Dialogue Homme-Machine : Modélisation de multisession

Ngoc-Hoá Nguyen

UNIVERSITÉ JOSEPH FOURIER – GRENOBLE I

SCIENCES & GÉOGRAPHIE

THÈSE

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ JOSEPH FOURIER – GRENOBLE I

Discipline : ***Informatique Système et Communication***

présentée et soutenue publiquement

par

NGUYỄN Ngọc **Hoá**

le 26 janvier 2005

Titre :

DIALOGUE HOMME-MACHINE : MODÉLISATION DE MULTISESSION

Directeur de thèse : M. Jean **CAELEN**

JURY

Mme. Marie-France **BRUANDET**	, *Présidente*
M. Jean Paul **SANSONNET**	, *Rapporteur*
M. Jean-Yves **ANTOINE**	, *Rapporteur*
M. Dominique **NOËL**	, *Examinateur*
M. Jean **CAELEN**	, *Directeur de thèse*

Thèse préparée au sein du laboratoire de Communication Langagière et Interaction Personne-Système,
Fédération IMAG – Université Joseph Fourier – Grenoble I

Kính tặng bố mẹ tôi !

À mes parents qui m'ont toujours soutenu durant mes longues études !

REMERCIEMENTS

Tout d'abord, je voudrais présenter tous mes remerciements, ainsi que toute ma gratitude, à mon directeur de thèse, Jean CAELEN, pour m'avoir accueilli dans l'équipe GEOD, et accompagné au cours des trois années de ma thèse avec ses conseils, son aide, et ses encouragements précieux.

Je tiens à remercier Jean Paul SANSONNET et Jean-Yves ANTOINE, qui m'ont fait l'honneur d'être mes rapporteurs, pour leur lecture attentive et pour toutes leurs remarques constructives sur le manuscrit. Un grand merci à Marie-France BRUANDET et Dominique NOËL, qui ont accepté de participer à ce jury.

Mes remerciements chaleureux s'adressent à Brigitte Meillon, Solange Hollard et Brigitte Bigi, qui ont consacré beaucoup de temps à la relecture du manuscrit avec une grande attention.

J'adresse également mes remerciements à Jean-François Serignat, qui m'a accueilli au sein de l'équipe GEOD et m'a particulièrement aidé au cours des trois années dernières à faire face aux multiples difficultés rencontrées. Merci à Alain Lecomte d'avoir consacré du temps pour relire mon manuscrit et de me donner des conseils appréciés. Je tiens également à exprimer ma gratitude envers toutes les personnes qui ont contribué scientifiquement ou humainement à la réalisation de ce travail de recherche. Merci à tous les membres de l'équipe GEOD (Laurent Besacier, Yannick Fouquet, Mohamed Ahafhaf, Richard Lamy, …) pour leur bonne humeur communicative, pour leur sympathie et pour tous les bons moments que nous avons passés ensemble. Un grand merci à mes amis vietnamiens à Grenoble (Bao-Quoc Ho, An-Te Nguyen, Hung Vo-Trung,…) qui ont partagé des grands moments de pause au cours de ma thèse.

Je pense à mes parents, mes frères et sœurs, qui m'ont apporté un soutien gigantesque, non seulement à l'aspect sentimental, mais également par leurs encouragements, dont j'avais besoin pour mener à bien ce travail.

Un grand merci à tous.

Hoá NGUYEN

ABSTRACT

This thesis proposes a dialogue negotiation model for spoken dialogue systems. Using a multisession dialogue approach, a negotiation is considered as a resolution of the dialogue system for a resource conflict among related users. Thus, a multisession dialogue is defined as a set of successive sessions, each one being a sub-dialogue between a single user and the dialogue system, and divided into three phases: emerging, negotiation process, and notification.

Based on the dialogue strategic model and the game theory, a model of the dialogue generic management has been developed. In this model, the dialogue management activities are achieved by three elements: dialogue act, goal and strategy. A dialogue is therefore composed of different themes, specified in a dialogue agenda. The misunderstanding of user's utterances can be dealt with dialogic markers, called MDI.

The multisession dialogue management is built from this model extended with a multisession dialogue approach. In order to validate our approach, a dialogue system dedicated to the meeting organizing, called Melina, has been built. In the case of a room conflict, speaker can ask Melina to contact all relevant users to begin negotiation and obtain a compromise.

Keywords: *multisession dialogue, dialogue strategy, dialogue goal, dialogue act, dialogue agenda, dialogue theme, human-machine spoken dialogue*.

RÉSUME

Cette thèse présente un modèle de gestion générique dédiée au dialogue oral homme-machine et à l'extension de ce modèle vers le dialogue multisession, qui engage successivement plusieurs utilisateurs et le système de dialogue.

En partant du modèle stratégique de dialogue, nous avons proposé une gestion de dialogue, qui se fonde sur les trois éléments principaux : l'acte de dialogue, le but de dialogue et la stratégie de dialogue. Cela constitue la généricité de notre modèle de gestion. Les activités d'un dialogue sont guidées par l'agenda de dialogue, qui est alors organisé comme un arbre hiérarchique de buts, réparti en thèmes de dialogue. De plus, dans notre modèle de gestion de dialogue, le problème d'incompréhensions concernant le système de dialogue est pris en considération et traité en utilisant deux marqueurs dialogiques d'incompréhension (MDI) et en appliquant des tactiques de traitement des MDIs. Par conséquent, une fois que l'incompréhension est signalée par ces marqueurs, le contrôleur du dialogue peut essayer de trouver la solution adéquate afin d'assurer l'avancement efficace du dialogue.

En se fondant sur cette gestion de dialogue, nous avons envisagé la négociation dans un dialogue homme-machine, sous l'angle de la résolution de conflits entre plusieurs utilisateurs. Le système joue donc un rôle intermédiaire pour résoudre le conflit entre deux ou plusieurs utilisateurs concernant le service visé par ce système. De là, le dialogue multisession est défini comme un ensemble successif de sous-dialogues discontinus et discrets, chacun d'entre eux représentant une session spécifique entre un seul utilisateur et le système. L'enchaînement d'un dialogue multisession est divisé en trois phases différentes. La première phase, dite *Emergence,* représente la session entre un utilisateur (appelé également *demandeur*) et le système, avec l'apparition de conflit entre son but et un ou plusieurs buts déjà satisfaits, manifestés par les autres utilisateurs (dits *patients*). La deuxième phase représente le *processus de négociation* contenant plusieurs sessions différentes entre le système avec ces patients, afin de trouver un compromis favorable pour le demandeur. La dernière phase, dite *Notification,* contient une session unique entre le système et le demandeur, dans le but d'informer celui-ci du résultat de la négociation.

Concernant la gestion de dialogue multisession, nous avons élaboré un module, dit **Expert**, dans le contrôleur du dialogue. Son rôle principal est de gérer l'arbre de buts en conflit avec un double objectif : trouver le meilleur chemin non encore parcouru dans cet arbre et planifier la négociation avec chaque patient dans ce chemin, de manière efficace et rapide. La gestion d'arbre de buts en conflit, ainsi que l'enchaînement des sessions dans le dialogue multisession, ont été explicitement présentés. De plus, nous avons également précisé la stratégie de négociation, de

même que les règles d'initiation d'une session de négociation, en mettant l'accent sur la souplesse au cours du processus de négociation.

Dans l'intention de conforter notre approche théorique, nous avons construit une infrastructure de système de dialogue, qui s'appuie sur une architecture modulaire, et distribuée, divisée en trois couches, avec sept modules principaux : la reconnaissance de la parole, la compréhension sémantique, l'interpréteur pragmatique, le contrôleur du dialogue, le contrôleur de la tâche, le générateur textuel et le synthétiseur de la parole. Cette infrastructure organise effectivement ses modules sous forme d'agents communicants dont l'interaction est effectuée via des messages.

En s'appuyant sur l'infrastructure mentionnée ci-dessus, ainsi que sur nos approches théoriques, nous avons développé un système de dialogue dédié au service d'organisation de réunion, nommé Mélina. Tous les modules de cette infrastructure ont été implémentés, notamment le contrôleur du dialogue équipé d'un *Expert* dédié à la gestion de négociations. Avec la gestion générique de dialogue, Mélina offre effectivement des dialogues souples avec l'utilisateur, tolère des erreurs provenant du module de reconnaissance de la parole… De plus, en cas de conflits, l'utilisateur peut demander à Mélina de négocier avec les utilisateurs concernés, afin de trouver un bon compromis entre eux. L'expérimentation que nous avons faite autour de Mélina a donné des premiers résultats très encourageants, prouvant l'intérêt de nos approches théoriques.

Mots-clés : *dialogue multisession, stratégie de dialogue, but de dialogue, acte de dialogue, agenda de dialogue, thème de dialogue, dialogue oral homme-machine.*

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES EXEMPLES

INTRODUCTION

« Dialoguer est un art de vivre », cet adage est certainement vrai, non seulement dans la vie quotidienne, mais également dans l'aspect communicatif entre l'homme et le système d'information.

Le dialogue oral homme-machine se situe dans un domaine de recherche qui fait intervenir plusieurs disciplines : philosophie, sciences cognitives et sociales, informatique, télécommunication,… Il est souvent conçu à l'image de la communication entre les hommes dans la vie quotidienne et il s'appuie sur des réflexions philosophiques portant sur cette communication. De plus, un système capable de cognition, au même titre qu'un homme, doit être envisagé dans le contexte global engagé entre l'homme et le système d'information. Dans ce contexte, les connaissances et l'information se déroulent et s'échangent entre ces entités par un moyen important : la parole.

D'un point de vue général, le dialogue oral homme-machine se situe dans le domaine de l'interaction homme-machine, qui est sensée représenter une image restreinte de l'interaction entre humains. Le dialogue oral homme-machine permet effectivement de profiter de l'avantage de la parole, en la considérant comme modalité d'entrée/sortie entre l'être humain et un système d'information. Ce fait réside naturellement dans les caractères intrinsèques de la parole portant sur la nature, ainsi que sur la vitesse d'élocution constituant véritablement son efficacité, appréciée non seulement dans la communication humaine, mais également dans la communication homme-machine.

Au sein du domaine du dialogue oral homme-machine, certains problèmes sont pris en considération, et nous encouragent à faire de la recherche approfondie concernant ces problèmes. Nous allons présenter dans les sections qui suivent, tout d'abord la problématique actuelle concernant le dialogue oral homme-machine, ensuite notre objectif et finalement l'organisation de la présente thèse.

Problématique

Actuellement, la recherche portant sur le dialogue oral homme-machine s'efforce de plus en plus de modéliser la capacité de communication humaine dans la machine. Le but ultime de ces travaux vise à améliorer l'efficacité du système de dialogue au point de vue dialogique, c'est-à-dire à diminuer de plus en plus la distance entre un dialogue humain et un dialogue homme-machine.

Sous l'angle du génie logiciel, le système de dialogue est un système d'interaction. Il nécessite en pratique une infrastructure ayant des bonnes primitives, de manière à l'adapter à un ensemble de systèmes différents. Plusieurs infrastructures ont été proposées dont Galaxy Communicator [Mitre, 2002], dédiées à la construction de systèmes de dialogue autour de modules distribués, contrôlés par un noyau central, et avec des communications via des messages. Néanmoins, cette infrastructure souffre véritablement du problème de la complexité et surtout de l'insuffisance d'un gestionnaire efficace de dialogue.

L'exigence d'un gestionnaire efficace dans un système de dialogue mobilise la recherche sur des modèles de dialogue. Ainsi, l'analyse de dialogue humain doit être pris en considération dans le but de l'adapter au modèle de dialogue homme-machine. Des philosophes et psychologues ont proposé beaucoup de théories importantes, jouant un rôle déterminant dans l'apparition des modèles de dialogue.

Le traitement d'erreurs dans un système de dialogue pose également un grand défi, non seulement aux chercheurs dans le domaine de la reconnaissance de la parole, de la compréhension, de l'interprétation… mais également aux chercheurs dans le domaine du dialogue. En effet, des erreurs sont souvent produites au cours d'un dialogue homme-machine pour diverses raisons. A titre d'exemple, la reconnaissance de la parole n'est pas encore parfaite ; les modules de compréhension et d'interprétation ne peuvent envisager tous les contextes et peuvent donc provoquer des incompréhensions, des malentendus…. La défaillance d'efficacité d'un système de dialogue réside véritablement dans la sensibilité à de telles erreurs. Il est certain que le contrôleur du dialogue doit posséder des mécanismes adéquats, afin de surmonter de tels problèmes.

La capacité de négociation d'un système de dialogue est également une demande à considérer. Des travaux menés ces dernières années visent notamment à résoudre ce problème et commencent à produire des résultats. Effectivement, la négociation est la conséquence des conflits de but entre les interlocuteurs au cours d'un échange. Ainsi, l'intérêt de notre travail dans cette thèse réside également dans la résolution de conflits par une gestion efficace de la négociation.

Motivation de recherche : projet « Portail Vocal d'Entreprise - PVE»

Le téléphone joue actuellement un rôle crucial dans la communication quotidienne. L'exploitation de téléphones en service ne cesse croître. Avec les nouveaux services fournis par les opérateurs télécoms, le téléphone ne s'arrête pas d'être seulement un dispositif de communication entre les humains mais ouvre une voie de communication de personne à système d'information. Grâce aux progrès des systèmes de reconnaissance, et de synthèse de la parole, la communication par téléphone de type personne-système a véritablement échappé à la restriction du clavier, c'est particulièrement adéquat à ce type de terminal.

Dans ce champ d'applications avec le téléphone mobile ou fixe, le projet « Portail Vocal d'Entreprise - PVE » a pour objectif de proposer une méthodologie de conception et d'évaluation ergonomique, ainsi que des outils logiciels pour la génération automatique de dialogues. Un démonstrateur a été réalisé, ciblé sur des types d'applications concernant les échanges d'information en entreprise comme : l'organisation de réunion, l'interrogation de l'annuaire du personnel, la gestion de l'agenda des groupes de travail et la gestion personnelle du courrier électronique.

Méthodologie

Comme tout système interactif destiné à interagir avec des humains (qu'il s'agisse d'interface textuelle, graphique, vocale, multimodale, etc.). La conception d'un système de dialogue doit s'appuyer sur une démarche « centrée sur l'utilisateur » dont le parti pris est d'intégrer les études d'usage et d'ergonomie de manière précoce. Ceci permet de garantir une compatibilité maximum du système avec les besoins de ses utilisateurs tout en réduisant les coûts et le temps de conception.

Par ailleurs, un système de dialogue a ceci de particulier qu'il repose sur l'usage de la langue naturelle, sujette à de nombreuses variantes, tant sur le plan des présupposés socioculturels que sur le plan du contenu linguistique. Il convient donc d'intégrer, tôt dans le processus, des analyses linguistiques à propos des actes de dialogue, du vocabulaire, des formes linguistiques, etc.

Le diagramme suivant illustre de façon schématique les principales étapes de la démarche : état de l'art, analyse, modélisation, conception, évaluation et démonstration.

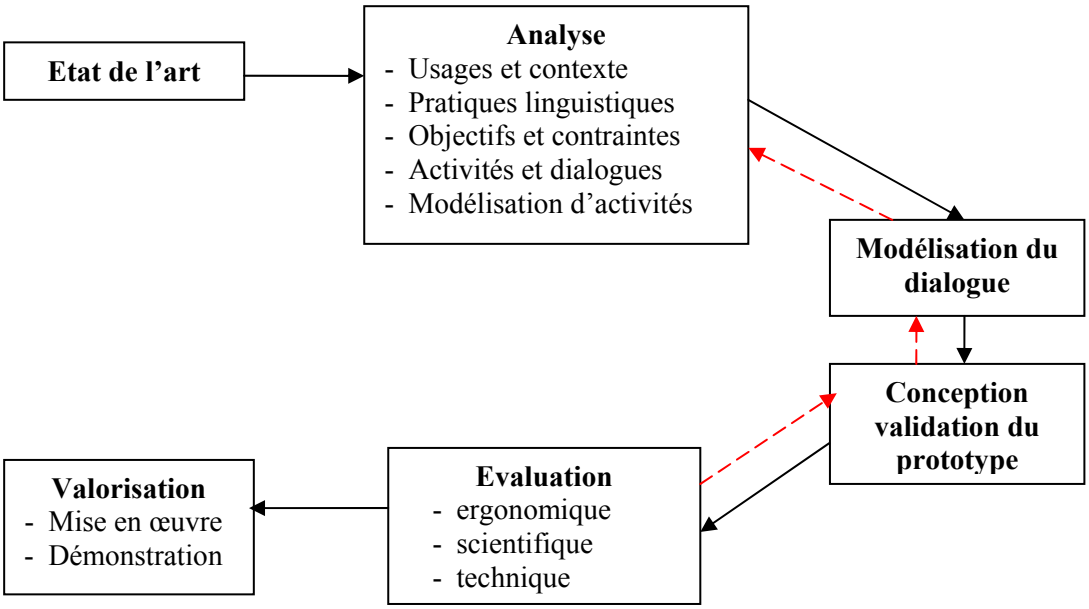


Figure 0.1 : Méthodologie de travail pour le projet PVE

Les flèches pointillées indiquent d'éventuels retours arrière vers les étapes précédentes. Toutefois, l'objectif méthodologique est de limiter au maximum les itérations lourdes en les circonscrivant, dans la mesure du possible, à des échanges locaux au sein du processus de conception/validation.

Analyse d'usage et d'activité

Cette phase d'analyse a comme objectif de recueillir l'ensemble des informations factuelles utiles à l'identification des besoins desquels doit découler la conception d'un système d'information.

Une étude socio-cognitive menée en début de projet a permis de recueillir un maximum d'informations pour la spécification d'un portail vocal qui s'intègre parfaitement dans les pratiques et les habitudes des utilisateurs et de l'entreprise. Elle couvre les aspects suivants :

- Analyse des usages et du contexte :
    - pratiques en cours dans l'entreprise, processus de travail, systèmes (portails) existants, contexte organisationnel, place du vocal,
    - profils d'utilisateurs.
- Analyse des pratiques linguistiques dans l'entreprise : termes et jargon, styles de communication, etc.
- Identification des objectifs visés par le portail vocal et les critères sur lesquels sera évalué le système (acceptation, performance, utilisabilité)

- Identification des facteurs de contraintes pouvant influencer la conception, la mise en œuvre ou l'utilisation d'un portail vocal en entreprise : contraintes techniques, organisationnelles, éthiques, financières, etc.
- Analyse des activités pertinentes pour le projet PVE, ainsi qu'une analyse des dialogues recueillis, a permis d'établir l'ensemble des données nécessaires à la modélisation du dialogue (concepts, vocabulaire, tournures linguistiques, stratégies discursives). L'analyse des dialogues enregistrés permettra de dégager la structure générale du dialogue, et de la tâche.

Des méthodes différentes ont été proposées pour réaliser cette étude comme :

- Observations terrain par l'enregistrement audio de dialogues par téléphone réalisés dans un contexte d'exécution d'activité réelle de personnes (secrétaires de l'université),
- Entretiens individuels sur une population cible, enregistrés dans trois contextes : Centre Hospitalier Universitaire - CHU (médecins, infirmières), entreprises industrielles (commerciaux, chefs de projets, chef de production, …), notaire.
- Simulations en utilisant la technique de magicien d'Oz, pour les activités qui ne sont pas observables en situation réelle.
- Modélisation d'activité pour obtenir des diagrammes d'activité pour chaque service testé.
- Analyse des dialogues via la transcription orthographique des dialogues oraux collectés, puis segmentation de chaque dialogue en « tours de parole » et annotation de chaque tour de parole. Il en est résulté une « grille » d'analyse de dialogue.

Résultat de l'analyse d'usage

L'analyse d'usage [PVE-SR2.1, 2002], que nous avons effectué, dans le cadre de ce projet, dans des hôpitaux, des services d'administration universitaire, chez des professions libérales et des entreprises montre que les services vocaux sont très utiles dans les applications de renseignement, de demande de service d'urgence, d'accès aux dossiers techniques, de secrétariat (par exemple la redirection d'appel, l'organisation d'un rendez-vous à distance, la réservation de salle, l'annulation d'une réunion en urgence, etc.). Le dialogue oral dans ces situations est vraiment utile lorsqu'il s'agit de résoudre un problème en face à face : obtenir un accord, se coordonner pour une action, obtenir une information clef pour débloquer une situation, etc. Cette situation de face à face se retrouve bien au téléphone, moyen privilégié pour la négociation et la résolution de problèmes avec un interlocuteur identifié. En dehors de ces situations, le téléphone n'est pas le moyen de communication le plus prisé :

(a) la messagerie électronique permet d'archiver les messages, de répondre en temps différé, de joindre des dossiers volumineux, de s'adresser facilement et simultanément à des groupes d'individus,

(b) le fax permet de sécuriser ses envois en les authentifiant, etc.

Ainsi, après cette étude les ergonomes nous ont recommandé d'utiliser tous ces canaux de manière multimodale : peut-être commencer par envoyer un message électronique contenant un document attaché, puis téléphoner pour discuter de ce document de vive-voix et enfin confirmer la décision prise ensemble par fax. C'est à ce type de scénario que nous souhaitons apporter des solutions et des briques logicielles pour un système de dialogue homme-machine orienté service, et non plus seulement orienté tâche.

L'analyse d'usage nous a montré que l'utilisateur ne met pas l'accent sur tel ou tel moyen de communication particulier, mais qu'il a besoin d'un service complet : par exemple organiser une réunion comme il le ferait avec une secrétaire, c'est-à-dire, réserver une salle de réunion, s'assurer que le matériel de projection sera opératoire, convoquer ou inviter les participants à la réunion, leur envoyer des documents préparatoires, éventuellement négocier la date de réunion, etc. On se rend compte que dans ces conditions le système de dialogue, qui se suppléerait à une secrétaire, devrait avoir des capacités pour mémoriser les problèmes, pour rappeler plusieurs fois tous les interlocuteurs (les informer de la réunion, leur demander leurs disponibilités de dates, leur confirmer la date et le lieu finalement choisis), pour collecter des demandes ou des contraintes, bref pour gérer des tâches multiples et de haut niveau. Pour ces tâches, le système devrait utiliser les moyens de communication les plus appropriés

Objectifs

En envisageant la problématique ci-dessus mentionnée, nous nous proposons de résoudre les problèmes suivants, certains étant abordés dans cette thèse :

- Spécifier un modèle de dialogue permettant d'envisager un ensemble de tâches différentes,
- Proposer des solutions pour la gestion générique et efficace de dialogue,
- Traiter des incompréhensions au niveau du contrôleur du dialogue,
- Appliquer les premiers résultats à la modélisation et à la gestion de dialogue multisession : les premiers pas pour doter le système de dialogue de la capacité de négocier,
- Construire d'une part une architecture et d'autre part une infrastructure génériques dédiées aux systèmes de dialogue,
- Expérimenter les résultats théoriques en mettant en œuvre un système de dialogue homme-machine.

Synopsis de la thèse

La présente thèse commence par une vue globale destinée à la problématique et à l'objectif de recherche. Elle s'organise ensuite en cinq grands chapitres et se termine par une conclusion, ouvrant sur des perspectives.

Dans le premier chapitre, nous exposons tout d'abord la notion de « *dialogue oral homme-machine* » de manière détaillée. En s'appuyant sur cette notion, la définition d'une application vocale est précisée avec des exemples en utilisant la norme VoiceXML, l'objectif consistant à étudier ses limites ainsi que ses points forts. De là, nous caractérisons ensuite la notion de « *système de dialogue oral homme-machine* » par ses principes généraux, l'architecture générale et les défis portant sur le système de dialogue. En outre, quelques systèmes de dialogue sont également abordés dans ce chapitre.

A la suite de l'architecture générale du système de dialogue, nous passons en revue dans le deuxième chapitre les modèles de dialogue en trois grandes catégories : modèles orientés plans intentionnels, modèles structurels et modèles orientés négociation en considérant le contrôleur du dialogue comme le « *cerveau* » d'un tel système. Certaines remarques et critiques sont également données en présentant quelques modèles typiques de chaque catégorie. Nous mettons l'accent par la suite sur la présentation du modèle stratégique de dialogue, sur lequel se fonde le travail de cette thèse. Dans ce modèle, le dialogue oral homme-machine est considéré comme un jeu qui engage l'utilisateur et le système. Tous les deux interagissent par des actes de dialogue. Plus précisément, l'utilisateur vise à réaliser son propre but, tandis que le système s'efforcer de satisfaire ses exigences. On peut donc dire, dans ce cas, que le dialogue est une activité conjointe et coopérative. Des approches différentes également précisées dans ce modèle permettent au système de bien gérer l'avancement du dialogue : la stratégie, le but de dialogue, le thème de dialogue… Tout cela constitue donc les primitives efficaces dédiées à la gestion générique présentée dans le chapitre suivant.

La gestion générique de dialogue sera proposée dans le troisième chapitre qui se divise en deux grandes parties : la première pour la gestion de dialogue et la seconde consacrée au traitement des erreurs par le contrôleur du dialogue. La gestion de dialogue s'appuie sur les éléments principaux comme acte, but et stratégie de dialogue. De plus, nous proposons le concept d'agenda de dialogue, qui est considéré comme un squelette représentant les connaissances statiques du service visé par le système de dialogue, pour guider l'activité de dialogue tandis que l'information stockée dans chaque élément de cet agenda, réalisé à l'aide du contrôleur de la tâche, constitue ses connaissances dynamiques. Tout cela permet au contrôleur du dialogue de bien gérer l'arrière-plan du système, et de là, la stratégie, ainsi que le but actuel du dialogue de manière générique – le but ultime de notre travail.

En partant du modèle de gestion générique de dialogue, le quatrième chapitre ouvre une nouvelle voie en considérant la notion de *multisession* dans un dialogue. Dans ce chapitre, le problème de la négociation est abordé sous l'angle de la résolution de conflit dans un dialogue via un ensemble de sessions différentes qui engagent plusieurs utilisateurs dans ce conflit. Sous cet aspect, une session se représente donc comme un sous dialogue entre un seul utilisateur et le système et toutes ces sessions constituent donc le dialogue multisession. La modélisation, ainsi que la gestion de dialogue multisession seront présentées en détail dans ce chapitre.

