. *A Computational Theory of Grounding in Natural Language Conversation*. PhD thesis, University of Rochester, 1994.

[24] D. Vanderveken. *Meaning and Speech Acts*, volume 2 : Formal Semantics of Success and Satisfaction. Cambridge University Press, 1990.

[25] M. Weisser. Spaacy : A tool for annotating dialogue. *International Journal of Corpus Linguistics*, 8(1), 2003.