Le cinquième chapitre s'adresse à l'aspect pratique en présentant l'expérimentation de toutes nos approches théoriques. Nous exposons tout d'abord l'infrastructure générique dédiée au système de dialogue avec son architecture modulaire, distribuée et multi-agent. En se fondant sur cette infrastructure, nous présentons par la suite la mise en œuvre de notre système de dialogue nommé Mélina. Les premiers résultats obtenus en effectuant des tests pour Mélina sont également présentés afin d'évaluer et montrer un triple objectif : l'exactitude, l'intérêt et les avantages de notre travail.

Le bilan général de cette thèse est présenté dans la conclusion et diverses perspectives sont également proposées.

Chapitre 1
DIALOGUE ORAL HOMME-MACHINE ET SYSTEME DE DIALOGUE

Dans ce chapitre, nous présentons des notions de base concernant premièrement l'interaction homme-machine, ensuite le dialogue oral homme-machine. L'usage métaphorique de ces termes pose des questions importantes vis-à-vis de la complexité de la communication homme-machine, retenue de celle des phénomènes de communication homme-homme, des capacités qu'un système informatique devrait posséder, afin de devenir un véritable interlocuteur, etc.

1.1 Introduction

L'interaction humaine est naturellement exprimée par de nombreux moyens : la parole, le geste, les clins d'œil, les postures… Par contre, la communication humaine a, en général, pour finalité l'atteinte des buts de chacun des interlocuteurs engagés eux-mêmes dans un contexte communicationnel précis. Selon [Thórisson 95], elle est symbolisée par trois caractéristiques principales : tours de parole en temps réel, entrées/sorties multimodales (parole, geste et autres comportements visibles) et présence des acteurs pour que les tours de parole prennent leur sens en fonction des comportements visuels et sonores des participants. L'exportation de cette notion à ce qui concerne les échanges entre l'homme et la machine (contrôlés par un système d'information) garde ces caractéristiques.

Faciliter les échanges entre l'homme et la machine est donc l'objectif actuel de recherche dans le domaine de communication homme-machine (désormais CHM). En effet, l'objectif primordial de la recherche sur les nouvelles interfaces homme-machine est de trouver des modes de communication entre l'homme et la machine qui permettent d'accroître la robustesse, la fiabilité, la souplesse et bien évidemment la convivialité des interactions et des transactions [Ogden et Bernick, 1996].

L'efficacité de la parole est bien connue. Une personne normale frappe au clavier environ vingt mots à la minute, écrit à vingt-quatre mots à la minute, alors qu'elle parle autour de cent cinquante mots à la minute [Lea, 1980]. Sans aucun doute, la parole constitue toujours un moyen naturel de communiquer.

L'interface homme-machine utilisant la parole comme modalité d'entrée/sortie apporte véritablement beaucoup d'avantages. Il n'est pas nécessaire de se déplacer pour obtenir une information quelconque. On peut envisager de naviguer dans le monde des hypertextes multimédias où les informations sont à la portée de tous en parlant à une machine via le téléphone.

Nous abordons dans ce chapitre des notions de base concernant la communication orale homme-machine avec un système d'information. Des définitions concernant le domaine du dialogue oral homme-machine sont présentées. Nous parlons également des applications vocales qui sont développées en s'appuyant sur la norme VoiceXML (cf. section 1.3) pour rendre compte des avantages appréciables du système de dialogue par rapport à telle application vocale. Quelques systèmes de dialogue oral homme-machine sont également abordés autour de remarques qui ont guidé notre approche.

1.2 Dialogue oral homme-machine

Avant de présenter des connaissances liées au système de dialogue, nous abordons dans cette section des principes généraux du dialogue oral homme-machine, ainsi que ses caractéristiques.

1.2.1 Interaction homme-machine

La recherche dans le domaine de l'interaction homme-machine (IHM) attire beaucoup d'attention non seulement des chercheurs, mais également des entreprises. En général, l'IHM est un domaine très large qui regroupe plusieurs disciplines comme la psychologie, l'ergonomie, la linguistique, les sciences cognitives, les arts graphiques, l'électronique… et surtout l'informatique. Au point de vue informatique, l'IHM peut être considérée comme l'interface entre les mondes informels manipulés par l'utilisateur et le monde formel de l'informatique qui représente un système informatif. Sa nature multidisciplinaire et sa situation par essence constituent donc un écueil dans le processus de développement d'un système informatif en général.

L'interaction entre l'utilisateur et le système informatif peut être effectuée par plusieurs modalités différentes. Bien évidemment, les objectifs des chercheurs dans ce domaine sont d'apporter tous les moyens de communication efficace, du point de vue psychologique, à intégrer au système informatif. En effet, l'être humain perçoit naturellement l'environnement en convoquant les cinq sens : vue, ouïe, toucher, goût et odorat. Bien que la comparaison entre le système nerveux chez l'être humain et la capacité cognitive dans un système informatif soit une chose absurde, cependant, on pourrait de plus en plus équiper de ces cinq sens le système informatif.

Comme l'interaction homme-homme, l'interaction homme-machine dans les systèmes d'information peut se composer de trois parties : perception, action et cognition. L'action est une chose purement informatique. Par contre, la perception et la cognition dépendent forcément des demandes et des contextes de chaque application. La plupart des applications actuelles utilisent des modalités avec le toucher, la vue et l'ouïe comme perception et implémentent la capacité cognitive sous forme de la mémorisation et de l'apprentissage (si nécessaire). Dans le domaine du dialogue oral homme-machine, nous ne nous intéressons donc qu'à l'ouïe qui est bien représentée par la parole, et la cognition dans de telles applications dédiées.

1.2.2 Ingénierie de la parole

La parole est une modalité dédiée à l'interaction homme-machine. Tenant compte des avantages de la parole, au cours des décennies dernières, des recherches importantes dans le domaine du traitement de la parole se sont déjà imposées et ont donné également beaucoup de résultats qui promeuvent non seulement ce domaine, mais également d'autres applications vocales.

En adaptant la structuration de la parole de Schmandt [Schmandt, 1994], nous envisageons celle-ci sous forme de huit niveaux différents comme illustré dans la figure 1.1. Les niveaux plus bas sont donc implémentés dans les moteurs de reconnaissance/synthèse de la parole, tandis que les niveaux hauts résident évidemment dans les applications vocales, les systèmes de dialogue oral (nous abordons ces deux concepts dans les sections ci-dessous). Les deux niveaux « syntaxique » et « lexical » sont considérés à la fois dans les applications vocales et dans les noyaux de traitement du langage.

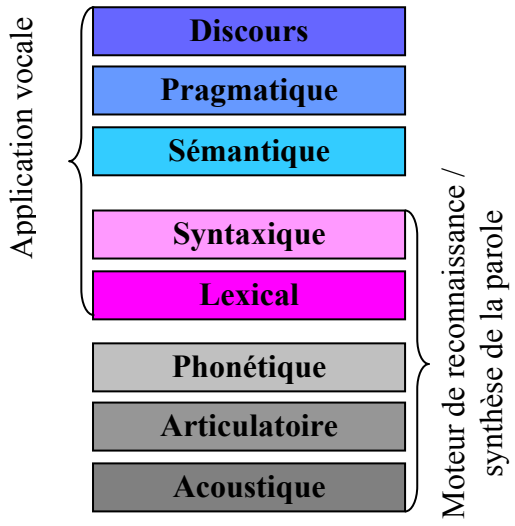


Figure 1.1 : Niveaux de traitement de la parole

Naturellement, la recherche qui s'impose dans le domaine du dialogue homme-machine n'est intéressée que par les cinq niveaux en haut. Cependant, les autres niveaux apportent également des intérêts nettement utiles aux travaux encadrés de ce domaine.

1.2.3 Dialogue oral homme-machine : de la parole à l'action au monde

Intuitivement, le dialogue oral homme-machine (désormais DHM) engage deux interlocuteurs : l'un est l'homme qui représente l'entité à la fois dynamique et interagissante, et l'autre est le système d'information qui est abstrait (résumé par le mot « machine »). Il est le fruit de la rencontre entre les deux domaines : interface homme-machine et ingénierie de la parole (reconnaissance et synthèse de la parole). De manière générale, on peut considérer le dialogue oral homme-machine comme un ensemble d'interactions entre l'utilisateur et le système d'information via l'interface vocale.

L'interaction est caractérisée par des énoncés, soit prononcés par l'utilisateur, soit générés par le système. L'utilisateur parle au système en donnant ses ordres, ses demandes, ses souhaits… et le système lui répond en visant à satisfaire ses intentions.

Une paire de deux énoncés consécutifs de chaque interlocuteur constitue un tour de parole qui est également considéré comme un élément de base du dialogue. Un dialogue oral homme-machine est donc un ensemble de tours de parole entre un utilisateur et le système d'information, avec l'objectif de satisfaire de manière maximale le souhait de l'utilisateur.

1.2.4 Propriétés du dialogue oral homme-machine

De nombreuses différences entre le mode de transformation d'information entre un dialogue écrit et un dialogue oral existent naturellement dans la vie sociale : on peut modifier ce que l'on écrit jusqu'à la satisfaction, et à l'inverse, le lecteur peut relire une phrase, un paragraphe en cas d'incompréhension, de doute, d'ambiguïté. Par contre, quand on utilise la parole pour donner des informations, les erreurs peuvent être corrigées par répétition, autocorrection…, mais non effacées [Minker et Bennacef, 2001]. Une autre différence importante à souligner entre les deux modes de dialogue réside dans le fait que la compréhension d'un dialogue écrit est statique, alors que l'oral est dynamique : il évolue au cours du dialogue.

De plus, un dialogue oral présente généralement des caractéristiques intrinsèques comme la spontanéité, la structure agrammaticale des énoncés et l'apparition de sous-dialogues incidents [Néel et Minker, 1999]. La spontanéité se caractérise par des attributs comme redondance d'information, répétitions, hésitations, contradictions au cours des énoncés. En ce qui concerne la structure agrammaticale, elle est liée évidemment à la langue parlée de l'interlocuteur : l'énoncé doit naturellement respecter des règles de grammaire. La dernière caractéristique concerne l'apparition de sous-dialogues qui servent à clarifier ou reformuler les intentions de l'interlocuteur.

Un dialogue oral homme-machine garde, en bien et en mal, les caractéristiques d'un dialogue homme-homme. Heureusement, sauf en cas d'amusement, l'utilisateur a toujours en conscience les différences entre la communication quotidienne entre les hommes et devant une machine. Il tolère des fautes du système mais pas toujours chez l'être humain.

En ce qui concerne la souplesse d'un dialogue homme-machine, Sabah [Sabah, 1989] a envisagé des aspects en fondant sur des aspects linguistiques, sur la flexibilité, sur l'adaptation à l'interlocuteur et sur le métaraisonnement. Nous reprenons donc ces notions pour viser à réaliser le système de dialogue oral homme-machine.

1.3 VoiceXML et application vocale

La notion d' « application vocale » couvre une zone plus vaste dans l'ensemble des applications informatiques. Nous définissons une application vocale comme une application informatique utilisant la parole pour réaliser/accomplir certaines tâches. Dans ce type d'applications, l'utilisateur peut dialoguer avec l'application en utilisant seulement les mots clés, une phrase courte et simple, … ou toute la complexité de la langue

Nous présentons dans cette section, tout d'abord, la notion de VoiceXML, ses avantages, ainsi que ses inconvénients. Ensuite, des environnements dédiés au développement d'applications vocales sont abordés avec des expérimentations que nous avons effectuées pour montrer les capacités de VoiceXML. Nous donnons, en conclusion, des remarques importantes qui nous motivent pour faire plus de recherches portant sur le système de dialogue.

1.3.1 Introduction

VoiceXML est le nom d'une norme de technologie proposée initialement par le forum de VoiceXML[1]. Elle est basée sur des veilles technologies telles que VoXML de Motorola et de SpeechML d'IBM, pour créer une nouvelle façon d'interagir avec des applications via une interface vocale, en apportant les avantages de développement du WEB aux applications interactives par la parole.

La première version de VoiceXML a été élaborée par AT&T, Lucent Technologies, Motorola, et IBM et approuvée par le W3C en mars 2000. La deuxième version est également apparue avec l'aide des membres du groupe « Voice Browser » du W3C[2].

Au point de vue technique, VoiceXML est considéré comme un langage qui permet d'intégrer aisément la téléphonie et l'Internet. Il s'agit d'un interpréteur (browser) vocal de pages dans une forme dérivée du XML. Un interpréteur de ce type possède une connexion au réseau téléphonique d'un côté, une connexion au réseau Internet de l'autre, des ressources technologiques et un

[1] Site web : http://www.voicexml.com/
[2] Site web : http://www.w3.org/TR/voicexml20/

algorithme pour traiter les pages et interagir avec l'utilisateur. Les ressources technologiques couvrent la majorité de technologies vocales, à savoir la synthèse de la parole, la reconnaissance de la parole et l'annulation d'écho.

L'objectif principal de VoiceXML est premièrement d'apporter tous les avantages de développement de services Web à des systèmes d'application utilisant la parole pour interagir, et deuxièmement de permettre au développeur de programmer et de gérer des ressources au haut niveau. De plus, VoiceXML vise à satisfaire les besoins suivants :

- Minimiser les interactions client/serveur en précisant plusieurs interactions par document,
- Séparer le code d'interaction d'utilisateur (VoiceXML) de la logique (scripts CGI - Common Gateway Interface),
- Favoriser la portabilité de service à travers des plates-formes d'implémentation. VoiceXML est un langage commun pour les fournisseurs de contenu, les fournisseurs d'outil, et les fournisseurs de plates-formes,
- Etre facile à utiliser pour des interactions simples, mais fournir des possibilités pour supporter des dialogues complexes.

Les documents VoiceXML couvrent donc les éléments suivants : sortie pour la synthèse de la parole TTS (Text To Speech), sortie des fichiers sonores, reconnaissance d'entrée parlée, reconnaissance d'entrée DTMF[3], enregistrement d'entrée parlée, contrôle de dialogue et caractéristiques de téléphonie tels que le transfert et la déconnexion d'appel.

1.3.2 Caractéristiques

1.3.2.1 Modèle d'architecture

Le modèle d'architecture d'une application vocale, développée en se fondant sur la norme VoiceXML, est illustré par la figure 1.2 :

[3] DTMF: **D**ual **T**one **M**ulti-**F**requency

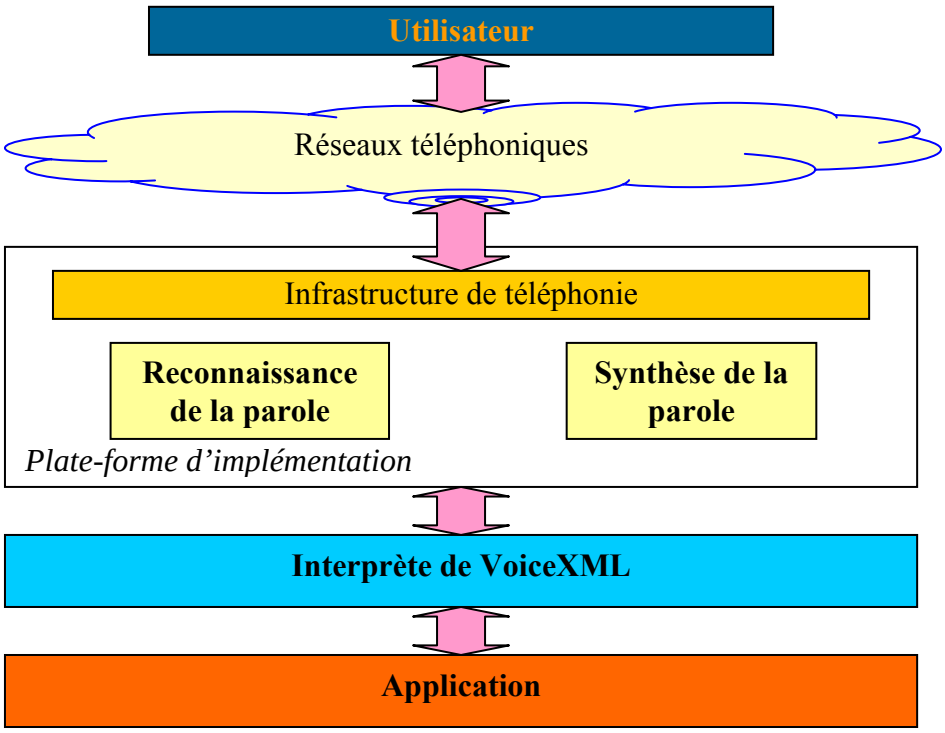


Figure 1.2 : Modèle d'architecture de VoiceXML

Dans ce modèle, la plate-forme d'implémentation a pour but de fournir les primitives dédiées aux événements concernant les actions à la fois d'utilisateur et de système. Elle se compose une infrastructure de téléphonie pour capturer et diffuser les appels téléphoniques, un module de reconnaissance automatique de la parole (ASR – Automatic Speech Recognizer), et un module de synthèse de la parole (TTS – Text To Speech). L'interprète de VoiceXML doit traduire les événements spécifiant dans les documents VoiceXML produits par une application en des actions concrètes sur le monde. L'interprète de VoiceXML doit également assurer la coordination ces actions.

1.3.2.2 Avantages

La norme VoiceXML présente des avantages suivants :

- Réutilisation de qualifications : les développeurs à base des technologies de Web s'accordent pour dire que VoiceXML est facile à apprendre, en raison de sa similitude avec d'autres langages de markup tels que HTML. Leurs compétences, par exemple pour la génération dynamique du contenu, pourront être réutilisées afin développer des applications vocales.

- Facilité de construction : pour une application vocale simple, sa conception ainsi que son développement peuvent être facilement effectués en se fondant sur des environnements

développés de VoiceXML (cf. section 1.3.3). La raison de cette facilité réside aux objectifs posés de cette norme, mentionnés à la section 1.3.1.

- Portabilité : les applications développées en VoiceXML peuvent fonctionner sur une grande variété de plates-formes et peuvent migrer facilement.

1.3.2.3 Inconvénients

Les applications à base de la norme VoiceXML ne sont appropriées que dans les cas où les utilisateurs savent ce qu'ils veulent. L'information qu'ils écoutent est courte et au point, c'est-à-dire qu'elle est seulement constituée de mots clés ou de phrases simples. Cela est donc un grand inconvénient pour l'utilisateur quand il veut exprimer ses demandes par des longues phrases à telle application.

De plus, le dialogue, entre l'utilisateur et l'application, n'est constitué que par des questions/réponses, dans lesquelles l'application garde toujours sa propre initiative (cf. section 1.3.5)

VoiceXML est conçu principalement pour fonctionner avec le téléphone, qui est le dispositif de transmission le plus omniprésent. Néanmoins, les limitations, imposées par le téléphone comme un niveau sonore faible, un taux de bruit élevé, etc., amènent de la faiblesse au module de reconnaissance automatique de la parole, qui peut être prédéfini et fixé dans l'environnement d'exploitation de cette norme.

1.3.3 Environnement développé de VoiceXML

Actuellement, plusieurs fournisseurs de services vocaux s'intéressent à la construction d'une plate-forme générale afin de faciliter le développement d'une application vocale. Au point de vue technique, les environnements de développement de VoiceXML peuvent être divisés en trois catégories :

- Environnement de développement de VoiceXML complet : il comporte presque tous les outils nécessaires pour construire, tester et exploiter une application de VoiceXML : des outils pour construire les fichiers VoiceXML statiques, dynamiques et les grammaires, un moteur de TTS[4] et d'ASR[5], un interprète de VoiceXML, un serveur Web,
- Environnement de développement de VoiceXML fondamental : il comporte alors un outil pour écrire/valider les fichiers VoiceXML et grammaires. De plus, il peut être muni de quelques fonctionnalités pour tester l'application, par exemple il y a peut-être un moteur de TTS mais sans moteur d'ASR,
- Editeur de VoiceXML : Dans cet environnement, on peut écrire/valider les fichiers VoiceXML, des grammaires mais on ne peut pas les tester.

[4] TTS : Text-To-Speech
[5] ASR : Automatic Speech Recognizer

Nous présentons maintenant les deux environnements très puissants et efficaces Nuance-V_builder et IBM Websphere VoiceToolkit, afin de mieux comprendre l'intérêt, ainsi que la limite de VoiceXML.

1.3.3.1 Nuance-V_Builder

Nuance-V_Builder est un ensemble de produits fondé sur des technologies développées par l'université de Stanford en Californie. Ces produits sont rassemblés pour créer un environnement de développement qui permet de construire des applications vocales.

Au point de vue technique, V_Builder est vraiment un environnement de développement de VoiceXML complet. Il comporte un ensemble d'outils nécessaires pour la gestion de projets, l'édition de VoiceXML, de grammaires, l'enregistrement de parole, les moteurs de synthèse et de reconnaissance vocale, l'évaluation des applications…

Les applications développées en utilisant V_Builder sont fondées sur l'architecture distribuée par Nuance, illustrée par la figure 1.3, afin d'apporter la flexibilité, l'efficacité, la fiabilité… aux applications vocales.

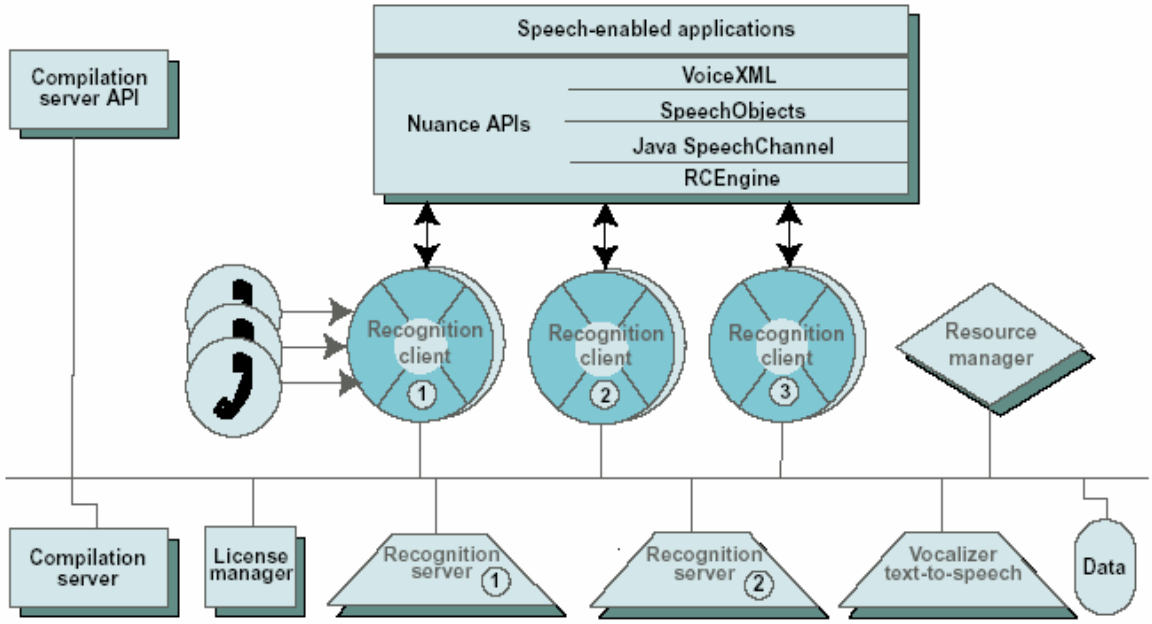


Figure 1.3 : Architecture de la plate-forme Nuance

Les deux modules, qui prennent le rôle le plus important dans cette architecture, sont ceux de reconnaissance de la parole. *Recognition client (recclient)* qui assure l'acquisition des énoncés des utilisateurs et contrôle des dispositifs tel que le téléphone. *Recclient* permet également de supporter le play-back des données audio pré-enregistrées ou générées par le moteur de TTS comme « Vocalizer ». Le module *Recognition server (recserver)* doit évidemment analyser et convertir l'énoncé reçu par le *recclient* en chaîne de caractères. Il est construit à base au modèle acoustique et aux grammaires.

Le point positif de Nuance est la possibilité de supporter plusieurs langages : dans la version Nuance 8.0, il supporte 28 langages comme l'anglais, le français, l'allemand … Fondé principalement sur la grammaire spécifiée sous forme GSL - « Grammar Specification Language » [Nuance, 2001], Nuance est muni également de deux caractéristiques très intéressantes appelées *« Say Anything »* et *« Robust Natural Language Intrepretation - RNLI »* [Nuance, 2001]. Avec *Say Anything*, on peut utiliser un modèle de langage statistique pour l'intégrer et l'utiliser dans le package de reconnaissance. Tandis que *Say Anything* est une capacité utile qui peut conduire à parler librement, RNLI est une autre fonctionnalité de Nuance basée sur la grammaire GSL pour comprendre le sens des phrases parlées.

1.3.3.2 IBM Websphere Voice Toolkit

IBM Websphere VoiceToolKit (VTK), accompagné d'IBM VoiceServerSDK[6], constitue un environnement complet de développement d'application de VoiceXML. Cet environnement est construit sur une plate-forme ouverte nommé Eclipse[7]. Il contient un éditeur de VoiceXML, des grammaires, des outils pour tester/mettre au point une application statique en VoiceXML et une application dynamique basée sur JEE2 (Java Enterprise Edition 2), et un large ensemble de composants réutilisables de dialogue simple.

Les composants principaux d'IBM WebSphere VoiceToolkit sont :

- Editeur de VoiceXML : Son point fort est la fonctionnalité '**Content Assist**' qui permet de générer automatiquement les balises de VoiceXML,

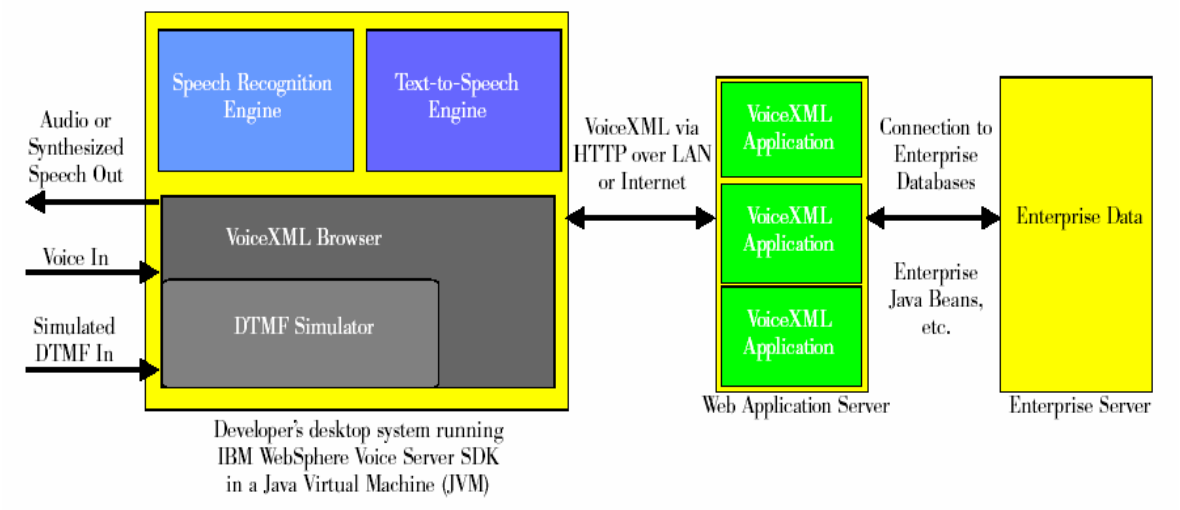


Figure 1.4 : Interaction dans WebSphere VoiceToolkit

[6] Disponible à IBM, http://www-3.ibm.com/software/speech/enterprise/vtoolkit.html
[7] Environnement de développement : http://www.eclipse.org

- Editeur de grammaire : VTK supporte les formats de grammaires comme JSGF (Java Speech Grammar Format) et BNF (Backus-Naur Form),

- Constructeur de prononciation : c'est un composant de VTK permettant de composer l'IPA (International Phonetic Alphabet) basé sur les prononciations de mots inconnus. Ces prononciations composées sont utilisées par le moteur de synthèse de parole d'IBM,

- Enregistrement de son : pour enregistrer les fichiers « .au/.wav »,

- *Reusable Dialog Components* : un ensemble étendu des composants réutilisables de dialogue comme des nombres téléphoniques, heures, dates… Bien évidement, on peut également ajouter d'autres sous-dialogues typiques à cet ensemble. Cette fonctionnalité offre alors plusieurs avantages comme réduire le temps de développement, assurer l'uniformité dans l'interface utilisateur et permettre de manipuler le code qui peut être copié et réutilisé dans les autres applications,

- Débogueur d'applications : qui permet de déboguer pas à pas l'application de VoiceXML.

Avec deux fonctionnalités accompagnées – Database Web Page et JavaBean Web Page - WebSphere VoiceToolkit permet de développer non seulement des pages statiques VoiceXML mais aussi des applications dynamiques de VoiceXML en utilisant les spécifications Java Server Page (JSP), Java Servlet et les requêtes SQL.

En résumé, IBM WebSphere VoiceToolkit représente un ensemble complet et intégré d'outils pour le développement de VoiceXML. Les points dominants par rapport aux autres concurrents sont la facilité de mise au point d'applications vocales, la génération d'application dynamique à partir de JavaBean, de bases de données relationnelles et un ensemble important de composants réutilisables de dialogue.

1.3.4 Expérimentation avec VoiceXML

En nous fondant sur la norme VoiceXML, nous avons développé une application vocale dédiée au service de réservation de salle. L'objectif de réaliser cette application est d'étudier évidemment les capacités de VoiceXML, les offres des environnements développés sur VoiceXML, et naturellement, les avantages, ainsi que les inconvénients de VoiceXML.

Dans cette application, le service de réservation est illustré par un automate comme dans la figure 1.5. L'utilisateur peut téléphoner au serveur développé à base de VoiceXML pour réserver une salle en précisant des informations comme la salle, le temps et le nombre de participants.

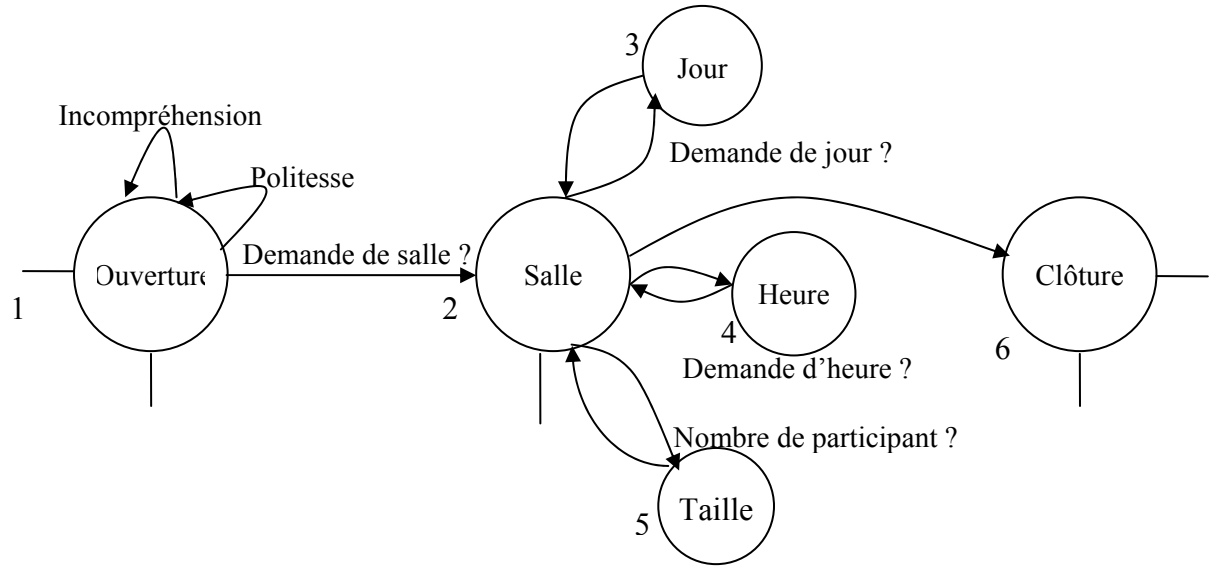


Figure 1.5 : Enchaînement dans un dialogue de réservation de salle.

L'environnement de développement est constitué des deux grandes plates-formes que nous avons présentées au-dessus. Nous abordons maintenant en détail ces deux expérimentations par les deux sections qui suivent :

1.3.4.1 Expérimentation avec IBM VoiceToolkit

IBM VoiceToolkit est un environnement de développement de VoiceXML complet avec des modules nécessaires comme reconnaissance, synthèse de la parole, éditeur de VoiceXML…

La reconnaissance de la parole, la compréhension et de même, l'interprétation de cette plate-forme sont fondés sur la grammaire utilisée le format « JSpeech Grammar Format » (JSGF) [Hunt, 2000]. Le but ultime de la grammaire dans les applications vocales développées en VoiceXML est de bien constituer, de manière claire et simple, un ensemble de règles grammaticales, qui spécifient tous les énoncés possibles prononcés par l'utilisateur devant l'application réelle.

Nous avons donc élaboré notre grammaire simple, illustrée en figure 1.6, pour ce service en appliquant évidemment la norme JSPG :

```
#JSGF V1.0;
grammar reservation;
public <politesse> = bonjour | bonsoir | salut {this.$value= "poli"};
<sujet> = je | on;
<salle> = salle | salles | salle de réunion ;
<acte> = réserver | réserve | réservez| prendre | prend ;
<vb> = voudrais | voudrait| veux | veut;
public <tache>  = [<politesse>] | [<sujet>] [<vb>] <acte> [une|la ]
<salle> {this.$value= "reserver"};
```

Figure 1.6 : Une grammaire JSPG pour la tâche de réservation simple

Dans cette grammaire, nous avons deux règles publiques et cela conduit donc à deux types d'énoncés possibles : l'énoncé de politesse qui concerne des phrases de salutation et l'énoncé de la demande de la tâche précise de l'utilisateur. Des grammaires prédéfinies, concernant l'heure, la date, le nombre …, sont déjà implémentés dans IBM VoiceToolkit et nous n'avons pas donc besoin de les spécifier dans notre grammaire.

Les trois états dans notre modèle, illustré par la figure 1.5, sont traduits en VoiceXML par trois formes différentes : une forme d'ouverture, celle d'échange et celle de clôture. En ce qui concerne les variables dans ce modèle, chacune est définie par un champ spécifié par la balise « ***field*** ». Les valeurs attendues de chaque variable à partir de la réponse de l'utilisateur sont précisées soit par le type, la règle prédéfinie, soit par notre grammaire. Par exemple, la grammaire de demande de réservation utilisée est la nôtre, celle de date/heure est choisie par le type qui correspond bien au type de données.

Des événements relativement à l'attitude de l'utilisateur pour chaque tour de parole peuvent être précisés dans l'application vocale. Dans la nôtre, nous avons capturé des incidents comme suit : l'application ne reçoit pas des signaux de l'utilisateur après un certain temps (`<noinput>`), il parle avec des énoncés hors de la grammaire (`<nomatch>`). Dans ce cas là, l'application va répéter soit son dernier énoncé, soit l'énoncé spécifié dans la balise `<prompt count="2">` du dernier énoncé, soit par un énoncé propre correspondant à cet événement.

Une fois que l'utilisateur prononce bien ce que l'application demande, l'événement `<filled>` est produit et nous pouvons préciser ce que nous voulons traiter avec la valeur acquise dans cette partie.

Le code source de notre application est illustré par l'annexe VoiceXML (cf. annexe A.1). Voici un extrait de dialogue qui est effectué avec notre application dans l'environnement IBM VoiceToolkit :

```
A : Bonjour, vous êtes en connexion avec le service de réservation de
   salle de réunion. Je vous écoute.
U : J'aimerais réserver une salle demain à huit heures
A : Désolé, je ne comprends pas ce que vous dites. Je m'occupe du service
   de réservation de la salle de réunion. Que puis-je faire pour vous ?
U : Je veux réserver une salle
A : pour quel jour s'il vous plaît ?
U : demain
A : le 17 août 2004, pour quelle heure s'il vous plaît ?
U : à huit heures
A : pour combien de personnes ?
U : dix
A : très bien, je réserve la salle de réunion pour vous le 17 août 2004 à
   huit heures pour dix personnes
```

Exemple 1.1 : Un dialogue extrait de l'application vocale construite autour de IBM VoiceToolkit

1.3.4.2 Expérimentation avec l'environnement V_Builder de Nuance

Avec le même objectif, nous avons élaboré une autre application vocale dédiée au service de réservation de salle de réunion, à base de l'environnement V_Builder de Nuance.

Nous avons utilisé la version 1.0 de V_Builder avec des packages de données concernant la reconnaissance et la synthèse de la parole provenant de Nuance également (Nuance v8.0 pour la reconnaissance et Vocalizer v1.0)

Comme tous les environnements de développement VoiceXML, V_Builder utilise des grammaires relativement aux tâches de reconnaissance, de compréhension et d'interprétation. Par contre, l'organisation, ainsi que la structure de sa grammaire, sont tout à fait différentes par rapport la norme JSPG de IBM VoiceToolkit. Dans V_Builder, la grammaire doit être spécifiée sous forme de GSL – Grammar Specification Language [Nuance, 2001]. Les points forts de GSL sont de permettre de spécifier la hiérarchisation de grammaires et de sous-grammaires, de donner des probabilités, et de plus, de construire des grammaires dynamiques, qui peuvent être créées/modifiées au cours de l'exécution de l'application.

A la différence d'IBM VoiceToolkit, V_Builder v1.0 ne contient pas encore des grammaires prédéfinies pour des types courants comme date, heure, nombre… C'est pourquoi nous avons construit des grammaires pour capturer la date, l'heure et le nombre. Tout cela respecte bien évidemment la forme GSL. Voici un exemple d'une grammaire qui exprime la salutation et la demande de réservation de salle de réunion :

```
Salutation [ bonjour salut bonsoir ]{ return ("poli") }

Demande [ cherche cherchons chercher réserve réserver réservons ]

Salle [ salle salles ( salle de réunion ) ]

Vouloir [ veux  voulons voudrais voudrions ]

PrenomPer [ je on nous ]

Main [ Salutation
       ( ?Salutation ?PronomPer ?Vouloir Demande ?[ la une ] Salle )
     ]
```

Figure 1.7 : Une grammaire pour exprimer la demande de réservation

Dans chaque grammaire, on peut récupérer la valeur retournée en utilisant la syntaxe *{return …}* à la fin d'une règle. Chaque règle est organisée par un ensemble de sous-règles, de mots, de patterns (groupe de mots entre les deux parenthèses) qui sont séparés par une espace. Les entités dans une règle peuvent être obligatoires, facultatifs (en utilisant le préfixe *?*) ou répétitif (avec le préfixe *+*). La combinaison de facultatif et répétitif peut également être appliquée. Par des interfaces graphiques conviviales, la construction de grammaires en V_Builder peut donc être effectuée aisément et rapidement.

Le code source de cette application implémentée dans V_Builder est retrouvé dans l'annexe de VoiceXML (cf. section A.2). Voici un dialogue extrait dans des tests réalisés avec cette application dans V_Builder :

```
A : Bonjour, vous êtes en connexion avec le service de réservation de
   salle de réunion. Je vous écoute.
U : je voudrais réserver une salle demain à huit heures
A : pour quel jour s'il vous plaît ?
U : demain
A : le 17 août 2004, pour quelle heure s'il vous plaît
U : à huit heures ?
A : pour combien de personnes ?
U : pour vingt personnes
A : très bien, je réserve la salle de réunion pour vous le 17 août 2004 à
   huit heures pour vingt personnes
```

Exemple 1.2 : Un dialogue extrait de l'application vocale construite par V_Builder

1.3.5 Critiques

Au cours des deux expérimentations avec deux plates-formes de développement VoiceXML, Nuance et VoiceToolkit, nous pouvons formuler les remarques générales suivantes :

- L'application vocale fondée sur VoiceXML ne permet d'accepter que des énoncés simples, prédéfinis à l'aide des grammaires.
- Le déroulement d'un dialogue dans telle application est une instance d'un plan déjà bien prévu, codé par des grammaires et par des balises de VoiceXML. Le tour de parole dans un dialogue ne doit apparaître que successif et l'échange n'est pas effectué de manière simultanée.
- L'utilisateur interagit réactivement avec une telle application vocale : il ne peut pas librement exprimer ce qu'il veut.

Dans IBM VoiceToolkit, la réutilisabilité des règles dans une autre grammaire n'est pas encore possible : la référence à une autre grammaire n'est pas acceptée dans une grammaire. Cela est donc un point faible concernant la construction de grammaires, qui est effectivement un élément décisif influençant directement la compréhension, la souplesse de l'application vocale.

En ce qui concerne V_Builder, il offre des bonnes fonctionnalités, à la fois de conception et d'efficacité pour l'application bâtie à l'aide de cette plate-forme : des grammaires souples, réutilisables, des bonnes performances au niveau de la reconnaissance et de la synthèse de la parole… Néanmoins, cet environnement reste encore limité concernant les inconvénients de la norme VoiceXML.

1.4 Systèmes de dialogue oral homme-machine

L'apparition de la norme VoiceXML, avec ses avantages, ainsi que ses inconvénients, apporte des facilités à la construction des applications vocales. Des environnements de construction des telles applications avec VoiceXML ont été développés au cours de dernières années et conduisent au succès de certaines applications vocales comme des serveurs vocaux dans des opérateurs téléphoniques (Orange, SFR…) dédiés au service de réponse automatique à des questions concernant le message électronique, le répondeur en utilisant des mots simples, des touches téléphoniques.

Pourtant, toutes ces applications ne permettent pas encore à l'utilisateur de communiquer correctement, de manière naturelle comme dans la conversation quotidienne. C'est pourquoi la nécessité et l'exigence de systèmes de vrai dialogue oral entre l'utilisateur et le système ont toujours attirés l'attention à la fois des chercheurs et des entrepreneurs.

Nous abordons donc dans cette section premièrement des notions de base concernant le système de dialogue, ensuite des besoins portant sur un tel système. Et puis, nous présentons également une architecture générale dédiée au système de dialogue oral homme-machine.

1.4.1 Principes généraux

Un système de dialogue oral homme-machine (désormais SDHM, ou par simplicité, système de dialogue) est un système informatique qui est capable d'interagir naturellement (c'est-à-dire d'une façon qui semble naturelle à l'homme, principalement en langue naturelle) avec l'utilisateur principal via des modalités d'interaction vocale pour accomplir une tâche concrète. Evidemment, un système de dialogue est une application vocale mais elle doit être capable de comprendre non seulement les mots clés, mais également des phrases complètes, c'est-à-dire que l'utilisateur peut vraiment dialoguer avec elle.

Un SDHM doit se comporter en respectant bien les trois processus principaux : perception, action et cognition. La perception est bien évidemment traduite au cours de reconnaissance automatique de la parole et de compréhension des énoncés de l'utilisateur. Tout ce que le système peut comprendre par sa propre manière formelle conduit directement l'action du système dans son propre monde. Evidemment, l'action du système peut satisfaire ou non les souhaits de l'utilisateur et donc le processus de cognition est exigé afin d'assurer une satisfaction maximale si possible.

Les composants minimaux d'un SDHM doivent se charger des tâches comme la reconnaissance, la synthèse de la parole, la compréhension, la coordination des tours de parole ou bien le contrôle de dialogue et la manipulation de tâches élémentaires ou bien le contrôle de tâche. Nous détaillons maintenant ces composants dans la section suivante décrite l'architecture générale d'un SDHM.

1.4.2 Architecture générale

En utilisant les principes abordés à la section précédente 1.4.1, nous proposons une architecture générique dédiée au système de dialogue, illustrée par la figure 1.8.

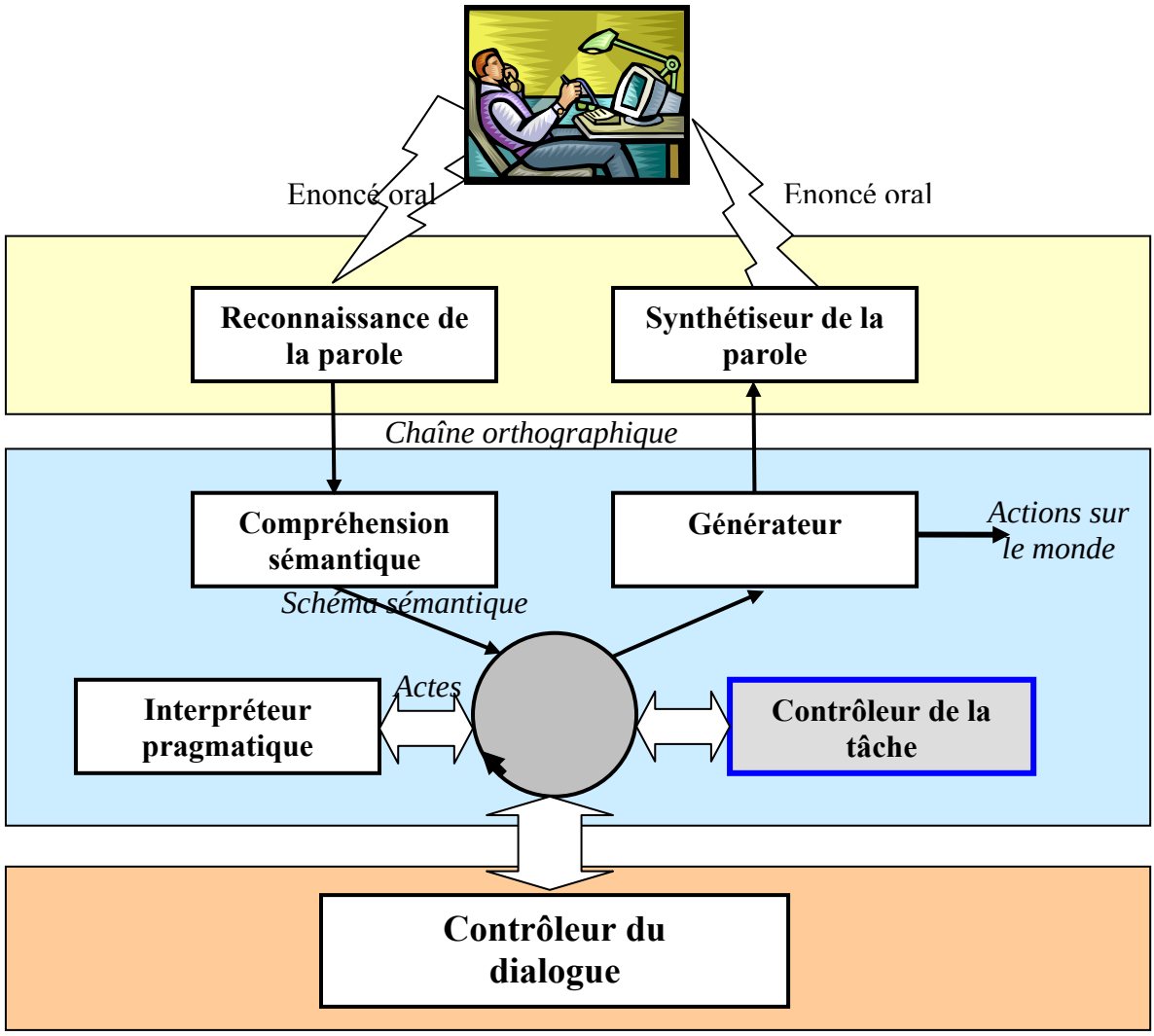


Figure 1.8 : Architecture générale d'un système de DHM

L'objectif principal de notre architecture est de séparer le plus nettement possible les composants d'un SDHM, afin que nous puissions les manipuler aisément. En général, nous visons effectivement deux caractéristiques principales : distribué et modulaire. Chaque composant doit être modulaire pour que des changements d'un des autres ne l'affectent pas. Pour la même raison, ils sont distribués en tirant profit de la puissance de plusieurs machines différentes. Les défis du système de dialogue oral homme-machine présenté dans la section suivante sont considérés en donnant à cette architecture des facilités de mise en œuvre.

Notre architecture se compose de sept modules différents. Nous abordons maintenant ces modules en détail :

1.4.2.1 Reconnaissance automatique de la parole

Les signaux sonores que l'utilisateur prononce arrivent au système et sont capturés par des interfaces spéciales (une carte téléphonique, une carte de son…). Ces signaux sont transmis au module de reconnaissance automatique de la parole afin de les convertir en une chaîne de caractères. Cette chaîne orthographique est transmise immédiatement au module de compréhension sémantique.

Normalement, le module de reconnaissance de la parole peut retourner une liste de n meilleures chaînes orthographiques qui représentent les meilleures candidates reconnues par ordre de ses *scores de confiance*. Le score de confiance est calculé directement à partir des scores acoustiques de chaque mot dans la chaîne de résultat et caractérise donc cette chaîne. Au niveau le plus bas, le score acoustique d'un mot est effectivement mesuré à partir du score de phonème.

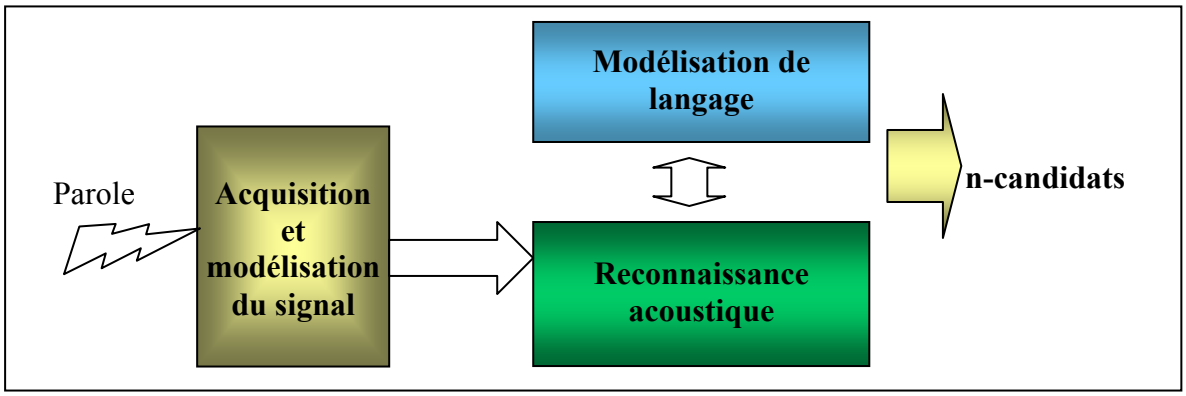


Figure 1.9 : Description d'un module de reconnaissance de la parole

Un module de reconnaissance se compose normalement de trois composants principaux illustrés ci-dessus. Le premier est pour acquérir le signal sonore de l'énoncé de l'utilisateur et le modéliser sous une forme généralement fréquentielle en gardant des paramètres pertinents. Ces paramètres sont utilisés dans le composant de reconnaissance acoustique qui identifie les sons présents dans le signal. La reconnaissance acoustique est effectuée en utilisant normalement la modélisation par modèles de Markov cachés [Rabiner et Juang, 1989] concernant des phonèmes, diphones, syllabes, etc. Il est ensuite nécessaire de mettre en correspondance une suite d'éléments acoustiques avec une forme lexicale en utilisant le composant de modélisation de langage. Il permet donc de spécifier le positionnement d'un mot dans l'énoncé de l'utilisateur par différentes techniques de modélisation à base soit de grammaire, soit de statistique, soit la combinaison des deux [Wang et al., 2000]. Selon la demande de chaque application, le résultat obtenu peut être soit une chaîne textuelle, soit une liste des n meilleures chaînes textuelles.

En ce qui concerne la dépendance à l'utilisateur, actuellement, il existe plusieurs moteurs de reconnaissance ayant un modèle acoustique déjà adapté aux utilisateurs, à l'environnement, au canal de communication. Cela apporte donc l'indépendance du locuteur à l'application vocale.

1.4.2.2 Compréhension sémantique

Le module de compréhension sémantique a la charge de caractériser la chaîne de caractères envoyés par le module de reconnaissance de la parole. Il doit l'analyser pour produire un schéma sémantique qui symbolise ce que l'utilisateur vient de prononcer au système. Comme pour un humain, il est certain qu'il y aura des erreurs à ce niveau là : le système (ce module) ne peut tout comprendre ou peut les comprendre de manière différente…

La compréhension doit être effectivement effectuée par l'analyse syntaxique et sémantique. L'analyse syntaxique est à base de grammaire formelle, de grammaire transformationnelle, ou de grammaire en chaîne… Plusieurs approches différentes sont également utilisées au cours des analyses sémantiques comme grammaire sémantique [Bonnet, 1980], grammaire de cas [Fillmore, 1968], grammaire fonctionnelle [Bresnan, 1982]…

1.4.2.3 Interpréteur pragmatique

Le schéma sémantique sorti du module de compréhension arrive ensuite dans un central qui coordonne les modules principaux du système : interprétation, contrôleur du dialogue et contrôleur de la tâche. Ces trois modules peuvent mutuellement interagir afin d'obtenir des données nécessaires pour l'action.

Le central transfère tout d'abord le schéma sémantique au module d'Interpréteur pragmatique et attend des actes de dialogue en réponse. Ce module doit résoudre des problèmes concernant la référence (noms propres, expressions de désignation, anaphores, déictiques, etc.), les présuppositions et les implicatures conversationnelles… en se fondant sur des connaissances de l'historique (acquises par les tours précédents de parole), des référents de contexte obtenus en interagissant avec le contrôleur de la tâche. Le central retrouve à la sortie de ce module des actes de dialogue qui sont ensuite pris et traités par le module de contrôleur du dialogue.

1.4.2.4 Contrôleur du dialogue

Le contrôleur du dialogue assure effectivement un rôle important dans un système de DHM :

- coordonner toutes les interactions entre l'utilisateur et le système,
- assurer l'avancement, la cohérence du dialogue,
- être le pont qui relie la parole et l'action réelle dans le monde.

En effet, l'acte de dialogue sorti du module d'interpréteur pragmatique est analysé et traité par le contrôleur du dialogue. En s'appuyant sur les tours précédents de la parole, il doit déterminer le but souhaité par l'utilisateur. Afin de bien conduire le dialogue au but posé, il doit également calculer la stratégie appliquée pour générer l'acte (l'action et de même, la réponse) du système (cf. chapitre 2). A la sortie de ce module, on trouvera l'acte de dialogue du système et des données supplémentaires qui sont tous transférés au module de génération. Bien évidemment, des

interactions nécessaires au contrôleur de la tâche peuvent être invoquées au cours de ces calculs pour qu'il puisse avoir suffisamment de connaissances ciblées sur l'action du système.

1.4.2.5 Contrôleur de la tâche

Le contrôleur de la tâche est un module concernant purement l'application réelle. Il doit représenter des tâches, des services visés par le système. Au point de vue de la conception logicielle, nous considérons ce module comme une application réelle enrichie par les interfaces entre cette application et les composants du système de dialogue. Toutes les actions, visées par le système, doivent être traitées dans ce module. Il contient donc des objets, des données, des connaissances… indispensables au système visé. Toutes ces informations sont servies à d'autres modules en cas de nécessité via des commandes du central.

1.4.2.6 Générateur textuel

L'acte de dialogue, généré par le contrôleur du dialogue, est pris par le module de génération textuelle. Ici, il traduit cet acte en une chaîne de caractères et/ou des actions concrètes qui y sont codés. Bien évidemment, la génération peut avoir besoin des données supplémentaires comme le but souhaité par l'utilisateur, la stratégie de dialogue, des données détaillées stockées dans le contrôleur de la tâche… La chaîne de caractères est ensuite transférée au module de la synthèse de la parole.

1.4.2.7 Synthétiseur de la parole

Ce module n'effectue que la conversion d'une chaîne de caractères en des signaux sonores représentant sa prononciation. La synthèse de la parole est alors définie comme la production automatique de la parole, grâce soit à une transcription graphème-phonème des phrases à lire, soit à la concaténation des morceaux préenregistrés.

Les signaux sonores de sortie seront conduits vers les dispositifs visés par le système de dialogue, par exemple les haut-parleurs, le téléphone, etc.

1.4.3 Défis d'un système de dialogue

Avec une telle architecture du système de dialogue oral homme-machine, les défis sont : exactitude, fiabilité, flexibilité, adaptabilité, extensibilité et généricité.

- L'exactitude est une exigence très importante envers le système de dialogue oral homme-machine. Elle est manifestée par la cohérence entre un tour de parole (la réponse de la machine doit être adéquate par rapport à la question posée par l'utilisateur) ainsi que la cohérence dans tout le dialogue. Cela demande donc l'exactitude de tous les modules dans le système : le module de reconnaissance doit bien produire le texte de l'énoncé de l'utilisateur, la compréhension est obligatoirement assurée avec d'excellents résultats, l'interprétation doit

résoudre tous les problèmes pragmatiques… Nous espérons donc que, dans un temps proche, avec des progrès dans tous les domaines concernés, toutes ces exigences pourront être satisfaites.

- La fiabilité réside dans le contrôleur de la tâche si l'on ne compte que les données au niveau de l'application. La flexibilité doit être exigée principalement des modules de compréhension et de synthétiseur. De plus, le système doit s'adapter non seulement aux utilisateurs, à leurs exigences, mais également au contexte différent (environnement, langue, etc.) et l'adaptabilité est donc une exigence à envisager dans tous les modules du système.
- Une autre exigence qu'on ne peut ignorer est l'extensibilité. Ce défi n'est pas encore respecté dans la plupart des systèmes actuels de dialogue qui sont dédiés seulement à des tâches prédéfinies. Les deux modules qui peuvent réaliser un tel défi sont évidemment l'interprétation et le contrôleur du dialogue s'ils sont génériques.
- Le dernier défi qui peut être considéré comme la synthèse de tous les défis précédents : la généricité. En effet, tous les modules dans un système de dialogue doivent être envisagés sous l'angle de généricité afin d'assurer les exigences mentionnées au-dessus. Cependant, dans le cadre de notre travail, nous concentrons surtout la généricité au niveau du contrôleur du dialogue en souhaitant ainsi surmonter tous les autres défis vis-à-vis de ce module.

1.5 Etat de l'art

Nous présentons dans cette section quelques systèmes de dialogue. Bien évidemment, chacun a ses propres points forts, ainsi que des points faibles mais en général, ils sont dédiés à la tâche réelle qu'ils visent à réaliser, donc non génériques.

1.5.1 COALA (COAdaptation Langagière pour l'Apprentissage)

COALA (COAdaptation Langagière pour l'Apprentissage) [Lehuen, 1997] est un système de dialogue qui a pour but l'aide documentaire dans une médiathèque municipale. Bien que l'interaction entre l'utilisateur et ce système ne soit que des énoncés écrits, son dialogue écrit présente des points forts par des mises en œuvre des propres techniques bien ciblées. En se fondant sur le modèle *hypothético-expérimental* [Lehuen et al, 1996], celui qui s'est bâti sur les deux notions essentielles « *dynamicité* et *erreur* » en entretenant avec les deux caractéristiques recherchées concernant l'interaction et l'apprentissage, COALA répartit structurellement ses connaissances en cinq catégories différentes. Ces catégories sont représentées par cinq « experts » correspondants : l'expert dialogueur assure la gestion du dialogue ; l'expert interprétateur concerne la catégorie d'interprétation ; l'expert tacticien s'occupe de la méta-expertise d'acquisition des connaissances ; l'expert bibliothécaire surveille le progrès de la tâche sous-jacente ; et le dernier expert est assigné à l'usager par l'abus de langage en considérant qu'il participe activement lui-même au processus de résolution de son problème.

L'expert usager interagit avec le système COALA en lisant et écrivant dans une fenêtre d'interface pour exprimer en français sa recherche documentaire. COALA pose des hypothèses au fur et à mesure du dialogue, et les valide ou les infirme selon la suite des événements. Conceptuellement, il effectue aussi l'étape d'apprentissage au cours de chaque tour de parole afin d'enrichir de plus en plus ses connaissances.

Bien que le modèle hypothético-expérimental vise à faire acquérir des connaissances au système de dialogue avec, pour objectif, d'interpréter correctement les énoncés de ses utilisateurs, il ne décrit pas encore explicitement comment on pourrait construire des méta-connaissances [Nicolle, 1997]. Par cette limite, d'autres objectifs de ce modèle tels que « non-dédié », « ouvert » rencontrent des difficultés lorsque l'on veut réutiliser ce modèle à partir de COALA. La généricité n'est donc pas satisfaite comme prévu dans ce modèle.

1.5.2 HALPIN (Hyperdialogue avec un Agent en Langage Proche de l'Interaction Naturelle)

HALPIN (pour Hyperdialogue avec un Agent en Langage Proche de l'Interaction Naturelle) est un système de dialogue dédié la recherche d'information en utilisant l'hyperdialogue en français [Rouillard, 2000]. La notion d'*hyperdialogue* est définie comme un dialogue homme-machine qui est coopératif et finalisé dans un environnement hypertextuel avec la combinaison éventuelle de modalités de communication comme l'écrit, l'oral, le geste. L'usager dialogue avec le système HALPIN, pour l'instant, par deux voies : soit qu'il énonce sa demande de recherche des documents oralement et la machine lui répond en donnant le résultat de recherche sur le même mode ; soit qu'il fait tout cela par écrit.

Dans HALPIN, la compréhension des énoncés de l'utilisateur est réalisée par la reconnaissance des concepts. Le processus de reconnaissance des concepts est inspiré par des travaux de Brun [Brun, 1998], qui compose en deux phases : premièrement identifier les concepts par des mots isolés et les remplacer par leur concept pertinent ; et deuxièmement, remplacer des groupes de concepts par un concept d'ordre supérieur. De manière générale, la compréhension est fondée strictement sur l'arbre des concepts, des groupes de concepts.

A partir des concepts sortis de la phase de compréhension, ces concepts constituent l'acte de dialogue de l'utilisateur par une grammaire sémantique à balayage gauche-droit avec seulement quatorze actes prédéfinis « *Ordonner ; Suggérer ; Confirmer ; Infirmer ; Demander ; Aider ; Saluer ; Insulter ; Remercier ; Alerter ; Justifier ; Commander ; Promettre ; Informer* ». Cela crée donc un point faible d'interprétation de HALPIN à la fois d'extensibilité et de réutilisabilité.

Au niveau du contrôleur du dialogue, il a déjà adapté la stratégie de dialogue décrite dans [Caelen 1997] à son propre problème en gardant trois stratégies : directive, réactive et coopérative. Le résultat de cette adaptation conduit à une nette amélioration du dialogue dans HALPIN à la fois en souplesse et en efficacité.

Un de ses points forts que nous pouvons citer est de tirer profit des ressources libres existantes afin de construire son propre système : lemmatisation morphologique de Xérox [Gaussier et al., 1997], synthèse de la parole de ELAN[8], serveur de base de données documentaire de l'INRIA.

1.5.3 Famille de Galaxy Communicator

Galaxy Communicator[9] est une infrastructure distribuée dont les communications entre ses composants sont basées sur les messages transmis autour du central. Il est une extension/évaluation du système MIT Galaxy[10] et développé dans le cadre du programme « DARPA Communicator » afin de construire des systèmes de dialogue. Ses codes sources sont ouverts et gratuits. Galaxy Communicator est illustré par la figure 1.10.

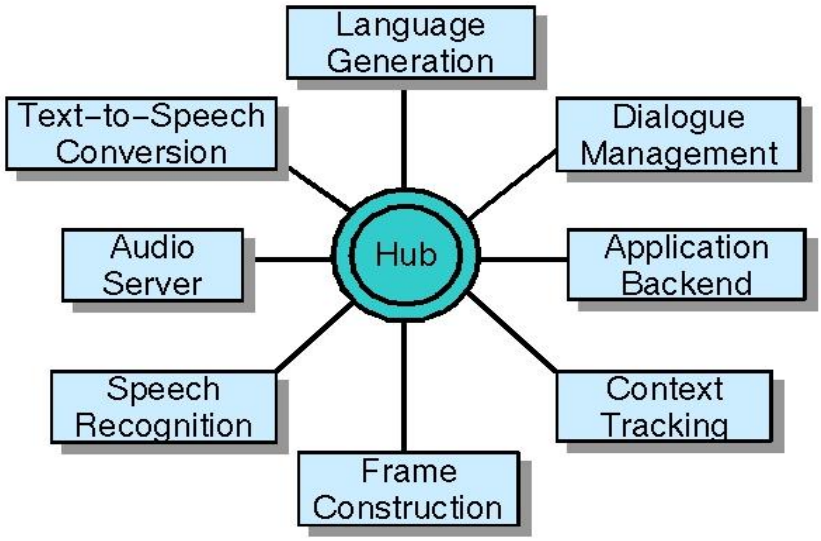


Figure 1.10 : Infrastructure de Galaxy Communicator

Dans cette architecture, les données envoyées par les composants sont formées dans un « *frame* ». Chaque composant est considéré comme un serveur et peut être distribué. Le HUB a pour rôle de coordonner tous les autres serveurs dans le système : les composants ne communiquent que via le HUB. Chaque composant peut s'installer matériellement indépendamment des autres.

Les rôles principaux du HUB sont : le routage de messages entre les modules, le contrôle de flux de données et de système. De plus, un point fort du HUB est de fournir la possibilité de programmer en langage script, afin de gérer les autres modules. Cela apporte véritablement de la flexibilité au système.

[8] http://www.elan.fr
[9] http://communicator.sourceforge.net/
[10] http://www.sls.lcs.mit.edu/GALAXY.html

Actuellement, avec le parrainage de DARPA, plusieurs systèmes de dialogue sont développés en s'appuyant sur cette infrastructure. Nous citons ici quelques systèmes reconnus comme :

- Le CU Communicator[11] est un système de dialogue pour fournir des services comme l'information sur les vols, les hôtels (horaire, prix, disponibilité...) par le téléphone en langage naturel ou par l'Internet. Il est élaboré par « *The Center for Spoken Language Research, University of Colorado* » et également gratuit,

- Endinburg Communicator[12] provenant du groupe *Edinburgh Language Technology Group*. Il a pour but de traiter un itinéraire de voyage complexe comme la réservation de vols, d'hôtels, de voitures en se basant sur l'information obtenue à partir de l'Internet. L'utilisateur peut communiquer avec le système via le téléphone ou l'Internet,

- Mercury provient du laboratoire SLS-MIT[13] qui permet à l'utilisateur de réserver et évaluer des itinéraires de voyage complexe vers plus de 200 villes aux Etats-Unis et autour du monde,

- CMU Communicator est un système de dialogue provenant du laboratoire de la parole de l'université de Carnegie Mellon. Il a pour objectif de permettre à l'utilisateur de planifier un voyage en avion, de louer une voiture, de réserver une chambre en utilisant un serveur vocal et le réseau téléphonique [Rudnicky et al., 1999]. Dans ce système, on propose une architecture de gestion de dialogue basée sur un agenda qui gère les topiques concernant l'accomplissement de tâche [Xu et al., 2000].

1.5.4 COLLAGEN

COLLAGEN (COLLaborative AGENt) est un middleware en Java qui fournit un moyen de développer des agents collaboratifs qui permet à l'utilisateur de résoudre des problèmes complexes dans un domaine peu familier. Visiblement, un agent collaboratif a pour but d'expliquer le problème, de donner des suggestions, de corriger des fautes…

Le modèle de dialogue dans COLLAGEN est rattaché au modèle de la tâche qui est totalement utilisé comme connaissances globales. Inspiré fortement du modèle orienté plan (cf. section 2.2.1), COLLAGEN n'est considéré que comme un interprète du modèle de la tâche, qui spécifie toutes les séquences des actions élémentaires, qui permettrent d'atteindre un but posé précis.

[11] http://communicator.colorado.edu/
[12] http://www.ltg.ed.ac.uk/Communicator/
[13] http://www.sls.lcs.mit.edu/sls/applications/mercury.shtml

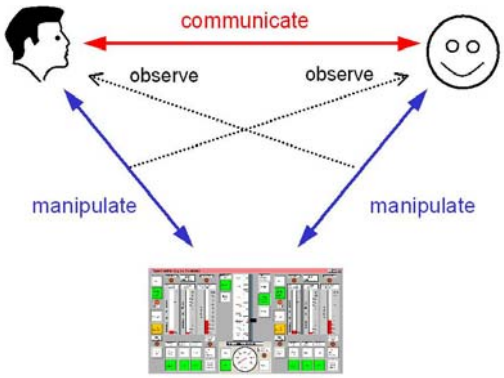


Figure 1.11 : Interaction dans COLLAGEN

Des applications élaborées fondées sur COLLAGEN peuvent être listées comme « Power plant operator training » [Rickel et al., 2001], remplir des pages Web multimodal, robot à guide [Sidner et al., 2003]…

1.5.5 JUPITER

JUPITER est un système conversationnel dédié au service de consultation de la météo par téléphone. Ses connaissances couvrent plus de 500 villes dans le monde entier en collectant les ressources fondées sur 4 sites Web différents [Zue, 1997].

Depuis mai 1997, Jupiter a accepté plus de 100.000 appels et par statistique de son site web, le taux de satisfaction par les utilisateurs normaux a déjà atteint 90% avec son vocabulaire ayant seulement 2000 mots. Ses capacités représentent éventuellement des réponses adéquates pour des questions concernant la météo générale, la température, la vitesse de vent, l'humidité, l'heure du lever du soleil, la population de certaines régions…

L'enchaînement d'un dialogue avec Jupiter est illustré comme suit : les signaux sonores prononcés par l'utilisateur au téléphone sont analysés par les modules de traitement du signal, de la parole du groupe SLS[14] (reconnaissance de la parole, compréhension…). Le sens de chaque énoncé de l'utilisateur est décodé et Jupiter va chercher la réponse adéquate à cet énoncé. Les données concernant la météo sont collectées dans différentes sources dans le monde entier par des sites Web de météo. Néanmoins, ces données brutes sont analysées et cette tâche est assurée par le module TINA [Seneff, 1992]. Une fois obtenue la réponse pour l'utilisateur, Jupiter génère les signaux sonores de la réponse par le module de synthèse de la parole.

[14] SLS : Spoken Language Systems Group, http://www.sls.csail.mit.edu/sls

1.6 Conclusion

Nous avons présenté dans ce chapitre une vue globale, générale du domaine du dialogue oral homme-machine. Partant de l'interaction homme-machine, la parole a été envisagée dans ce cadre. Nous avons par la suite abordé le dialogue oral homme-machine, ses propriétés. L'apparition de la norme VoiceXML apporte vraiment des avantages pour construire des applications vocales simples mais nos analyses montrent qu'il ne répond pas encore aux exigences de l'utilisateur en raison de ses caractéristiques intrinsèques, de son objectif… A partir de cela, nous avons donné la définition d'un système de dialogue oral homme-machine afin d'avoir une vue sur une application vocale générale. Ensuite, nous avons décrit en détail une architecture générale dédiée au système de dialogue avec sept modules principaux. La séparation de fonctionnalités de chaque module permet effectivement de le mettre en œuvre de manière distribuée un système de dialogue.

En faisant un état de l'art de certains systèmes de dialogue existants, les limites de la plupart d'entre eux sont apparues. La dépendance de la tâche, la restriction de l'extensibilité et ainsi que la capacité de négociation résident encore dans ces systèmes. Ces remarques sont donc un des points nécessaires et indispensables à lever. Cette thèse a évidemment l'objectif de contribuer à résoudre ces problèmes.

Nous présenterons dans le chapitre suivant un modèle de dialogue, qui permettra de diminuer davantage la dépendance de la tâche et d'introduire la possibilité de négociation dans le système de dialogue.

Chapitre 2
MODÈLE STRATÉGIQUE DE DIALOGUE

Nous allons voir tout d'abord dans ce chapitre la modélisation en général du dialogue oral homme-machine, en présentant des modèles de dialogue existants, dans l'intention de trouver un modèle adéquat pour la gestion générique du dialogue. Puis, nous décrivons le modèle stratégique qui offre des avantages par rapport à la généricité d'un système de dialogue. Et finalement, les remarques seront présentées à la conclusion.

2.1 Introduction

Le dialogue oral homme-machine pose effectivement l'enjeu important d'intégrer la machine au processus de communication humaine. Dans ce sens là, la modélisation du dialogue oral homme-machine est essentiellement une étape nécessaire, indispensable et décisive pour l'efficacité ainsi que la robustesse d'un tel système.

Partant du fait d'envisager le DHM sous l'angle d'un jeu, nous allons voir dans ce chapitre la modélisation de dialogue en se fondant sur trois activités principales : acte de dialogue, but et stratégie de dialogue. Avant de présenter des différents modèles de dialogue, afin d'en mesurer les points forts ainsi que les limites, nous présentons maintenant des aspects théoriques concernant le domaine de la modélisation de dialogue.

2.1.1 Jeux de dialogue : une activité conjointe

La parole est normalement utilisée afin d'échanger des informations. L'énoncé d'un locuteur est effectivement interprété en des actions sur le monde par l'autre ou bien par lui-même. [Clark, 1996] considère que « *la pratique du langage est une forme d'action conjointe* ». Partant de la notion de type d'activité, [Levinson, 1979] caractérise les activités conjointes par huit points suivants :

i- une activité conjointe implique au moins deux participants,

ii- les participants jouent un rôle au sein de cette activité,

iii- les participants cherchent à réaliser certains buts collectifs,

iv- les participants cherchent à effectuer certains buts privés,

v- une activité conjointe se compose d'actions ou d'activité conjointe,

vi- les participants sont susceptibles d'utiliser des procédures spécifiques pour atteindre leurs buts,

vii- les participants s'accordent sur le début et la fin de l'activité conjointe,

viii- une activité conjointe peut être simultanée ou séquencée.

Nous pouvons constater que le dialogue oral homme-machine s'impose également ces caractéristiques en considérant la machine comme un participant particulier au sein des interlocuteurs. Naturellement, l'activité conjointe dans un dialogue homme-machine nécessite vraisemblablement d'être coordonnée. De plus, dans un dialogue engagé entre plusieurs utilisateurs et la machine (cf. chapitre 4), cette coordination doit être envisagée par tous les participants pour que chacun puisse prendre en compte les actions réalisées par les autres.

La théorie des jeux a inspiré le domaine du dialogue homme-machine et attiré l'attention des chercheurs dans ce domaine. Leur objectif principal est de tenter de modéliser le dialogue en appliquant cette théorie. En effet, les philosophes s'intéressent déjà très tôt à la conversation sociale régulée. [Wittgenstein, 1957] a le premier considéré qu'une conversation sociale est un jeu de langage. Les travaux de [Levin et Moore, 1980] ont utilisé explicitement cette notion, afin de proposer la structure d'un jeu de dialogue, qui se compose de paramètres pour les participants et le sujet de l'interaction, de spécification de ces paramètres et de composants concernant les sous-buts des participants. L'approche de [Lewin, 2000] considère un jeu composé par une paire du type de jeux et du contenu propositionnel et par les règles des jeux. Les règles des jeux sont formalisées par des réseaux de transition récursifs RTN (Recursive Transition Network).

Partant de l'activité conjointe d'un dialogue et des notions de jeux, nous considérons un dialogue homme-machine comme un jeu engageant au moins deux interlocuteurs : un ou plusieurs utilisateurs et le système représenté par la machine. Dans ce jeu, les tours de parole représentent des coups de jeu. Ces coups sont plus ou moins pertinents (c'est-à-dire qu'ils atteignent plus ou moins leur but). On avance dans le jeu en en suivant les règles et en tentant de maximiser ses gains ou de réduire ses pertes (par exemple le gain à la sortie du jeu sera d'avoir obtenu un renseignement, ou d'avoir résisté aux arguments de son interlocuteur, etc.). On distingue plusieurs catégories de modèles : les modèles logiques et les modèles stratégiques. Dans les modèles logiques l'accent est mis sur les gains (convaincre son interlocuteur par exemple) sans s'intéresser outre mesure à la manière dont est obtenu le résultat, tandis que dans les modèles stratégiques c'est l'inverse, il s'agit d'atteindre un résultat de manière optimale.(cf. section 2.3)

2.1.2 Modèle de dialogue et système de DHM

Le modèle de dialogue est normalement un modèle qui représente une description des diverses situations concernant l'application considérée en précisant les liaisons entre ses diverses phases. Le modèle de dialogue est souvent pris comme un mot-valise, il sous-tend tout autant l'approche théorique que les modules qui le composent : gestion des connaissances ou de la tâche, gestion des tours de parole, contrôle de l'avancée du dialogue, etc [Caelen, 1997]. Les modèles de dialogue généralement utilisés dans le cadre du DHM sont des modèles issus de la linguistique descriptive. La fonction générale de ces modèles est de décrire structurellement et fonctionnellement les conversations entre l'utilisateur et le système sans compter sur une quelconque compétence conversationnelle [Lehuen, 1997].

Selon l'aspect cognitif, nous considérons le contrôleur du dialogue comme le « cerveau» du système de dialogue avec le modèle de dialogue représentant toute l'infrastructure de ce cerveau. Intuitivement, il est considéré comme la tête d'un enfant qui vient de naître avec naturellement certaines connaissances figées au début. Pour un homme, ses connaissances dynamiques sont acquises au fil du temps via des communications, des apprentissages, des raisonnements, etc. Cela vient également au système de dialogue. Au cours des interactions, ses connaissances dynamiques seront enrichies en considérant des tâches, des contextes précis. L'enrichissement de ses connaissances se déroule évidemment avec le modèle de tâche dans le contrôleur de la tâche (cf. section 1.4.2)

En ce qui concerne l'objectif général d'un modèle de dialogue, [Bilange, 1992] précise des exigences portant sur un tel modèle à satisfaire comme :

- l'indépendance de la langue et de la tâche pour des raisons évidentes de généricité.
- l'identification des différentes entités du discours : leurs types, leurs rôles et les relations entretenues entre eux.

La validité d'un modèle de dialogue n'est pas jugée que par l'aspect théorique. *« Du fait qu'étudier et modéliser le dialogue homme-machine est à notre avis une activité encore empirique, il n'y a pas de moyen a priori de garantir la validité d'un modèle de dialogue »* [Bilange, 1992]. La méthode adéquate pour valider un modèle est donc de se fonder non seulement sur l'expérimentation, mais également sur l'évaluation en cours d'exploitation du système de dialogue, qui est construit en utilisant ce modèle.

Nous présentons maintenant quelques éléments importants concernant la modélisation du dialogue.

2.1.3 Théories de la conversation

2.1.3.1 Des maximes de la conversation à la théorie de la pertinence

La première théorie parmi les plus fécondes retenue pour le dialogue homme-machine il faut citer les maximes de conversation de Grice [Grice, 1975] qui permettent d'orienter une conversation. Ces maximes sont subsumées par le principe général de coopération : *« Make your conversational **contribution** such as is required, at the stage of the talk exchange in which you are engaged »*[15][Grice, 1975]. Partant de ce principe, Grice définit quatre maximes suivantes :

- **Maximes de quantité** d'information à échanger :
  - Que votre contribution contienne autant d'information qu'il est requis,
  - Que votre contribution ne contienne pas plus d'information qu'il n'est requis.
- **Maximes de qualité :** Que votre contribution soit véridique :
  - Ne rien dire que vous croyez faux,
  - N'affirmez pas ce pour quoi vous manquez de preuves.
- **Maxime de relation** : pertinence de la contribution :
  - L'information donnée doit être pertinente.
- **Maximes de manière** (*manner maxim*) : Soyez clair :
  - Evitez d'utiliser des expressions obscures,
  - Evitez d'être ambigu,
  - Etre bref (éviter toute prolixité inutile),
  - Donner les informations dans le bon ordre.

Sur la base de ces maximes, un locuteur accède au sens de l'énoncé par des inférences dites « *implicatures* ». Par ce terme, Grice veut distinguer certaines conclusions que l'on peut tirer des énoncés, par des implications logiques.

Le principe de coopération de Grice est controversé principalement par Sperber et Wilson [Sperber et Wilson, 1986], qui développent une conception cognitive de la communication sous le nom de « *Théorie de la pertinence* ». L'idée principale de cette théorie est :

« Un énoncé est d'autant plus pertinent qu'avec peu d'information, il amène l'auditeur à enrichir ou modifier davantage ses connaissances ou ses conceptions.

La pertinence d'un énoncé est en proportion directe du nombre de conséquences pragmatiques qu'il entraîne pour l'auditeur et en proportion inverse de la richesse d'information qu'il contient.

[15] « Que votre contribution conversationnelle corresponde à ce qui est exigé de vous, au stade atteint par celle-ci, par le but ou la direction acceptée de l'échange dans lequel vous êtes engagé »

On dira ainsi qu'une proposition p est d'autant plus pertinente par rapport à un ensemble M de propositions que l'union de p et M permet le calcul d'un grand nombre de conséquences nouvelles. »

La pertinence repose pratiquement sur deux facteurs principaux : l'effort cognitif produit chez le destinataire, et l'effort qu'il a fallu pour le produire. Le rapport de ces deux quantités est considéré comme le juste équilibre du coût de traitement chez les deux interlocuteurs.

Le fait d'étudier ces théories de la conversation cognitive nous apporte l'hypothèse du modèle inférentiel du dialogue homme-machine en trois thèses suivant [Sperber, 2000] :

(i) L'utilisateur produit un indice du sens voulu,

(ii) La machine, dans tous les cas, infère ce sens voulu à partir de l'indice fourni et du contexte,

(iii) Un énoncé linguistique est un indice complexe du sens voulu par l'utilisateur ainsi que la machine. Ce n'est pas un encodage de ce sens voulu.

Ces trois thèses sont retenues dans l'approche de modélisation du dialogue présenté dans la section 2.3.

2.1.3.2 Théorie des actes de langage

Durant les dernières années, la théorie des actes de langage a eu des influences importantes dans de nombreuses disciplines, non seulement en philosophie, linguistique et sémiotique mais aussi en logique, intelligence artificielle, psychologie cognitive. Dans le domaine de DHM, cette théorie se trouve en pratique dans une position importante.

L'insuffisance d'une sémantique générale en linguistique demandait aux travaux de la philosophie analytique de mettre l'accent sur la pragmatique linguistique, en analysant bien la nature du langage. Les travaux initiaux d'Austin [Austin, 1962] ont amené à conclure de manière générale que tout énoncé est d'abord et avant tout un ***acte de langage***. Selon lui, un acte de langage peut se distinguer en trois composants :

(i) l'***acte locutoire***, c'est l'acte de dire quelque chose, l'acte d'énonciation proprement dit,

(ii) l'***acte illocutoire***, qui est l'acte effectué en produisant l'énoncé. Les mêmes paroles peuvent être comprises différemment (par ex : affirmer, conseiller, menacer, promettre, etc.),

(iii) l'***acte perlocutoire*** qui est la conséquence sur l'interlocuteur de l'acte illocutoire. Il est caractérisé en termes d'effets réalisés par l'énonciation chez l'interlocuteur (par ex : effrayer quelqu'un en le menaçant)

Il est certain que l'acte illocutoire est l'acte essentiel de la parole, caractérisé par son aspect conventionnel, pour le domaine de DHM. Les travaux de [Searle, 1969] ont décrit une structure interne générale de l'acte illocutoire en deux composants : d'une part son contenu propositionnel, noté par ***p***, et d'autre part sa force illocutoire, noté par ***F***. L'acte illocutoire est ainsi représenté par

la fonction F(p), par exemple : l'énoncé *« est-ce que la salle Aquarium est disponible ? »* signifie une question et donc est représenté par

F(p) = question(Aquarium être disponible).

La taxonomie des actes illocutoires, proposée par Austin et complétée par [Searle, 1979] en donnant douze critères comme *but illocutoire, direction d'ajustement mots-monde, état psychologique du locuteur, contenu propositionnel…*, définit cinq grands types différents [Vanderveken, 1990]. [Caelen, 2002] propose une variante suivante :

Acte assertif : la composante illocutoire décrit un état de fait existant. Le locuteur dit comment sont les choses. Le but est de rendre le contenu propositionnel conforme au monde. Cet acte relève des croyances du locuteur et est noté par $\mathbf{F^S}$ (faire savoir). Par exemple es actes d'affirmer, constater, décrire, expliquer, critiquer, contester…

Acte directif : le but illocutoire est de mettre l'interlocuteur dans l'obligation de réaliser une action future. Le locuteur essaie de faire faire les choses et manifeste ses désirs ainsi que sa volonté. Cet acte est noté $\mathbf{F^F}$ (faire faire une action), ou $\mathbf{F^D}$ (faire devoir, ordonner) lorsque l'obligation est forte, ou bien $\mathbf{F^{FS}}$ (faire faire savoir) pour une demande d'information.

Acte promissif : le locuteur s'engage à réaliser une action future. Cet acte relève l'intention du locuteur et est noté $\mathbf{F^P}$ (faire pouvoir : promettre, s'engager…)

Acte expressif : le but illocutoire de l'acte expressif est d'exprimer un état psychologique qui lui est associé. La direction d'ajustement n'est pas de rendre le monde conforme aux mots ou vice versa. La proposition exprimée est présupposée. Cet acte est très peu présent en DHM et peut être négligé car il ne fait pas avance le jeu de dialogue.

Acte déclaratif : le but illocutoire de l'acte déclaratif est de rendre effectif son contenu. Les déclarations du locuteur provoquent des changements effectifs dans le monde et rendent simultanément deux directions d'ajustement entre le langage et le monde. Cet acte est noté par $\mathbf{F^A}$ (faire action).

L'évolution des recherches de la philosophie analytique divise la philosophie du langage en deux grands courants rivaux : le courant *logique* et celui du langage *ordinaire* [Minker et Bennacef, 2001]. Vanderveken, fondateur de la logique des actes de langage, a contribué à faire converger ces deux courants philosophiques en formalisant une logique illocutoire en vue de développer une sémantique formelle de la langue naturelle, susceptible de caractériser à la fois les conditions de succès et de vérité (véri-conditions), et illocutoire de la signification des énoncés [Vanderveken, 1988]. Il part de l'hypothèse de base que tout énoncé doit contenir un marqueur de force illocutoire. Normalement, le mode du verbe principal, le type syntaxique, l'ordre des mots, ainsi que l'intonation ou les signes de ponctuation…, sont les traits constitués habituellement des marqueurs de force illocutoire. De même, certains traits contextuels sont généralement pertinents au cours de la détermination de l'acte illocutoire exprimé par un énoncé dans un contexte concret [Minker et Bennacef, 2001]. Ces remarques sont très importantes en vue de l'idée d'obtenir un module de compréhension robuste présenté à la section 5.4.2.

La force illocutoire, dans cette logique, dépend de six facteurs : le but illocutoire, le mode d'accomplissement de ce but, des conditions sur le contenu propositionnel, des conditions préparatoires, des conditions de sincérité et un degré de puissance.

En communication sociale, certains énoncés doivent être des actes de langage indirects. Dans ces cas là, leur véritable force illocutoire n'est pas celle directement déduite de la forme linguistique de l'énoncé. Par exemple l'énoncé *« pouvez vous me réserver une salle ? »* ne signifie pas l'acte « faire faire savoir » mais l'interprétation, qui vient à l'esprit, est naturellement une demande de faire une chose précise, c'est-à-dire un acte « faire faire : *réserver une salle* ». Il est certain que la détermination d'un acte de langage indirect à partir de son énonciation demande forcément d'appliquer les inférences nécessaires.

2.2 Modèles de dialogue : Etat de l'art

Nous abordons dans cette section quelques approches différentes de modélisation du dialogue dans l'optique d'obtenir un panorama sur la méthodologie de conception du système de DHM. Nous distinguons ces approches en quatre catégories principales : modèles orientés plans intentionnels, modèles structurels, modèles conversationnels et modèles logiques.

2.2.1 Modèles orientés plans intentionnels

En sciences cognitives, la notion de « plan » est définie comme une représentation schématique et/ou hiérarchisée qui est susceptible de guider l'activité du sujet. L'élaboration et/ou la mise en œuvre de plans constituent l'activité de *planification* [Hoc, 1987].

Les modèles dits orientés plans intentionnels partent du modèle qui spécifie une séquence d'opérations dans un plan permettant d'atteindre des buts depuis un état initial. Le plan, dans ces modèles, est alors organisé par des actions qui puissent conduire à un but souhaité une fois qu'on le suit correctement. Les buts sont généralement décomposés en arbres hiérarchiques de sous-buts, avec les feuilles (nœuds terminaux) étant les actions à réaliser.

L'idée principale d'un modèle orienté plan intentionnel part du fait que le dialogue est considéré comme une activité coopérative et intentionnelle. Cette approche se situe généralement dans le cadre de la théorie de la planification. La gestion de dialogue est donc effectuée avec l'aide de plans, de la structure et de la fonction des actes de dialogue, et de l'analyse des intentions des interlocuteurs. L'idée principale de cette approche est : si l'on suppose que le locuteur a des buts, et qu'il planifie sa tâche et le dialogue en produisant des actes de langage, alors la machine doit reconnaître le plan à travers les actes de langage de manière à déduire les buts du locuteur.

2.2.1.1 Modèle de Allen et Perrault

Le modèle de dialogue, proposé par Allen et Perrault dans [Allen et Perrault, 1980], est construit en se fondant sur les quatre éléments principaux : la notion de croyance représentée par la logique modale, les schémas d'action structurés par les actes de langage en les considérant comme les actions, le processus de reconnaissance et de construction de plans de l'interlocuteur.

Selon Allen et Perrault, un schéma d'action est une règle qui regroupe les élements : un nom avec un ensemble de paramètres ; un ensemble de formules comportant chacune des préconditions à exécuter ; des effets après l'exécution ; et un corps spécifié l'action sous forme de sous-buts ou sous-actions (voir l'exemple d'un schéma d'action illustré à la figure 2-1)

Name	***Inform***(**S**peaker, **H**earer, *Proposition*)
Preconditions	Know(**S**peaker, *Proposition*)
Body	MutuallyBelieve(**S**peaker, **H**earer, Want(**S**peaker, Know(**H**earer, *Proposition*)))
Effect	Know(**H**earer, *Proposition*), Know(**H**earer, Know(**S**peaker, *Proposition*))

Figure 2.1 : Schéma d'action pour l'acte de langage « *Inform* »

Le plan est ensuite défini comme une séquence linéaire d'occurrences d'actions menant d'un état initial du monde à un état final but. La reconnaissance de plans consiste donc à observer les actions afin de remonter au but qui est posé par l'interlocuteur. Lorsque le système a inféré tous les buts, les intentions supposées de son interlocuteur, il peut donc construire un plan qui permet de réaliser ces buts.

L'avantage du modèle d'Allen et Perrault est de modéliser le dialogue comme une activité en-soi et de la distinguer de l'activité déployée pour la tâche. Par contre, ce modèle reste limité à l'analyse d'un seul but représenté dans un énoncé et pose donc des problèmes quand il faut traiter des énoncés complexes ayant plusieurs buts à la fois. Les difficultés de reconnaissance de buts, des intentions implicites sont également des points faibles d'un tel modèle [Caelen, 2002].

2.2.1.2 Modèle de Litman

Le modèle de Litman [Litman et Allen, 1990] part des travaux d'Allen et Perrault en distinguant deux catégories de plans : les plans du domaine qui modélisent les tâches sous-jacentes au dialogue de l'application, tandis que les plans du discours sont indépendants de la tâche et permettent de manipuler de différentes manières la structure d'autres plans. Afin d'éviter la limite de mise en relation des actes de langage et des plans du domaine par des phénomènes locaux, ce modèle vise en plus à rendre compte de phénomènes globaux tels que la gestion de sous-dialogues de clarification en utilisant des plans du discours. Le principe de la *métaplanification*, introduit par [Wilensky,1983], est donc utilisé pour spécifier des plans du discours. Ces plans (*métaplans*) ne manipulent pas des données normales comme le temps, le lieu… mais des données constituées de

plans du domaine ou de suivi du dialogue. La structure du dialogue est modélisée par une pile de plans du domaine et de métaplans. [Carberry, 1990] précise que ce modèle, bien que restrictif, semble assez conforme à la majorité des comportements dialogaux observés.

Le modèle de Litman est ensuite repris dans [Nerzic, 1993], en visant à traiter certaines situations d'erreurs concernant la qualité des intentions, que le système attribue à son locuteur. Il se concentre sur la reconnaissance de plans invalides, incorrects qui sont les inconvénients essentiels du modèle de Litman.

2.2.1.3 Modèles mentaux

Les modèles présentés ci-dessus ne considèrent que la planification sous l'angle de l'organisation de la tâche et du discours. Les travaux de Pollack [Pollack, 1990] initient l'approche mentaliste en montrant qu'il est nécessaire de modéliser des attitudes mentales complexes par des plans d'organisations des intentions. Selon Pollack, les plans sont construits comme des collections mentales (des croyances et intentions) de chaque locuteur (plan individuel), qui sont les composants principaux raisonnés par les interlocuteurs dans un dialogue. Il définit le plan individuel « effectuer une action α » constituante par quatre attitudes mentales :

(i) croyance que l'exécution de certaine action β_i peut entraîner l'exécution de α, la β_i a constitué « une recette pour α »,

(ii) croyance que le locuteur peut exécuter chaque β_i,

(iii) intentions de faire chaque β_i,

(iv) intentions de faire α en faisant β_i,

Dans la même optique, [Grosz et Sidner, 1990] ont étendu la notion de « plan individuel » de Pollack en proposant la notion « ***plan partagé*** » pour modéliser le dialogue entre deux locuteurs dans la situation de collaboration. Un plan partagé est également une collection d'attitudes mentales comme dans l'approche de Pollack, mais qui sont modifiées afin d'incorporer des actions concernant plusieurs locuteurs (*multi-agent actions*) et des aspects mentaux requis pour deux locuteurs afin de coordonner leurs activités. [Grosz et Kraus , 1996] ont révisé la notion de « plan partagé » en tentant de corriger des critiques, tout particulièrement le fait que *les locuteurs ne sont pas engagés en tant que groupe à propos du but commun* [Hoobs, 1990]. Les principes d'amélioration consistent à permettre les actions non-individuelles à tous les niveaux de décomposition, et surtout à engager formellement les partenaires sur la réalisation des actions réalisées par les autres. Ses travaux visent également à reformuler la notion de plan individuel ainsi que celle de plan partiel proposé par [Lochbaum, 1994] pour résoudre le problème où la recette à réaliser d'une action dans le plan individuel n'a pas toutes les connaissances nécessaires.

2.2.2 Modèles structurels

L'approche fondée sur la structuration du discours est également utilisée pour modéliser le dialogue homme-machine. Les modèles structurels visent à décrire la structure des dialogues à partir de marqueurs linguistiques et d'indices pragmatiques [Lehuen, 1997]. La grammaire formelle est éventuellement le fondement théorique dans la plupart des travaux qui s'appuient sur une modélisation structurelle du dialogue.

2.2.2.1 Modèle Genevois

Le modèle Genevois [Roulet, 1985] est considéré comme la source et la référence des modèles structurels. Ainsi, plusieurs systèmes de dialogue ont utilisé des modèles de dialogue adaptés ou inspirés du modèle Genevois, par exemple le système SUNDIAL pour la réservation de billet d'avion [Bilange, 1991], STUDIA pour l'évaluation interactive des connaissances dans le domaine statistique [Chevallier, 1992], etc. Les divers constituants du discours et de la conversation sont spécifiés dans ce modèle avec des liens entre eux déterminés essentiellement par l'analyse *hiérarchique* et *fonctionnelle* de la conversation.

- ***Catégories discursives*** : E (échange), I (intervention) et A (acte de dialogue).
- ***Fonctions*** : S (subordonné) et D (directeur) attaché aux catégories discursives.
- xS et xD (où x est une des trois catégories discursives) signifient respectivement constituant x subordonné et directeur. Une expression du type {A|I}D signifie soit un acte directeur, soit une intervention directrice. CD et CS se lisent respectivement constituant directeur et constituant subordonné.
- ***Règles syntagmatiques*** : $E \rightarrow I\{I\}^+$
  $I \rightarrow CD|CSCD|CDCS$
  $CD \rightarrow \{A|I\}D$
  $CS \rightarrow \{E|I|A\}S$

Figure 2.2 : Grammaire générative d'un discours selon Roulet et Mœschler [Mœschler, 1989]

Le modèle Genevois décrit le dialogue selon le principe d'existence de constituants directeurs (CD) et constituants subordonnés (CS) en trois niveaux : échange, interventions et actes de langage. Son énonciation simplifiée est précisée sous forme de règles de réécriture dans une grammaire générative [Mœschler, 1989] et illustrée par la figure 2.2 ci-dessus.

Des règles structurelles sont énoncées d'abord dans ce modèle afin rendre compte de l'enchaînement des constituants de la conversation. Etant donné des fonctions attachées aux constituants, le modèle rend compte ensuite des aspects fonctionnels des énoncés du dialogue. Ici, seuls les actes de langage et les interventions ont les fonctions illocutoires. Les interventions sont manifestées seulement par l'initiative ou l'action.

2.2.2.2 Modèle de Luzzati

Dans ses travaux, [Luzzati, 1995] a élaboré son modèle en se fondant sur des analyses du corpus SNCF de dialogue [Morel, 1989]. Ce modèle distingue trois niveaux de couples question/réponse (Q/R) : les Q/R principales, les Q/R secondaires et les Q/R incidentes.

Les Q/R principales constituent la représentation thématique d'un dialogue en spécifiant les espaces de référence. On change d'espace de référence lorsqu'on pose une nouvelle question principale, c'est-à-dire de tâche. La réponse principale est simplement la réponse à la question principale. Les Q/R secondaires servent à modifier, clarifier ou éventuellement ajouter des éléments à la question principale. Les Q/R incidentes sont des demandes soit de reformulation, soit de précision.

Ces trois niveaux de Q/R sont liés par l'utilisation des *procédures de chevauchement et de relance* pour assurer la continuité du dialogue. Le dialogue finalisé peut s'orienter sur deux axes : l'axe régissant qui fait progresser la tâche par les interventions (inspiré du modèle Genevois) et l'axe incident qui correspond aux couples Q/R incidentes. Ainsi, le critère de satisfaction d'une question principale est montré par la complétude interactionnelle sur l'axe régissant et le critère de réussite d'une illocution est dénoté par la complétude interactive sur l'axe incident [Lehuen, 1995].

Parmi les aspects négatifs de ce modèle, notons qu'il ne fait aucune distinction entre la structure du dialogue vue par la machine et la structure du dialogue vue par l'usager. La justification de l'incident n'est pas clairement précisée. De plus, en ce qui concerne l'acte de langage, ce modèle ne le distingue pas de l'intervention et il ne peut donc pas traiter des interventions ayant plusieurs actes de langage.

2.2.3 Modèles orientés négociation

En négligeant l'idée de modélisation de dialogue fondée sur la planification des intentions, sur la structuration des interventions, cette approche utilise les *enjeux conversationnels* [Baker, 1994] ou bien les jeux de dialogue [Maudet, 2001] comme l'idée principale pour modéliser le dialogue. Le dialogue est donc considéré comme une activité conjointe, qui engage au moins deux interlocuteurs.

2.2.3.1 Modèle de Baker

Le modèle de dialogue proposé dans [Baker, 1994] est présenté sous forme d'un ensemble de règles logiques avec des formules de type prédicat-argument. Ce modèle contient quatre composants principaux : un état initial, un état final, des objets de négociation, et des processus de négociation.

Dans une négociation, les locuteurs ont des attitudes sur des propositions. Les attitudes de base du modèle sont l'offre, l'acceptation, la ratification. La figure 2.3 suivante représente le schéma général du modèle de Baker. Les *negocia* sont représentés sous la forme de deux cercles pour

indiquer que les niveaux tâche et communication sont toujours présents simultanément lors d'une négociation. Les relations Ri entre les *negocia*, qui dépendent des croyances et des buts de chacun des agents, sont établies par les liens entre les actes de communication (par les fonctions de transformation dans le cas de l'argumentation). Une négociation sera représentée par un parcours à l'intérieur du triangle (cf. Figure 2.3).

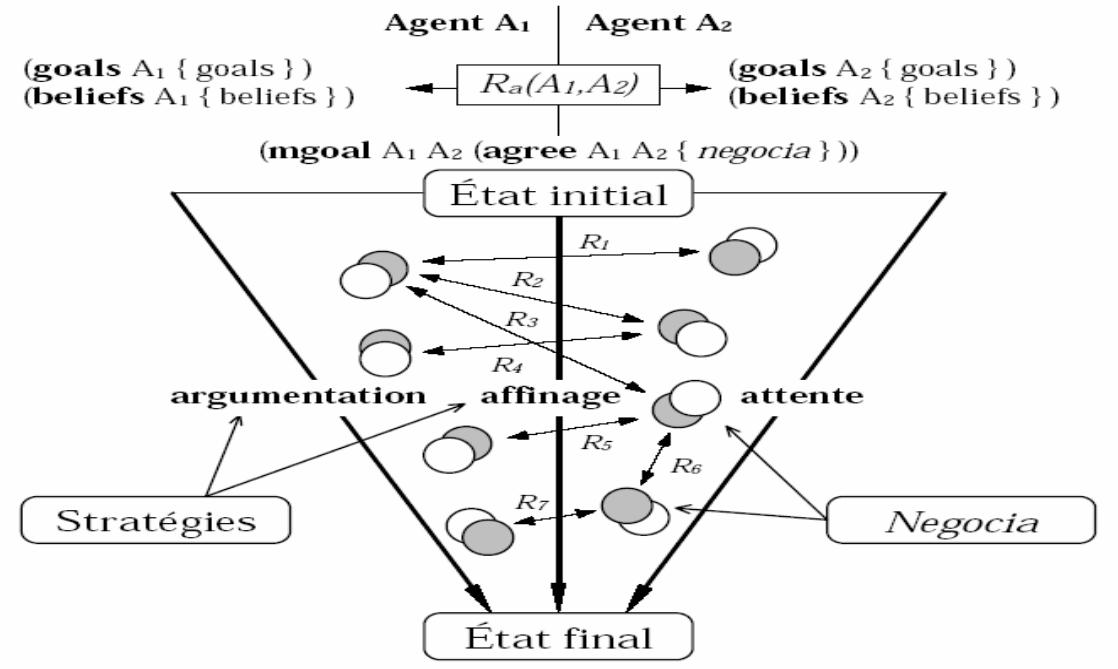


Figure 2.3 : Schéma général de négociation [Baker, 1994]

2.2.3.2 Modèle de Maudet

Les travaux de [Maudet, 2001] ont proposé une approche pour la modélisation de dialogue qui s'appuie sur les jeux de dialogue. Il distingue deux types de jeux : les jeux de dialogue qui sont spécifiques et introduits explicitement au cours du dialogue ; et les jeux de communication qui traduisent des conventions implicites valables dans toute conversation.

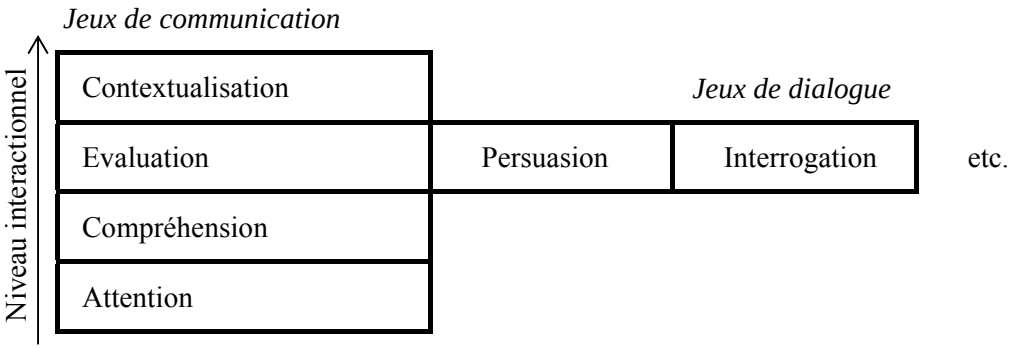


Figure 2.4 : Jeux de dialogue et jeux de communication [Maudet, 2001]

Les interlocuteurs communiquent par des coups, désignés par des actes de langages, qui leur permettent de s'engager de différentes façons au cours du dialogue. Le modèle schématique de l'interaction proposé par Maudet est illustré par la figure 2.5 suivante. Dans ce modèle, le tableau de conversation représente l'état du dialogue et répertorie les engagements des locuteurs (joueurs). Il distingue des engagements suivants : engagements propositionnels, engagements en action, engagements conjoints, engagements conditionnels. L'évolution de ce tableau est assurée par des opérations inspirées de [Singh, 1998].

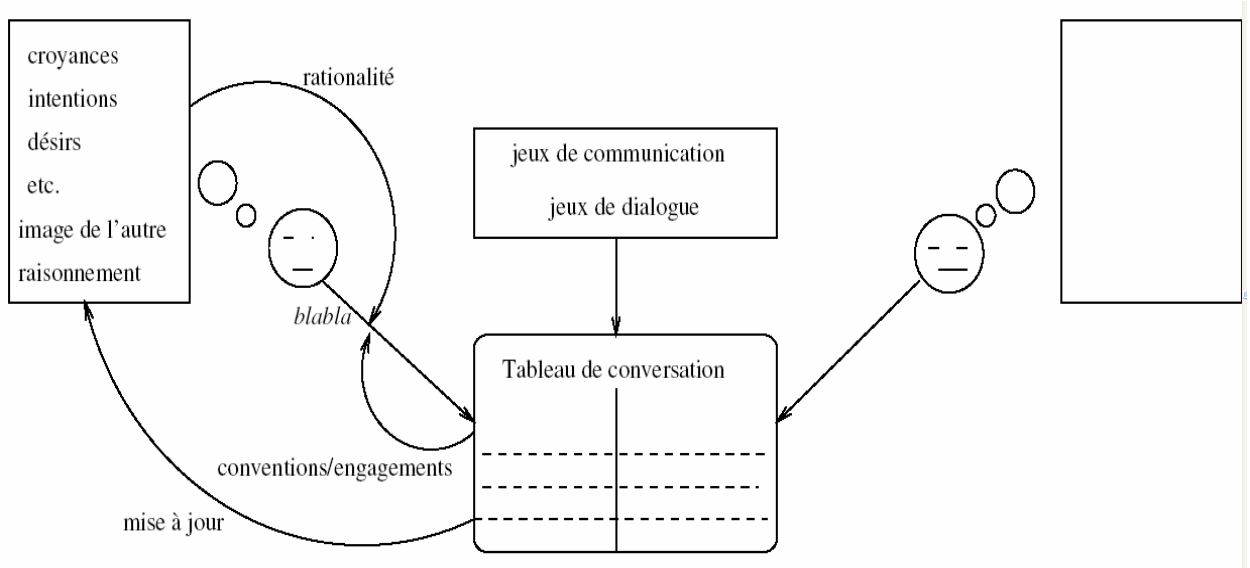


Figure 2.5 : Vision schématique du modèle de Maudet

2.3 Modèle stratégique de dialogue

Les différentes approches présentées ci-dessus ont des avantages ainsi que des inconvénients. L'inconvénient primordial des ces approches part du fait que les relations contextuelles de la tâche sont fortement présentes dans ces modèles. Il est évident que la généricité est violée de manière implicite ou bien explicite. Dans l'optique de trouver un modèle de dialogue qui réduit au maximum la dépendance du modèle de la tâche, nous présentons maintenant le modèle stratégique de dialogue proposé initialement par Caelen dans les années quatre vingt dix [Caelen, 1992] et qui évolue depuis. L'application de ce modèle dans le cadre du dialogue oral homme-machine sera explicitement présentée dans cette section.

2.3.1 Aspects psychologiques

L'idée principale de ce modèle réside dans la définition du dialogue oral homme-machine comme une activité conjointe qui engage un ou plusieurs utilisateurs et le système de dialogue. Cette activité est évidemment envisagée dans un cadre de la coopération des utilisateurs. En effet, comme en situation d'interaction humaine, l'utilisateur et le système en face-à-face construisent

leur dialogue de façon rationnelle en vue de satisfaire un certain but. Leur engagement est imposé également dans des conventions sociales respectées à la fois par l'utilisateur et par le système. L'activité conjointe et coopérative est donc au centre du modèle de dialogue.

Dans le modèle stratégique initial [Caelen, 1992], le dialogue est dirigé non seulement par les états mentaux (buts, intentions) de l'utilisateur, mais également par des attitudes qui sous-tendent les actes (choix, engagements).

2.3.2 Jeu de dialogue oral homme-machine

Dans ses travaux, le philosophe Wittgenstein [Wittgenstein, 1957] proposa initialement la notion « jeu de langage » (*language game*) et conçut la conversation comme une activité sociale régulée. Au cours des années, beaucoup d'autres philosophes, ainsi que des psychologues, des linguistiques…, ont apporté la notion de « jeu » au terrain de la modélisation de conversations. [Levin et Moore, 1980] considèrent un jeu de dialogue comme une structure avec des paramètres, les spécifications de paramètres et les composants. Plus tard, [Mann, 1988] définit un jeu comme une structure <IP, GR, CC> composée de : but de l'initiateur du jeu IP (illocutionary Point), celui du partenaire GR (Goal Responder) et l'ensemble des conditions conventionnelles du jeu CC (Conventional Conditions). Dans le cadre du projet TRINDI [Lewin, 2000], les jeux de dialogue sont également envisagés sous forme des paires <type de jeux ; contenu propositionnel>…

Selon nous, le jeu de dialogue dans le cadre du DHM est représenté sous une autre forme. Dans un dialogue, l'utilisateur et la machine (le système de dialogue) sont engagés dans un jeu (dit *jeu de dialogue*) dont les tours de parole représentent des coups d'échange entre eux. Chacun des deux, l'utilisateur et le système, tente essentiellement de maximiser ses gains et réduire ses pertes en suivant les règles du jeu. Les règles du jeu sont construites en s'appuyant sur la théorie des jeux [Myerson, 1991].

Le jeu de dialogue est manipulé par les trois éléments principaux : les actes de dialogue, les buts de dialogue et la stratégie de dialogue. Il est réglé par la gestion de stratégie et un mécanisme de contrôle du but. Des règles de reprise sont également considérées afin de gérer des sous-dialogues qui permettent notamment de re-négocier le jeu. Ces éléments seront détaillés dans la section ci-dessous.

2.3.3 Eléments principaux

2.3.3.1 Acte de dialogue

A partir de la notion d'« *acte de langage* » que nous avons abordée dans la section 2.1.3.2, nous définissions la notion d'« **acte de dialogue** » comme un acte de langage qui est enrichi par la force illocutoire avec la logique illocutoire de [Vanderveken, 1990]. Les forces illocutoires retenues dans le domaine du dialogue oral homme-machine sont [Caelen, 1992] :

Acte	Signification	Equivalant chez Searle
F^A	faire ou exécuter une action (en verbal ou non-verbal)	déclaratif
F^F	(faire-faire) demander de faire une action à l'allocutaire	directif
F^S	(faire-savoir) communiquer une information	assertif ou expressif
F^{FS}	(faire faire-savoir) demander une information	directif
F^P	(faire pouvoir) donner un choix, faire une invite	promissif avec délégation
F^D	(faire devoir) obliger sans donner d'alternative	directif

Tableau 2.1 : Actes de dialogue

Dans notre gestion de dialogue, un acte de dialogue est représenté par une force illocutoire **F** qui spécifie ce que le locuteur veut obtenir, et un contenu propositionnel **P** représentant le schéma pragmatique de l'énoncé. Le schéma pragmatique est construit en se fondant sur le schéma sémantique sortant du module de compréhension (cf. section 5.4.2.1) en le mettant dans le contexte du système de dialogue. Il est organisé par des concepts dont chacun doit avoir soit une valeur donnée par l'utilisateur, soit une valeur indéfinie à remplir par le contrôleur de la tâche. Le schéma pragmatique d'un acte de dialogue peut comporter plusieurs concepts reliés par des opérateurs logiques avec l'exigence que tous les problèmes concernant des anaphores (co-référence), des ellipses, des présuppositions, des implicatures… soient résolus.

La construction de l'acte de dialogue doit être fondée sur le sens sémantique, ainsi que pragmatique et peut être illustrée par la figure 2.6 suivante :

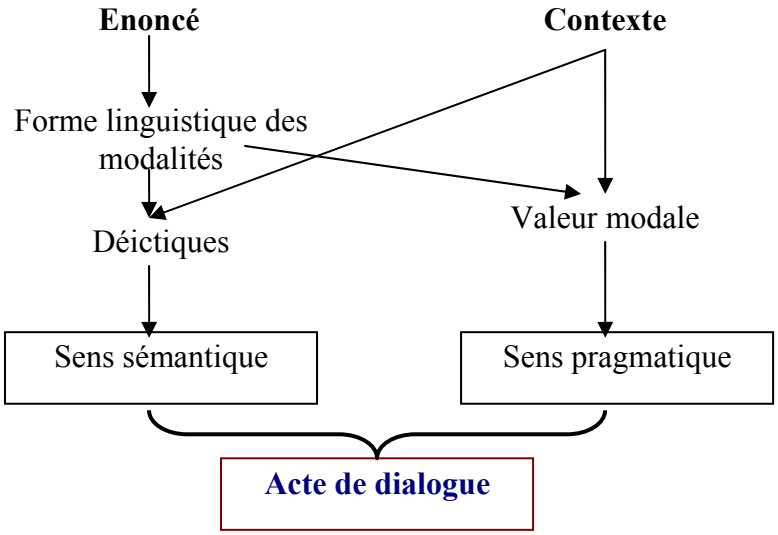


Figure 2.6 : Constitution de l'acte de dialogue

La modalité énonciative est aussi un élément qui a des influences sur la gestion du dialogue (cf. section 2.3.3.1). Au niveau de la grammaire classique, il y a les six modalités suivantes :

déclarative (celle qui est au mode indicatif), *interrogative* (questions), *impérative* (ordres), *conditionnelle*, *exclamative* (expression des états mentaux) et *optative* (expression des souhaits). Cependant, dans le cadre du dialogue oral homme-machine, nous avons retenu les quatre premières et laissé les deux dernières à traiter dans l'avenir (en combinaison avec d'autres modalités)

Voici un exemple dans lequel l'énoncé *« ici jean caelen je voudrais réserver une salle »* peut être interprété comme les deux actes de dialogue suivants :

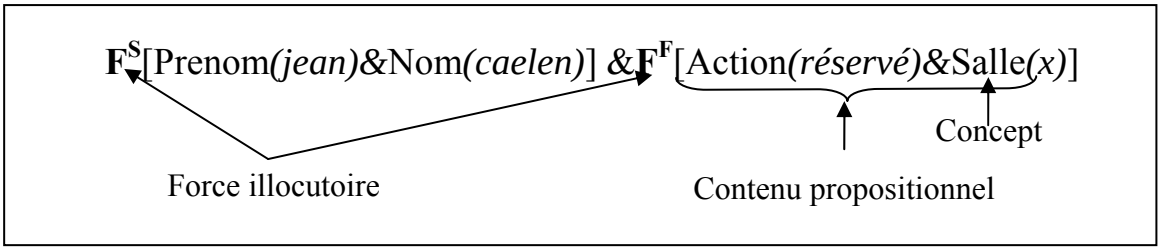


Figure 2.7 : Schéma pragmatique de l'acte de dialogue

2.3.3.2 But de dialogue

Un but est généralement un état du monde ou un état mental que l'on veut atteindre par exemple pour obtenir un renseignement, pour donner une information, etc. Dans le DHM, nous supposons l'utilisateur et le système comme étant deux locuteurs qui communiquent, chacun visant au départ un certain but dans l'arrière-plan. Un dialogue se présente comme une suite d'échanges, les échanges visant à résoudre des sous-buts ou des préalables.

Un échange est défini comme une suite d'actes de dialogue. Le début d'un échange est marqué par l'apparition d'un nouveau sous-but, ce sous-but se transforme éventuellement au cours de l'échange et devient un but final, atteint ou non, à la suite d'une série d'échanges. Le dialogue se termine par un succès ou par un échec. Le succès obéit à la double condition d'être un but *atteint* et un but *satisfait* [Vanderveken, 1997]. Remarquons enfin que le but final n'est pas toujours prévisible au départ [Caelen, 2002].

La distinction entre le but de la tâche (sur ce quoi porte l'activité) et le but du dialogue doit être bien claire. Ainsi, un acte de dialogue, lors d'un tour de parole, peut parfaitement réussir sans que le but avance, par exemple :

```
Utilisateur : allô, je voudrais réserver une salle pour lundi
         prochain
Système : désolé, toutes les salles sont déjà prises…
Utilisateur : bon… merci, au revoir
```

Dans ce cas il est clair que le dialogue s'est parfaitement déroulé, que tous les actes de dialogue ont réussi, que le dialogue a été efficace, que l'utilisateur a eu un renseignement (la tâche a été effectuée), mais malgré tout, nous constatons qu'il n'a pas pu réserver une salle, ce qui était pourtant son désir au départ et la motivation de ce dialogue. C'est cela que nous appelons le *but du*

dialogue et que nous cherchons à atteindre par tous les moyens, par exemple en proposant d'autres alternatives.

```
Utilisateur : allô, je voudrais réserver une salle pour
              lundi prochain
Système : désolé, toutes les salles sont déjà prises… mais
          je vais me renseigner pour savoir si une
          permutation est possible. Je vous rappelle.
Utilisateur : bon… merci, au revoir
```

Au début d'un dialogue, l'utilisateur et le système partent initialement avec leurs propres buts et puis cherchent à se rapprocher afin de réaliser le but commun en assurant la convergence du dialogue. Ici, la notion de « convergence du dialogue » est reprise comme illustration dans le modèle projectif de Vernant [Vernant, 2003] :

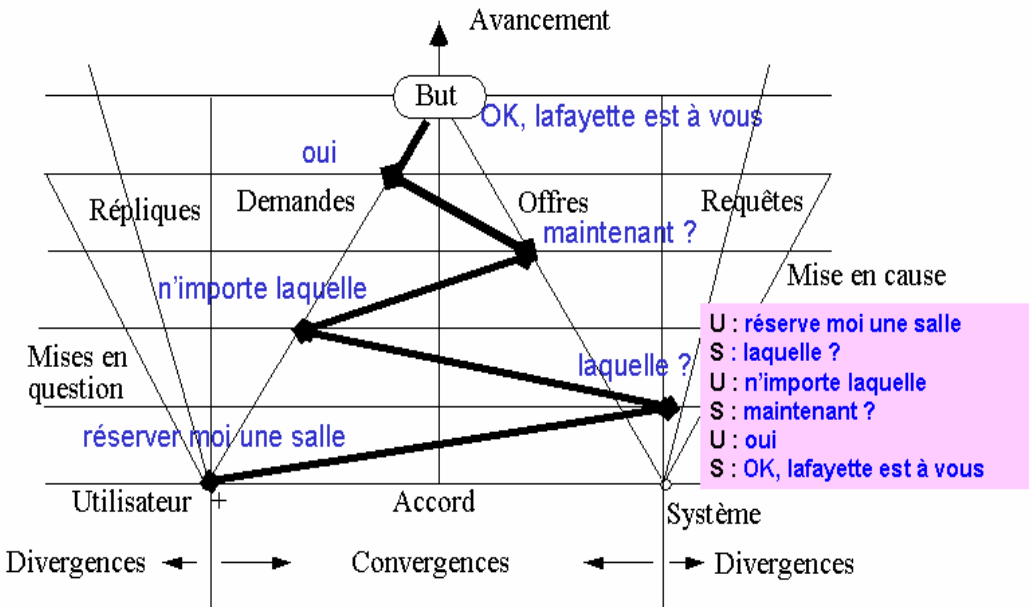


Figure 2.8 : Modèle projectif appliqué à l'avancement du but de dialogue

Pour modéliser un but, qu'il soit relatif à la tâche ou au dialogue, nous le définissons par un prédicat, par exemple : b_U = (∃x,∃y) : salle(x) ∧ jour(y) ∧ réservé(x,y,U) qui signifie que le but à atteindre pour l'utilisateur U est de réserver « *une salle x pour le jour y* ». Le but à satisfaire pour le système est donc de rendre b_U vrai, c'est-à-dire dans cet exemple, affecter des valeurs à x et y tel que *réservé*(x,y,U) devienne un fait vérifié.

Pour représenter l'état d'un but, on note [Caelen, 1997] :

Symbole	Etat	Description
?b	nouveau but	Ce but vient d'être exprimé par l'utilisateur ou la machine.
†b	but atteint	L'état de la situation rend le prédicat b vrai.
‡b	but satisfait	L'utilisateur manifeste son accord sur †b, cet accord pouvant être explicite ou implicite.
-b	but mis en attente	L'utilisateur ou la machine résout temporairement un autre problème.
b'	but réparé	A la suite d'une incompréhension ou d'un malentendu le but est modifié, on ne revient pas sur le but précédent.
b'	but déplacé	A la suite d'une négociation ou d'un ajustement le but est modifié, mais on peut revenir sur le but précédent.
sb	sous-but	Le problème est décomposé en sous-problèmes nécessaires à la résolution du problème principal.
@b	but abandonné	A la suite d'un échec ou d'un souhait d'abandon de l'utilisateur.

Tableau 2.2 : Etats d'un but de dialogue

Dans notre modèle de gestion de dialogue, un but est déterminé par l'abstraction de l'acte de dialogue avec l'aide de l'agenda de dialogue (qui est spécifié dans le modèle de tâche par les buts élémentaires et géré par le module de gestion de tâche) (cf. section 3.2.2). Par exemple avec l'énoncé « *Je voudrais réserver une salle* » et l'agenda de réservation de salle comme Réservé[Salle:Taille:Matériel], nous obtenons un acte de dialogue comme F^F[Action(réservé)&Salle(x)], et un nouveau but de dialogue comme ?*Réserver*.

Une fois que le contrôleur du dialogue a formé le but dialogique, il l'envoie au contrôleur de la tâche pour déterminer si ce but est soit atteint (†b), soit impossible à atteindre, soit en manque d'information (les états concernant la tâche). Le contrôleur du dialogue doit résoudre lui-même si le but de dialogue actuel est satisfait (‡b), mis en attente (-b), ou bien abandonné (@b).

2.3.3.3 Stratégie de dialogue

La stratégie de dialogue δ est la manière de gérer les tours de parole entre l'utilisateur et le système pour conduire efficacement le but de dialogue de l'utilisateur. La stratégie vise à choisir la meilleure direction d'ajustement du but à un moment donné. Nous distinguons la stratégie de dialogue selon les deux catégories différentes suivantes [Caelen, 1997] :

- **Stratégies non-inférentielles :** Ces stratégies sont dites non-inférentielles dans la mesure où le système ne cherche pas trouver un but conjoint avec son utilisateur et n'a donc pas à inférer nécessairement son but.

  - ♦ *Stratégie directive* consiste à garder l'initiative au système pour conduire le dialogue : en maintenant le but de l'échange, en imposant son but,

  - ♦ *Stratégie réactive* est invoquée afin de déléguer l'initiative à l'utilisateur soit en lui faisant endosser son but, soit en adoptant son but,

♦ *Stratégie constructive* consiste à déplacer le but courant momentanément afin de provoquer un détour, par exemple pour faire remarquer un oubli, une erreur…etc.

- **Stratégies inférentielles** : Ces stratégies exigent de la part des deux partenaires, l'utilisateur et le système, une connaissance perceptive fine de leurs buts respectifs.

♦ *Stratégie de coopération* consiste à tenir compte du but de l'utilisateur en lui proposant une ou des solutions qui les amènent rapidement tous deux à atteindre leurs buts. Dans le DHM, cette stratégie est utilisée le plus souvent en visant au dialogue coopératif,

♦ *Stratégie de négociation* est utilisée dans une situation où les buts sont incompatibles et que tous les deux, l'utilisateur et le système, veulent minimiser les concessions. La négociation est exprimée par des séquences argumentatives (argumentation / réfutation) avec proposition d'une solution sous-optimale jusqu'à convergence ou constat d'échec.

L'enchaînement des actes de dialogue est effectivement géré par des règles dépendantes des stratégies qui portent sur les comportements du dialogue, afin de calculer la stratégie pertinente dans chaque tour de parole. Nous reviendrons sur ce problème dans la section 3.2.4.

2.3.4 Stratégie de dialogue vis-à-vis des comportements

Une stratégie δ est calculée à l'aide de règles qui tiennent compte de la complétude d'un acte de dialogue, de la possibilité d'atteindre un but, des conditions de réussite de l'acte, de l'état du dialogue, des attentes de l'usager, de ses compétences, de la stratégie précédemment adoptée, etc. [Caelen, 2003]. Les comportements de l'utilisateur sont donc un élément influençant directement le calcul de la stratégie d'un tour de parole. De son côté, le système doit employer les comportements adéquats, afin d'atteindre rapidement le but de dialogue de l'utilisateur. Les comportements de l'utilisateur sont effectivement représentés dans l'acte de l'utilisateur.

2.3.5 Enchaînement de dialogue

L'enchaînement de dialogue oral homme-machine se compose de trois grandes étapes : Ouverture, Echanges et Clôture, illustrées par la figure 2.9 ci-dessous.

Le début d'un dialogue commence par l'étape d'ouverture avec des salutations prononcées par les deux locuteurs : système et utilisateur. Il y a éventuellement des boucles dans cette étape si l'utilisateur répète la salutation avec une intention d'amusement. Il faut donc avoir un mécanisme pour contrôler le nombre de répétitions dans cette étape pour éviter que l'ouverture s'éternise.

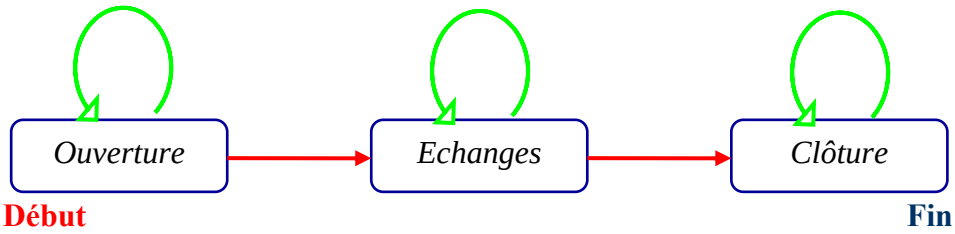


Figure 2.9 : Enchaînement d'un dialogue

Lorsque l'utilisateur manifeste un but, le dialogue arrive dans l'étape d'échanges. Tous les échanges dans un dialogue aboutissent à l'état que le but de l'utilisateur soit satisfait, soit mis en échec. Des sous dialogues peuvent également apparaître dans cette étape, afin d'enrichir le dialogue global ou de traiter des cas d'incidence.

Une fois que l'utilisateur a obtenu satisfaction, ou bien que le système n'a plus d'information à donner, le dialogue passe naturellement à l'étape de clôture. Bien évidemment, le but de départ de l'utilisateur peut être satisfait ou abandonné mais le dialogue se termine toujours par cette étape.

2.4 Conclusion

Nous avons donné dans ce chapitre une vue générale portant sur la modélisation du dialogue oral homme-machine. Partant des théories de base concernant la modélisation du dialogue comme la théorie des actes de langage, les maximes de Grice…, nous avons envisagé différents modèles de dialogue sous trois angles principaux : modèles orientés plans intentionnels, modèles structurels et modèles orientés négociation.

Les modèles dits orientés plans considèrent le dialogue comme une activité coopérative et intentionnelle. Ces modèles spécifient normalement une séquence d'opérations dans un plan permettant d'atteindre des buts depuis un état initial. Le plan, dans ces modèles, est hiérarchiquement organisé par des actions qui puissent conduire à un but souhaité une fois qu'on le suit correctement.

L'approche fondée sur la structuration est également utilisée pour modéliser le dialogue homme-machine. Les modèles structurels visent à décrire la structure des dialogues à partir de marqueurs linguistiques et d'indices pragmatiques [Lehuen, 1997]. La grammaire formelle est éventuellement le fondement théorique dans la plupart des travaux qui s'appuient sur une modélisation structurelle du dialogue.

En négligeant l'idée de modélisation de dialogue fondée sur la planification des intentions, et sur la structuration des interventions, les modèles orientés négociation utilisent les *enjeux conversationnels* [Baker, 1994], ou bien les jeux de dialogue [Maudet, 2001], comme idée principale pour modéliser le dialogue. Le dialogue est donc considéré comme une activité conjointe, qui engage au moins deux interlocuteurs.

Au cours de la présentation des modèles, nous avons présenté le modèle stratégique, qui considère le dialogue homme-machine comme un jeu dit jeu de dialogue. Dans ce jeu, l'utilisateur et le système sont engagés en visant à effectuer un but de manière rationnelle avec les conventions sociales considérées. Le dialogue est construit de façon coopérative et co-interactive : tous les deux – système et utilisateur – coordonnent leurs actions et leurs stratégies afin de satisfaire le but. L'avancement du dialogue est réalisé par des coups du jeu, dont chacun est un acte de dialogue, mais à côté du système, il est mis en place des règles portant sur les attitudes de l'utilisateur qui lui permettent de trouver la stratégie optimale pour conduire le dialogue de manière efficace.

Avec le modèle stratégique de dialogue, nous allons présenter dans le chapitre suivant la gestion générique de dialogue, qui représente toutes les activités dans le contrôleur du dialogue – le « *cerveau* » du système de dialogue (cf. sections 1.4.2.4 et 5.4.4 pour le détail).

Chapitre 3
GESTION GÉNÉRIQUE DE DIALOGUE

Dans ce chapitre, en partant du modèle stratégique présenté au chapitre précédent, nous allons voir ce modèle sous l'angle de la pratique, ainsi que de la mise en œuvre. En mettant l'accent sur la gestion de stratégies et de buts de dialogue, le modèle de gestion que nous proposons offre des avantages portant sur la généricité, un des points cruciaux de cette thèse. Le traitement des erreurs au niveau du contrôleur du dialogue sera également abordé, afin d'apporter une meilleure souplesse et une des grandes robustesses au système de dialogue.

3.1 Introduction

La gestion de dialogue prend en compte les activités qui résident au sein du contrôleur du dialogue d'un système de dialogue. Comme nous l'avons signalé dans la section 1.4.2.4 du premier chapitre, le contrôleur du dialogue doit avoir les rôles suivants :

- coordonner toutes les interactions entre l'utilisateur et le système,
- assurer l'avancement, la cohérence du dialogue,
- être le pont entre les énoncés de l'utilisateur et l'action réelle sur le monde géré par le contrôleur de la tâche.

Afin d'assurer ces rôles, le contrôleur du dialogue nécessite effectivement un mécanisme de gestion souple et efficace. Nous présentons maintenant notre approche qui vise à proposer une gestion générique du contrôleur du dialogue. En se fondant sur le modèle stratégique de dialogue, notre modèle de gestion distingue deux parties principales : la gestion de stratégie et celle de buts de dialogue. De plus, pour apporter de la souplesse au système de dialogue, nous abordons également des problèmes concernant les erreurs possibles dans tous les modules d'un système de dialogue, et puis nous mettons l'accent sur les erreurs, les incidents dans le contrôleur du dialogue.

Nous proposerons par la suite des solutions mises en place dans ce contrôleur afin de diminuer les incompréhensions, les mauvais dialogues, etc., dans un système de DHM.

3.2 Gestion de dialogue

En partant de l'architecture du système de dialogue présenté dans la section 1.4.2 et en supposant que la qualité de reconnaissance de la parole, de compréhension, d'interprétation soit bonne, la gestion du dialogue dans un système de DHM doit normalement assurer des tâches générales suivantes :

- mettre à jour le contexte du dialogue lorsqu'un nouvel événement est observé,
- décider quand parler et quoi dire,
- interagir avec le modèle de tâche pour convertir les énoncés oraux en des commandes, des actions précises que le système doit effectuer.

En exploitant le modèle stratégique de dialogue, nous proposons concrètement une gestion générique du dialogue qui présente plusieurs avantages concernant sa généricité. Dans ce modèle de gestion, nous élaborons deux gestions élémentaires : gestion de stratégies et gestion de buts. De plus, afin de guider l'activité, nous utilisons une connaissance de squelette d'un dialogue sous forme d'un agenda de dialogue. Nous allons voir dans les sections qui suivent les principes de notre modèle de gestion du dialogue.

3.2.1 Thème de dialogue

Afin de mieux gérer le dialogue, les échanges d'un dialogue sont divisés en thèmes différents dont chacun regroupe des topiques portants sur un même sujet. Un thème représente l'activité de dialogue autour d'un sous-but de dialogue, qui peut être considéré comme un constituant formel d'une tâche précise dans le service visé par un système de dialogue. Par exemple, pour la tâche de réservation d'une salle de réunion, il faut mener la conversation autour de trois thèmes : un thème d'identification qui représente le sous-but d'identification du demandeur ; un thème de réservation pour le sous-but de réservation de salle et de matériel ; et un thème de convocation qui représente le sous-but de convocation des participants (si le demandeur le veut).

Si un but est obligatoire (par exemple le thème d'identification), le système doit obliger l'utilisateur à atteindre ce but avant d'attaquer d'autres thèmes. Il y a éventuellement des thèmes que les deux peuvent ignorer (par exemple le thème de convocation de participants). De plus, les thèmes dans l'étape d'échanges peuvent être non linéaires. L'ordre d'attaque d'un thème dépend éventuellement de ses relations avec d'autres thèmes, de sa priorité… (cf. section 3.2.2)

Prenons l'exemple 3.1 suivant, nous constatons que ce dialogue contient donc deux thèmes : thème d'identification (*mots en italique*) et thème de réservation (**mots en gras**). Au début, le thème d'identification n'est pas encore abordé, malgré l'apparition du thème de réservation. Le

système demande aussitôt à l'utilisateur de s'identifier. Une fois que le thème d'identification est achevé (le but d'identification est satisfait), le système retourne au thème de réservation et tous les deux échangent des données autour de ce thème jusqu'à la fin, en ignorant le thème de convocation.

	Enoncé
S	Bonjour, ici Mélina, le service de réservation du laboratoire CLIPS, je vous écoute
U	Bonjour, **j'aimerais réserver la salle Lafayette pour le lundi 20 à 14h**
S	*Puis-je demander votre nom ?*
U	*Je suis Jean Caelen*
S	*Bonjour M. le directeur*, **la capacité de cette salle est de 25 places. Voulez-vous des matériels ?**
U	**Oui, j'ai besoin d'un vidéo projecteur**
S	**Je vous réserve donc la salle Lafayette lundi 20 septembre 2004, à 14h avec un vidéo projecteur…**
U	Parfait, merci bien et au revoir
S	Je vous en prie, au revoir monsieur le directeur

Exemple 3.1 : Un dialogue avec deux thèmes différents

L'enchaînement d'un dialogue (cf. section 2.3.5) peut être également illustré par des thèmes différents comme la figure 3.1 ci-dessous :

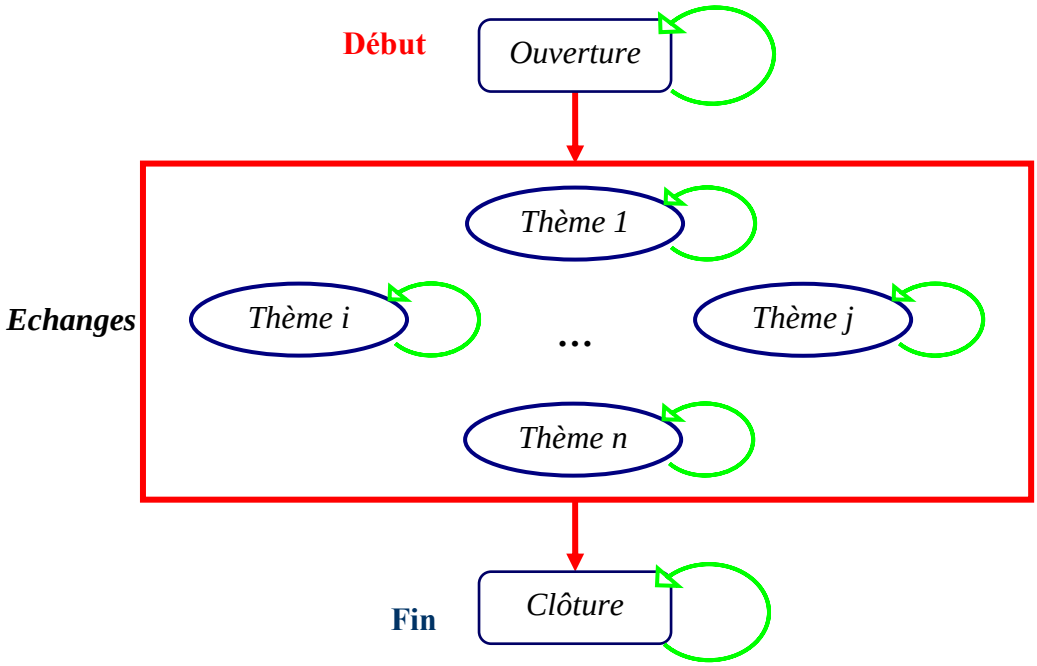


Figure 3.1 : Thèmes dans un dialogue

3.2.2 Agenda de dialogue

Dans notre modèle de dialogue, nous définissons un agenda de dialogue qui est utilisé comme guide de l'activité du dialogue. Il définit structurellement l'enchaînement de dialogue avec trois étapes, qui sont « ouverture », « échanges » et « clôture ». La structure d'un tel agenda est hiérarchiquement organisée sous forme d'un arbre de buts et ses états sont modifiés au fur à mesure des échanges, par l'avancement du but de dialogue et par des énoncés de l'utilisateur et/ou les interactions avec le modèle de la tâche.

L'organisation d'un tel agenda est analytiquement fondée sur les thèmes de dialogue représentant la tâche visée par le système (cf. section 3.2.1). Chaque thème représente donc un sous-but de dialogue. Un sous-but contient éventuellement un ou plusieurs sous sous-buts différents de manière hiérarchique et ainsi de suite.

La décomposition de sous-buts est fondée à la fois sur la structuration et sur l'ordre à réaliser. Les opérateurs ET, ALT et OU sont utilisés pour la décomposition structurelle comme présenté en tableau 3.1 :

Opérateur	Description
ET	Un but est atteint si et seulement si tous ses sous-buts sont atteint.
ALT	Il suffit d'atteindre un seul sous-but pour atteindre le but. Les autres buts sont ignorés.
OU	Au moins un des sous-buts doit être atteint pour atteindre le but.

Tableau 3.1 : Décomposition structurelle de buts

Nous décomposons également un but de dialogue en sous-buts de façon temporelle. Il s'agit d'utiliser des contraintes sur l'ordre de réalisation pour construire l'agenda futur. Les sous-buts sont décomposés en utilisant les opérateurs SEQ, PAR et SIM, comme décrit en tableau 3.2 :

Opérateur	Description
SEQ	Comme une séquence d'ordre, un sous-but ne commence que si le précédent dans la séquence est atteint.
PAR	Deux sous-buts peuvent s'entrelacer au cours de leur réalisation.
SIM	Les sous-buts doivent être réalisés simultanément.

Tableau 3.2 : Décomposition temporelle de buts

De plus, pour rendre toute sa souplesse au système de dialogue homme-machine, le poids d'un but est également introduit dans cet agenda. Nous distinguons les trois types de buts suivants :

- *but caché* : but qui ne s'instancie pas nécessairement au cours du dialogue, l'utilisateur et le système peuvent même ignorer l'existence de ce but. Le but caché est utilisé pour gérer des cas exceptionnels au cours d'un dialogue,

- *but négligeable* : but que l'on peut ignorer momentanément sans interrompre le dialogue; il sera repris plus tard si nécessaire,
- *but impératif* : but qu'il faut impérativement aborder et atteindre.

Par exemple, pour le service de réservation d'une salle de réunion, nous pouvons organiser l'agenda de dialogue comme présenté dans la figure 3.2. Les échanges d'un dialogue, dans ce service, peuvent être divisés en trois thèmes différents : identification, réservation et convocation. Les thèmes d'identification et de réservation sont donc obligatoires pour une demande de réservation d'une salle, tandis que le thème de convocation peut être négligé, si l'utilisateur ne veut pas que le système convoque les participants de sa réunion.

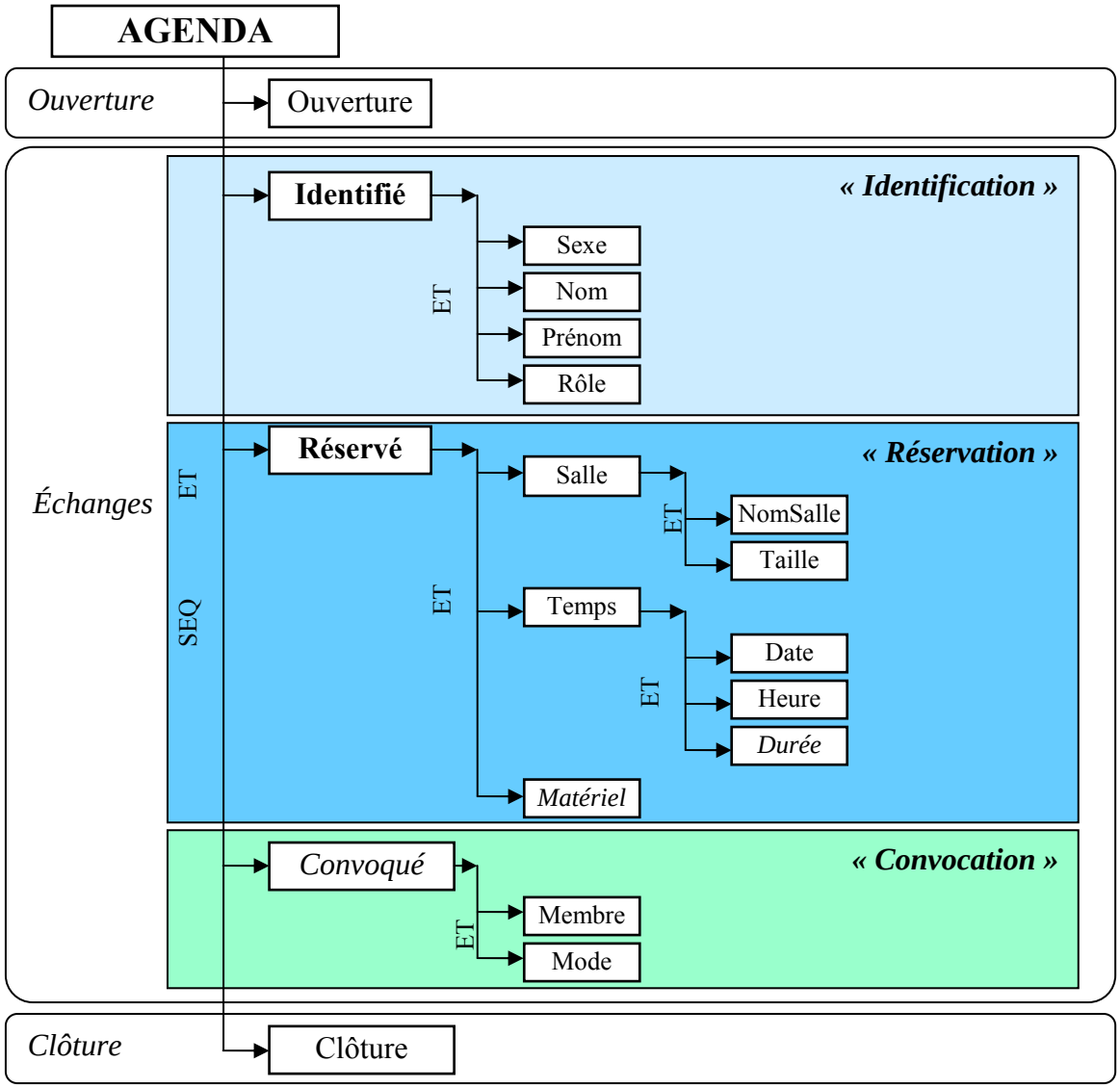


Figure 3.2 : Structure d'un agenda de dialogue.

Dans cette figure, l'agenda de dialogue est donc organisé en trois phases : la phase d'ouverture, la phase d'échanges et la phase de clôture. Les deux phases « *Ouverture* » et

« *Clôture* » sont représentées respectivement par les deux sous-buts « Ouverture », « Clôture ». La phase d'échanges comporte trois sous-buts « Identifié », « Réservé » et « Convoqué » représentant respectivement les trois thèmes « Identification », « Réservation » et « Convocation ». Les sous-buts en gras sont impératifs, par exemple « **Identifié** » et « **Réservé** ». Le sous-but en *italique* signifie un *but négligeable*, par exemple le sous-but *Convoqué* peut être négligé dans le cas où l'utilisateur n'aborde pas le sujet de convocation des participants.

La structure hiérarchique de l'agenda de dialogue est considérée comme un squelette représentant les connaissances statiques du service visé par le système de dialogue. L'information stockée dans chaque élément de cet agenda constitue donc ses connaissances dynamiques.

3.2.3 Tour de parole

Dans notre modèle, un tour de parole se déroule selon la procédure illustrée par la figure 3.3 suivante :

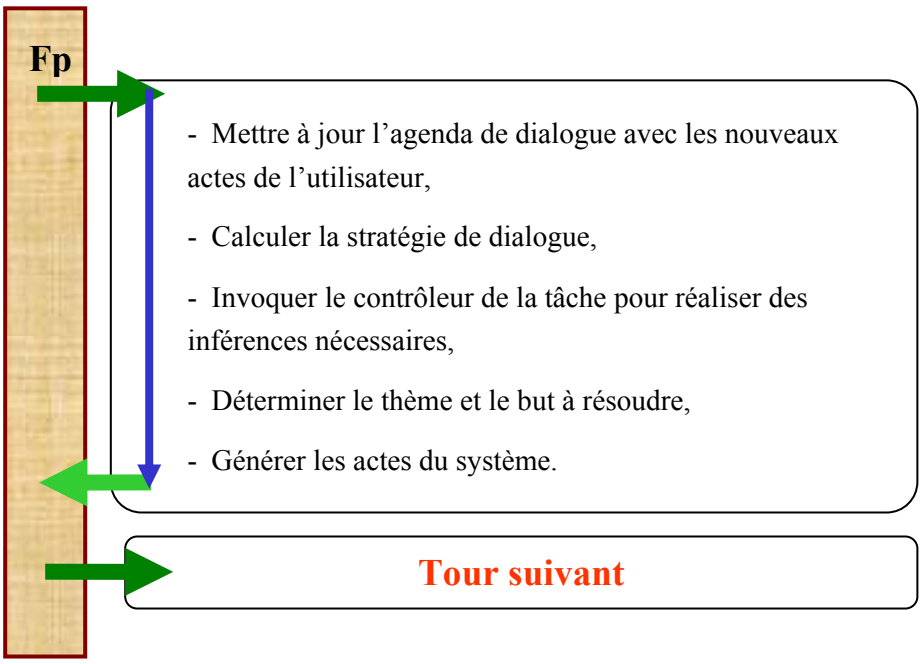


Figure 3.3 : Etapes de traitement dans un tour de parole du contrôleur du dialogue

Les actes de l'utilisateur généré par l'interpréteur sont envoyés directement au contrôleur du dialogue. Celui-ci met à jour tout d'abord l'agenda de dialogue avec l'aide du contrôleur de la tâche. L'agenda de dialogue, qui est organisé hiérarchiquement par un arbre de buts élémentaires, a le rôle d'enregistrer tous les échanges entre l'utilisateur et le système, par exemple les buts mentionnés, spécifiés ou mis à jour par l'utilisateur. Ensuite, il calcule la stratégie adéquate basée sur les forces illocutoires dans les actes de l'utilisateur de ce tour et les actes du système du tour précédent, sur la stratégie précédente, et sur le but actuel. Puis, il demande au contrôleur de la tâche d'effectuer les actions illocutoires contenues dans les actes de l'utilisateur, afin de réaliser des

inférences nécessaires. Les effets de ce fait sont enregistrés dans l'agenda de dialogue et facilitent la détermination du thème et du but de dialogue à résoudre dans ce tour. Finalement, avec l'aide du module de gestion de tâche, le contrôleur du dialogue va générer l'acte du système répondant à l'énoncé de l'utilisateur à base du but et de la stratégie de dialogue.

A la fin d'un tour de parole, le contrôleur du dialogue doit envoyer au module de génération tous les trois éléments principaux : les actes du système, le but et la stratégie de dialogue, et attendre le prochain tour de parole.

3.2.4 Gestion de stratégies

La gestion de stratégies de dialogue se fonde sur des règles qui tiennent compte de l'information d'un acte de dialogue, de la possibilité d'atteindre un but, de l'état actuel du dialogue, des attentes de l'utilisateur, de la stratégie précédemment adoptée, etc [Caelen, 2002]. Le calcul d'une stratégie adéquate est effectué en utilisant d'une part des règles de déclenchement, et d'autre part la logique d'enchaînement des tours de parole pour toutes les situations possibles.

Les règles de déclenchement d'une stratégie sont listées ci-après. Supposons ici que π est le nombre de tours de parole compté à partir du début ; δ est la stratégie de dialogue ; U et M désignent respectivement l'utilisateur et la machine (système) ; p est le contenu propositionnel.

- La stratégie est *directive* au début du dialogue ou dans une situation d'incompréhension ou bien aucune information n'est produite par l'utilisateur :

$$((\textit{Incompréhension}) \vee (\pi = 0) \rightarrow (\delta_\pi = \textit{directive})$$

- La stratégie devient réactive si le nombre de tours de parole π depuis la précédente action de type F dépasse un certain seuil π_0. Elle est également activée pour conclure le dialogue ou au cas où l'utilisateur manifeste son ordre en modalité impérative :

$$((\pi > \pi_0) \vee (F^{F}_{U\pi}(\text{impérative})) \vee ((F^{S}_{M\pi\text{-}1}(\text{clôture})) \rightarrow (\delta_\pi = \textit{réactive})$$

  Elle est appliquée également dans le cas où la proposition du système n'est pas acceptée par l'utilisateur si la stratégie du tour de parole précédent est coopérative, afin de le forcer à bien suivre le dialogue :

$$(\delta_{\pi\text{-}1} = \textit{coopérative}) \wedge (F^{FS}_{M\pi\text{-}1}(p)) \wedge (\neg F^{S}_{U\pi}(p)) \rightarrow (\delta_\pi = \textit{réactive})$$

  Dans le tour de parole précédent, si la stratégie est négociée et dans le tour actuel, si l'utilisateur donne une information *correcte* – c'est-à-dire que cette information est confirmée par le système, alors la stratégie est transformée en stratégie réactive (une chose qui est satisfaite à la fois par l'utilisateur et par le système ne peut absolument pas être discutée) :

$$(\delta_{\pi\text{-}1} = \textit{négociée}) \wedge (F^{S}_{U\pi}(p)) \wedge (F^{S}_{M\pi}(p)) \rightarrow (\delta_\pi = \textit{réactive})$$

- La stratégie passe en mode coopératif si l'énoncé de l'utilisateur donne une information incomplète pour le but actuel :

$$(F^{S}_{U\pi}(p) \wedge (\text{incomplet}(p)) \rightarrow (\delta_{\pi} = \textit{coopérative})$$

Dans le tour de parole précédent, si le système utilise la stratégie négociée et donne à l'utilisateur une information qui est réfutée par lui-même, alors la stratégie coopérative doit être appliquée, afin de trouver une solution raisonnable pour tous les deux :

$$(\delta_{\pi-1} = \textit{négociée}) \wedge (F^{S}_{M\pi-1}(p)) \wedge (\neg F^{S}_{U\pi}(p)) \rightarrow (\delta_{\pi} = \textit{coopérative})$$

Cette stratégie est également activée dans la situation où le tour de parole précédent contient la stratégie constructive, une question posée par le système, mais dans le tour actuel, l'utilisateur réfute cette question :

$$(\delta_{\pi-1} = \textit{constructive}) \wedge (F^{FS}_{M\pi-1}(p)) \wedge (\neg F^{S}_{U\pi}(p)) \rightarrow (\delta_{\pi} = \textit{coopérative})$$

- La stratégie constructive est invoquée si le dialogue ne progresse pas :

$$(b_{\pi} = b_{\pi-1}) \rightarrow (\delta_{\pi} = \textit{constructive})$$

- La stratégie devient négociée s'il y a une réfutation de l'utilisateur :

$$(\neg F^{S}_{U\pi}(p) \wedge F_{M\pi-1}(p) \rightarrow (\delta_{\pi} = \textit{négociée})$$

3.2.5 Gestion de buts

La gestion de buts de dialogue doit assurer le déroulement du dialogue, en envisageant les états des buts de dialogue, ainsi que le mécanisme de manipulation de ces buts.

3.2.5.1 Evolution du but de dialogue

L'état d'un but est affecté par un acte de dialogue. Par exemple, l'énoncé « *ici Paul Dupont je voudrais réserver une salle* » peut être interprété comme les deux actes de dialogue suivants : $\mathbf{F}^{\mathbf{S}}$[Prénom*(Paul)* & Nom*(Dupont)*]&$\mathbf{F}^{\mathbf{F}}$[*x*, Réserver*(x)* & NomSalle*(x)*].

En appliquant les maximes de Grice (cf. section 2.1.3.1), nous pouvons dire que ces actes manifestent directement le but de dialogue $?b_U = (\exists x)$: salle(x) $\wedge$ réservé(x,U) avec U = (Paul Dupont). De façon générale, nous distinguons les attitudes de l'utilisateur pour tous les états du but de dialogue illustrés dans le tableau 3.3.

Règles	Attitudes de l'utilisateur
$F^S_U b \wedge \neg b \rightarrow ?b$	U pose un nouveau but en le manifestant
$F^S_U b2 \wedge b1 \rightarrow -b1 \wedge ?b2$	si U manifeste un deuxième but b2 alors qu'un autre but b1 est déjà en cours, on met ce dernier en attente (car on ne traite le dialogue que sur un fil, c'est-à-dire échange par échange).
$\ddagger b \wedge F^S_U b \rightarrow @b$	U n'a pas de raison de maintenir un but satisfait.
$F^S(\dagger b1) \vee (\dagger b1 \wedge F^S b2 \wedge (b1 \neq b2)) \rightarrow \ddagger b1$	Le but atteint est satisfait après accord explicite ou bien si U ne le conteste pas (de manière implicite).
$\dagger b \wedge F^S_U(\neg b) \rightarrow @b$	si un but est atteint et que U le conteste, on l'abandonne.
$F^S_U (@b) \rightarrow @b$	U peut décider d'abandonner un but de propos délibéré.

Tableau 3.3 : Evolution du but de dialogue au cours d'un dialogue

3.2.5.2 Mécanisme de gestion de buts de dialogue

La gestion des buts relève du paradigme général de la planification du dialogue qui est considérée comme une image spécifiée les tours de paroles. Cette planification doit être représentée sous forme du squelette de l'agenda de dialogue qui joue donc un rôle essentiel dans la gestion des buts. Sa structure hiérarchique est construite en se fondant à la fois sur la décomposition structurelle (avec les opérateurs ET, ALT et OU) et temporelle (en utilisant les opérateurs SEQ, PAR et SIM). De plus, le poids d'un but est également considéré afin de donner toute sa souplesse au système de dialogue.

Les règles présentées dans le tableau 3.3, permettant de déterminer les comportements de l'utilisateur et du système au niveau du but de dialogue, doivent être appliquées dans la gestion des états des buts dans l'agenda.

Evidemment, pour savoir si un but est atteint ou non, le contrôleur du dialogue doit interagir avec le contrôleur de la tâche pour demander l'information concernée. Les autres états du but dialogique sont déterminés soit directement par l'acte de l'utilisateur, soit par les règles décrites en tableau 3.3.

3.2.6 Généricité

La gestion de dialogue est bien évidemment effectuée par le module « *Contrôleur du dialogue* » qui a normalement pour rôle de conduire le cycle de dialogue, de déterminer le but de dialogue, de calculer la stratégie adéquate, et de contrôler le générateur de l'acte de dialogue du système. Afin d'assurer la généricité du système de dialogue, la gestion du dialogue ne se fonde que sur les deux éléments importants : la gestion de stratégies et celle de buts du dialogue. De plus, notre modèle de gestion de dialogue ne s'est appuyé que sur les trois éléments principaux suivants : l'acte de dialogue, le but de dialogue et la stratégie de dialogue.

Comme dans la section 2.3 au chapitre 2, nous constatons que :

- l'acte de dialogue est analysé par le module *Interpréteur* et donc il est indépendant du modèle de tâche au niveau du contrôleur du dialogue,
- le but de dialogue est déterminé à l'aide de l'agenda de dialogue,
- la stratégie de dialogue est calculée par le contrôleur du dialogue et il ne concerne donc pas le modèle de la tâche.

Pour ces raisons, notre modèle de gestion de dialogue est évidemment générique et séparé du modèle de la tâche. Cette remarque est très importante et nous offre plusieurs avantages pour construire des systèmes de dialogue, par exemple la réutilisabilité du contrôleur du dialogue, l'extensibilité du système, etc. La validation de cette remarque sera illustrée via l'expérimentation présentée au chapitre 5, spécialement par la mise en œuvre du contrôleur du dialogue, décrite à la section 5.4.4

3.3 Traitement des erreurs

Dans la plupart des systèmes de dialogue actuels, les problèmes qui se posent en permanence restent toujours autour des incidents, soit par des incompréhensions entre le système et l'utilisateur, soit par les limites du système… En effet, il existe plusieurs raisons qui peuvent provoquer les erreurs au cours d'un dialogue, comme le problème de reconnaissance de la parole, le problème de compréhension, le problème d'interprétation et quelquefois, même pour le module de contrôleur du dialogue. C'est pourquoi, dans le DHM, l'efficacité d'un modèle de dialogue est estimée en évaluant également sa capacité de traitement des problèmes de mauvaises communications.

3.3.1 Situations

Dans le domaine du DHM, des erreurs apparues au cours d'un dialogue restent un des principaux problèmes pour les chercheurs. Elles sont malheureusement omniprésentes dans tout système de DHM actuel.

L'objectif de la recherche des problèmes, des erreurs dans un dialogue a bien évidemment pour but de rendre au système la capacité de dialoguer de manière plus naturelle, efficace et intelligente.

Les raisons qui causent de tels problèmes dans le DHM peuvent résider dans tous les modules dans un système de DHM. Dans son processus de traitement interactif, chaque module (cf. section 1.4.2) prend évidemment son rôle important par rapport au succès de dialogue, et donc un problème anormal apparu dans un module peut provoquer un mauvais dialogue pour le système complet. Pratiquement, la reconnaissance automatique de la parole, la compréhension et l'interprétation sont les modules les plus sujets à causer des erreurs pour le DHM. Par exemple, le dialogue illustré dans l'exemple 3.2 montre des erreurs provenant du module de reconnaissance de la parole qui sont présentes quasiment dans tous les énoncés de l'utilisateur.

Dans cet exemple, le système de dialogue, que nous avons élaboré, a bien pris en compte des mots essentiels, des mots apportant les sens. Il est donc capable de surmonter de telles erreurs. Néanmoins, il s'avère également qu'il y a des cas qui bloquent ce système et produisent des mauvais dialogues (cf. section 5.4).

S : Bonjour, ici le service de réservation du laboratoire CLIPS, je suis Mélina et vous ?
U : *bonjour jean* <bonjour c'est jean>
S : désolé, j'ai plusieurs personnes ayant votre prénom ! Quel est votre nom ?
U : *ce jugement caelen* <je suis Jean Caelen>
S : Bonjour monsieur le directeur ! Quelles sont vos préférences de réservation ?
U : *il faut une salle de vingt personnes pour lundi prochain* <il me faut une salle de vingt personnes pour lundi prochain>
S : la salle aquarium serait à votre disposition. A quelle heure s'il vous plaît ?
U : *quatorze heures* <à quatorze heures>
S : je vous réserve donc la salle aquarium à 14 heures lundi le 27 septembre 2004…
U : très bien merci bien au revoir
S : au revoir monsieur Caelen à bientôt.

Exemple 3.2 : Un dialogue avec des erreurs

Analysons maintenant certains systèmes de dialogue oral réels. L'analyse du corpus du système CMU Communicator, réalisée par [Carpenter et al., 2001], indique que 32% des énoncés contiennent des erreurs de compréhension. D'autres résultats similaires, qui sont mesurés dans les analyses réalisées sur d'autres systèmes, sont présentés dans le tableau 3.5. Dans ce tableau, nous pouvons constater que typiquement de 20 à 40% des énoncés ne sont pas compris correctement par le système de dialogue. Cela montre l'impact vraiment négatif de l'incompréhension vis-à-vis du système de dialogue.

Système de DHM	Ratio des erreurs sémantiques
CMU Communicator [Carpenter et al., 2001]	32%
CU Communicator [Rudnicky et al., 1999]	27%
How May I Help You [Walker et al., 2000]	36%
Jupiter [Hazen et al., 2002]	28%
SpeechActs [Yankelovich et al., 1995]	25%

Tableau 3.4 : Proportion des énoncés contenant des incompréhensions dans divers systèmes

3.3.2 Facteurs conduisant à un mauvais dialogue

En reprenant l'architecture générale d'un système de DHM présenté dans la section 1.4.2, nous envisageons des facteurs qui influencent plus ou moins la qualité d'un dialogue oral homme-machine dans chaque module ci-dessous :

1. Erreurs provenant du module de reconnaissance automatique de la parole (RAP) :

   1.1. L'utilisateur prononce des mots, mais le module de RAP ne produit aucune chaîne orthographique à la sortie,

   1.2. Le module de RAP produit une chaîne avec des erreurs, soit combinées, soit isolées : insertion de mots, suppression de mots, substitution de mots. Les raisons de ces erreurs sont nombreuses : l'utilisateur a prononcé des mots hors-vocabulaire, n'a pas correctement parlé ; le bruit environnement était trop fort, la qualité du module acoustique, ainsi que du langage n'était pas bonne… Ce type d'erreurs est malheureusement omniprésent dans tous les systèmes de reconnaissance de la parole et pose effectivement un grand défi au domaine du DHM.

2. Erreurs du module de compréhension en supposant que la chaîne orthographique sortie du module de RAP est correcte :

   2.1. Analyse sémantique impossible : aucun schéma sémantique n'est sorti,

   2.2. Analyse sémantique incomplète : des morceaux dans l'énoncé de l'utilisateur ne sont pas exploités et cela mène donc à un manque d'information,

   2.3. Analyse sémantique erronée : le schéma sémantique sorti ne représente pas correctement le sens sémantique de l'énoncé,

3. Erreurs parues dans le module d'interprétation pragmatique :

   3.1. Erreur de références,

   3.2. Erreur de prédicat,

   3.3. Erreur d'implicature conversationnelle,

   3.4. Erreur de thème,

   3.5. Erreur de l'acte de dialogue : la force illocutoire F_A n'est pas correcte, le contenu propositionnel ne relie pas à l'énoncé…,

   3.6. Confusion après une incompréhension : l'historique de ce module peut être bouleversé après des incompréhensions - l'enchaînement, la cohérence, …, est violée,

4. Erreurs portant sur la gestion du dialogue :

   4.1. Erreur de stratégie : en raison des erreurs des modules précédents, le contrôleur du dialogue ne calcule pas correctement la stratégie adéquate du système,

4.2. Erreur de but : via l'agenda de dialogue, le contrôleur du dialogue reçoit des mauvais retours provenant du contrôleur de la tâche (à cause des fautes de données, de problèmes de synchronisation, …) qui constituent éventuellement des erreurs durant la détermination du but de dialogue,

4.3. Mauvaise anticipation : les attentes calculées par le système dans la stratégie réactive et dans les cas ayant des erreurs ne sont pas appropriées,

5. Erreurs dans la génération :

5.1. Erreur de force illocutoire concernant les actes du système,

5.2. Enoncé mal formé : la concaténation de l'énoncé du système est mal formé et engendre des difficultés à comprendre de la part de l'utilisateur.

A notre connaissance, le synthétiseur de la parole est le seul module qui ne provoque pas des erreurs, à moins de négliger l'intonation, l'enchaînement des mots, l'accent… par rapport à l'exigence de l'utilisateur.

Pour le module de reconnaissance de la parole, bien que le taux de reconnaissance soit augmenté au fil du temps avec des recherches innovantes, le résultat de ce module pose vraisemblablement encore des difficultés pour la compréhension. Néanmoins, les erreurs de ce module ne concernent pas, pour l'instant, l'objectif de cette thèse.

En acceptant les erreurs possibles provenant du module RAP, nous nous concentrons sur les erreurs éventuelles provenant des modules de compréhension, d'interprétation et surtout du contrôleur du dialogue.

Les analyses des facteurs défaillants d'un dialogue, dans la section précédente, nous ont conduites à diviser les erreurs d'un dialogue oral en deux catégories principales : l'incompréhension et le malentendu. En effet, comme dans la communication quotidienne entre deux être humains, ce qui conduit à un mauvais dialogue est dû au fait que l'un parle de choses que l'autre ne comprend pas (incompréhension), ou bien qu'il parle d'une chose mais l'autre la traduit en une/des autres choses (malentendu) :

- *Incompréhension* **:** Formellement, l'incompréhension vient du fait que le système de DHM n'arrive pas à obtenir les schémas sémantiques de manière correcte quand l'utilisateur prononce sa demande. Précisément, l'incompréhension est le problème qui réside essentiellement dans le module de compréhension qui évoque également des erreurs cumulatives dans l'interpréteur et dans le contrôleur du dialogue.

Certes, l'incompréhension dans le DHM est un phénomène important à résoudre. Afin de pouvoir proposer une solution à ce problème, nous distinguons les différents niveaux suivants qui conduisent à l'incompréhension :

- Niveau lexical : l'incompréhension dans ce cas est due à l'emploi par l'utilisateur de mots hors du domaine compréhensible du module de compréhension,

- Niveau syntaxique : l'utilisateur prononce un énoncé qui ne respecte pas les règles syntaxiques ou qui contient des termes populaires non couverts par la base des ontologies dans le module de compréhension,
- Niveau sémantique : le système ne peut schématiser les concepts se trouvant dans l'énoncé de l'utilisateur et le module de compréhension ne le comprend donc pas,
- Niveau contextuel : les schémas sémantiques sortis du module de compréhension ne sont pas adéquates aux thèmes connus par l'interpréteur.

Il est clair que les trois premiers niveaux concernent le module de compréhension et le dernier l'interpréteur.

- *Malentendu* **:** Lorsqu'un énoncé de l'utilisateur est compris de manière différente par rapport à son intention, le système de DHM doit affronter le problème de malentendu. Le malentendu est donc le phénomène qui combine les erreurs provenant de divers modules : l'interpréteur pragmatique puis le contrôleur du dialogue et enfin le générateur. Pratiquement, il constitue un problème très difficile à résoudre pour un système de dialogue.

En envisageant des facteurs qui provoquent un mauvais dialogue, nous mettons l'accent sur le traitement des erreurs au sein du contrôleur du dialogue. L'incompréhension est donc la catégorie prioritaire à résoudre.

3.3.3 Stratégie de traitement d'erreurs

Dans le cadre de cette thèse, nous ne nous intéressons qu'à la recherche de la solution pour les incompréhensions au niveau du contrôleur du dialogue. Il s'agit des cas pour lesquels l'incompréhension a déjà eu lieu et pour lesquels le système de DHM doit, dans ces cas là, proposer des solutions différentes pour surmonter ces problèmes. Nous n'aborderons pas les solutions pour les incompréhensions au niveau du module de compréhension ainsi que les malentendus au niveau de l'interpréteur pragmatique, en raison de l'attente des résultats prometteurs dans les autres thèses de notre équipe.

3.3.3.1 Détection par marqueurs dialogiques d'incompréhension (MDI)

Afin de faire la détection des incompréhensions et de les signaler au contrôleur du dialogue, nous proposons de façon générique la notion de « *Marqueur Dialogiques d'Incompréhension* - MDI». L'apparition des MDI signalée au contrôleur du dialogue lui permet évidemment de trouver des solutions appropriées pour résoudre les incompréhensions correspondant à ces MDIs.

Nous distinguons maintenant deux types de marqueurs dialogiques qui indiquent sûrement des incompréhensions au contrôleur du dialogue :

- MDI-1 : Aucun acte de dialogue n'est fourni quand l'utilisateur prononce son énoncé : ce cas se produit quand le module de compréhension ne rend aucun schéma sémantique à la sortie

(facteur 2.1) ou bien quand tous ses schémas sémantiques sont hors du contexte actuel de l'interpréteur et donc qu'il produit un acte vide de l'utilisateur (facteur 2.3). MDI-1 est donc défini comme un marqueur d'incompréhension complète et le module de compréhension doit certainement signaler ce marqueur au système.

- MDI-2 : Il y a évidemment des cas d'incompréhension partielle (facteur 2.2). Dans ce cas, des morceaux de l'énoncé de l'utilisateur, qui apportent éventuellement des données, sont ignorés par le module de compréhension. L'interpréteur doit par conséquent déterminer l'apparition de ce marqueur en analysant la cohérence des actes du système au tour précédent de parole et les actes actuels de l'utilisateur : si cohérent, il ignore ces morceaux, sinon, il produit le marqueur MDI-2 pour le contrôleur du dialogue. Ce marqueur est ainsi considéré comme un marqueur partiel.

En s'appuyant sur ces marqueurs, le contrôleur du dialogue doit par la suite trouver des solutions adéquates en visant comme l'objectif de rendre efficace la capacité de dialogue du système. Les marqueurs d'incompréhension MDI signalés au contrôleur du dialogue sont représentés sous forme d'un acte de faire-savoir, par exemple : F^{S}(MDI-1).

3.3.3.2 Tactiques du système devant MDI

Dans le cas où le CD reçoit des marqueurs des incompréhensions MDIs, il peut traiter ces MDIs en appliquant des tactiques, illustrées par le tableau 3.6.

Tactiques	Exemples	MDI
Confirmation explicite	Voulez vous la salle Lafayette ? δ = réactive, F^{FS}(Salle(lafayette))	MDI-2
Confirmation implicite	En ce qui concerne l'Aquarium, quelle date voulez-vous réserver ? δ = coopérative, F^{S}(Salle(Aquarium)) $\wedge F^{FS}$(Date(x))	MDI-1 MDI-2
Désambiguïsation	Voulez vous le vidéo projecteur Sony ou Philips δ=coopérative, F^{P}(Matériel(Sony,Philips))	MDI-1 MDI-2
Suggestion de solution	Je suppose que vous voulez réserver une salle, voulez vous la salle Aquarium ? δ=réactive, F^{FS}(Salle(Lafayette,Aquarium))	MDI-1
Demande de répéter	Pourriez vous répéter quelle salle vous voulez ? δ = directive, F^{FS}(Salle(x))	MDI-1
Notification d'incompréhension	Je suis désolée mais je ne comprends pas bien ce que vous avez dit. δ=directive, F^{S}(Erreur(imcompréhension))	MDI-1

Tableau 3.5 : Tactiques du système devant MDI

L'initiative de demande de confirmation de l'utilisateur est invoquée dans le cas où il y a un doute concernant son souhait. Envisageons les exemples 3.3 et 3.4 :

	Enoncé	Actes de dialogue
S	Quelles sont vos préférences de salle ?	F^{FS}(Salle(x))
U	Je veux la salle *la fayette*	F^{S}(Salle(x)) $\wedge F^{S}$(MDI-2)
S	voulez vous la salle Lafayette ?	F^{FS}(Salle(*lafayette*))

Exemple 3.3 : Exemple de confirmation explicite

La suite de l'erreur du module de reconnaissance provoque par conséquent l'incompréhension partielle du module de compréhension pour l'énoncé de l'utilisateur dans l'exemple 3.3. Il demande donc sa confirmation explicite avec la stratégie réactive afin de continuer à dialoguer.

	Enoncé	Actes de dialogue
S	J'ai la salle aquarium…	F^{S}(Salle(aqurium))
U	Je ne sais pas mais j'ai besoin d'*un espace* de vingt places	F^{S}(Taille(20)) $\wedge F^{S}$(MDI-2)
S	En ce qui concerne l'Aquarium, quelle date voulez-vous réserver ?	F^{S}(Salle(*aquarium*)) $\wedge$ F^{FS}(Date(x))

Exemple 3.4 : Exemple de confirmation implicite

Dans l'exemple 3.4, devant l'incompréhension partielle causée au niveau lexical, le système doit nécessairement appliquer la tactique de demander la confirmation implicite de l'utilisateur en appliquant la stratégie coopérative : s'il ne la réfute pas, alors le système considère cette information comme sûre et continue à traiter.

La désambiguïsation, comme illustrée par l'exemple 3.5, peut être utilisée avec les marqueurs MDI-1 et MDI-2 tandis que la suggestion, présentée dans l'exemple 3.6, n'est utilisée que dans le cas des incompréhensions complètes, de manière à clarifier le dialogue et la tâche. Supposons, par exemple, que les deux salles concernées ont plus de dix places dans l'exemple 3.5.

	Enoncé	Actes de dialogue
S	Quelle salle voulez vous ?	F^{FS}(Salle(x))
U	La dernière **sale** <*la dernière salle*> (ou *la sale* pour dix personnes)	F^{S}(MDI-1) (ou F^{S}(Taille(>10) $\wedge$ F^{S}(MDI-2))
S	Voulez vous la salle Lafayette ou Aquarium ?	F^{P}(Salle(*aquarium,Lafayette*))

Exemple 3.5 : Tactique de désambiguïsation

	Enoncé	Actes de dialogue
S	Bonjour monsieur Dupont, je vous écoute …	F^S(Nom(Dupont))…
U	J'veux une **sale** < *je veux une salle*>	F^S(MDI-1)
S	Je suppose que vous voulez réserver une salle, voulez vous la salle Aquarium ?	F^{FS}(réservé(x)) and F^S(Salle(*Aquarium*))

Exemple 3.6 : Tactique de suggestion

Les deux dernières tactiques, demande de répéter et notification d'incompréhension, ne sont appliquées que dans le cas où aucune information n'est trouvée par les deux modules de compréhension et d'interprétation. Dans ce cas, la stratégie directive doit être utilisée, afin de conduire l'utilisateur à continuer le dialogue. La notification d'incompréhension peut être présentée comme l'exemple 3.7.

	Enoncé	Actes de dialogue
S	Quelles sont vos préférences de salle ?	F^{FS}(Salle(x))…
U	J'en sais rien moi…	F^S(MDI-1)
S	Désolé mais je ne comprends pas ce que vous dites.	F^S(Incompréhension)

Exemple 3.7 : Notification d'incompréhension

3.3.3.3 Stratégie de traitement d'incompréhension

La stratégie de traitement des erreurs est effectivement fondée sur les marqueurs dialogiques d'incompréhension, ainsi que sur les calculs des attentes de l'utilisateur. Nous distinguons les attitudes suivantes portant sur l'état du système par des MDIs signalés au contrôleur du dialogue :

- Si l'utilisateur et le système arrivent au thème de clôture de dialogue et qu'un MDI est signalé au contrôleur du dialogue (soit MDI-1, soit MDI-2), le contrôleur du dialogue manifeste le souhait de terminer le dialogue en appliquant la stratégie réactive :

  $b_{\pi-1}$(Clôture) $\wedge F^S{}_{U\pi}$(MDI) ➔ (δ = réactive) $\wedge F^S{}_{M\pi}$(Clôture)

- Si le but de dialogue actuel b est atteint ou satisfait, le système pose un nouveau but en demandant une confirmation implicite et en appliquant la stratégie coopérative pour une incompréhension MDI-2 et directive pour MDI-1 :

  ($b_{\pi-1}$ >= atteint) $\wedge F^S{}_{U\pi}$(MDI-2) ➔ (δ = coopérative) $\wedge F^S{}_M(b_{\pi-1})$ and $F^{FS}{}_M(?b_\pi)$

  ($b_{\pi-1}$ >= atteint) $\wedge F^S{}_{U\pi}$(MDI-1) ➔ (δ = directive) $\wedge F^S{}_{M\pi}(b_{\pi-1}) \wedge F^{FS}{}_{M\pi}(?b_\pi)$

- Si le système est en train de demander une information (le but actuel b n'est pas encore atteint), alors le système applique la stratégie réactive en essayant de rattacher à ce but des données qui ne sont pas encore consommées par les deux modules de compréhension et d'interprétation. Une suggestion est donc introduite pour guider l'utilisateur :

$F^{FS}_{M\pi-1}(p) \wedge F^{S}_{U\pi}(MDI) \rightarrow (\delta = \text{réactive}) \wedge F^{FS}_{M\pi}(q)$

dont q = inférer(b,MDI) (le calcul de q se fondant sur le but actuel et MDI)

- Si le système est en train de donner une information et que le but actuel n'est pas encore atteint, alors il informe l'état actuel de ce but en cas d'incompréhension :

  $F^{S}_{M\pi-1}(p) \wedge F^{S}_{U\pi}(MDI) \rightarrow (\delta = \text{coopérative}) \wedge F^{S}_{M\pi}(b)$

- Si le nombre d'incompréhensions consécutives N_{Incomp} dépasse un certain seuil N_{max} (nous choisissons actuellement la valeur du seuil $N_{max} = 3$), alors le système applique la stratégie directive en utilisant la demande de répéter ou la notification d'incompréhension :

  $(N_{Incomp} > N_{max}) \wedge F_{M\pi}(p) \wedge F^{S}_{U\pi}(MDI) \rightarrow (\delta = \text{directive}) \wedge F^{S}_{M\pi}(MDI) \wedge F_{M\pi}(p)$

(π désignant le nombre de tours de parole à partir du début ; δ représentant la stratégie de dialogue ; b pour le but de dialogue ; U et M désignant respectivement l'utilisateur et la machine (système) ; p représentant le contenu propositionnel.

L'extraction d'information complémentaire dans les morceaux de l'énoncé qui ne sont pas consommés par le module de compréhension se fonde sur les considérations suivantes :

- Mots apportant une valeur fixe, indéniable : par exemple « oui, ok, d'accord, non, ne … pas », etc.

- Nombre de mots dans l'énoncé, dans chaque morceau n'ayant pas été consommé : par exemple pour la question du système *« est ce que vous voulez réserver la salle C115 ?* » et si l'utilisateur prononce une série de mots quelconques « blabla blabla blabla … », nous pourrions imaginer que la réponse à cette question peut être « NON », car nous supposons par là qu'une réponse longue est une justification d'une négociation

- Connaissances provenant d'autres modalités comme la moue de l'utilisateur, son intonation, ses gestes…

Ainsi, l'extraction de telles informations complémentaires ne rentrent pas dans le cadre de cette thèse en raison de la généricité et de ses objectifs.

L'expérimentation liée à ces divers traitements est déjà mise en œuvre dans le système de dialogue Mélina que nous allons présenter au chapitre 5.

3.4 Conclusion

Nous avons abordé dans ce chapitre le modèle de gestion générique de dialogue. Partant du modèle stratégique que nous avons présenté à la section 2.3 du chapitre 2, nous avons élaboré un modèle de gestion, qui se concentre sur la gestion de stratégies et de but de dialogue. La stratégie de dialogue incarne véritablement le moyen de faire converger un dialogue, pour que l'utilisateur puisse atteindre rapidement son but de dialogue.

L'activité du modèle stratégique ne se fonde que sur trois éléments principaux : les actes, le but et les stratégies de dialogue. Cependant, nous constatons que : l'acte de dialogue est identifié par le module Interpréteur et donc qu'il est indépendant du modèle de tâche au niveau du contrôleur du dialogue ; le but de dialogue est déterminé à l'aide de l'agenda de dialogue ; et la stratégie de dialogue n'est calculée que par le contrôleur du dialogue et donc ne concerne pas le modèle de tâche. Pour ces raisons, notre modèle de gestion de dialogue est évidemment générique et sa dépendance au modèle de tâche est déjà amoindrie de manière maximale. Cela apporte beaucoup d'avantages comme la réutilisabilité, la portabilité, l'adaptabilité…

De plus, afin de bien gérer l'enchaînement du dialogue, nous avons proposé la notion de « thème de dialogue » avec l'idée de diviser un dialogue en plusieurs thèmes différents. Un thème, qui est déjà initié par un locuteur, entre dans le jeu de dialogue et doit être exploré avant que tous les deux puissent continuer à ouvrir d'autres thèmes. Du point de vue du jeu de dialogue, un thème de dialogue est considéré comme un sous-jeu dans un jeu complet de dialogue.

A partir de la notion de thème, nous avons élaboré la notion de « agenda de dialogue » qui représente structurellement l'enchaînement de dialogue avec trois étapes, qui sont « ouverture », « échanges » et « clôture ». La phase d'échanges est constituée par les thèmes concernant le service visé par le système de dialogue. La structure d'un agenda est hiérarchiquement organisée sous forme d'un arbre de buts. La décomposition de buts est fondée à la fois sur la structuration (par les opérateurs ET, ALT et OU), et sur l'ordre à réaliser (en utilisant les opérateurs SEQ, PAR et SIM). Afin d'apporter la souplesse au système de dialogue, le poids d'un but est également introduit dans l'agenda de dialogue, en distinguant un but caché, un but négligeable et un but impératif.

Dans notre modèle de gestion de dialogue, le problème de traitements des erreurs s'adressant au système de dialogue est également pris en compte. Nous avons soigneusement abordé les facteurs qui conduisent à des erreurs du système dans chaque module de l'architecture présentée en section 1.4.2 au premier chapitre. Toutes ces erreurs sont par la suite divisées en deux catégories : incompréhension et malentendu. Le malentendu est le problème le plus difficile à résoudre, non seulement pour le système, mais également dans le cadre de la communication quotidienne entre les hommes. Nous nous concentrons donc sur le traitement des incompréhensions, mais surtout sur le contrôleur du dialogue, le « cerveau » du système de dialogue. L'incompréhension est détectée à la fois au niveau du module de compréhension et d'interprétation en distinguant deux marqueurs dialogiques de dialogue : MDI-1 signifiant l'incompréhension complète et MDI-2 pour la partielle. Des tactiques de jugement du système sur ces MDIs sont également présentées : demande de confirmation explicite et implicite, désambiguïsation, suggestion, demande de répéter, notification d'incompréhension. Par conséquent, nous avons appliqué des stratégies différentes pour traiter une telle incompréhension au niveau du contrôleur du dialogue, grâce à l'emploi de ces marqueurs.

Nous allons utiliser cette gestion générique, dans le chapitre 5, en l'appliquant à un système de dialogue homme-machine avec des expérimentations effectuées au cours de son implémentation et de son évaluation.

Chapitre 4
MULTISESSION DANS UN DIALOGUE ORAL HOMME-MACHINE

Nous abordons, dans ce chapitre, une nouvelle notion concernant le « dialogue multisession ». Tout d'abord, nous présentons la problématique, qui montre l'exigence de négociation pour un système de dialogue. Le problème de la négociation est abordé sous l'angle de la résolution de conflit dans un dialogue via un ensemble de sessions différentes, qui engagent plusieurs utilisateurs dans le conflit. Sous cet aspect, une session se représente donc comme un sous dialogue entre un seul utilisateur et le système et l'ensemble de ces sessions constituent le dialogue multisession. La modélisation, ainsi que la gestion de dialogue multisession seront ensuite décrites en détail dans ce chapitre. La conclusion sera finalement présentée, afin de résumer le contenu de ce chapitre.

4.1 Introduction

4.1.1 Problématique

Actuellement, la plupart des systèmes de dialogue oral homme-machine (DHM) sont conçus pour un court dialogue, entre le système et un seul utilisateur, en visant à réaliser une seule tâche. Bien évidemment, dans un tel système, le dialogue oral homme-machine ne comporte que l'enchaînement des échanges continus, entre un utilisateur et le système dans l'intention d'atteindre le but souhaité. Des systèmes de DHM, permettant d'utiliser le téléphone interactivement, peuvent être cités ici au fil des dernières années. CMU Communicator [Rudnicky et al., 1999] est dédié au service d'arrangement d'itinéraires et de réservation d'hôtels, de voitures. WITAS [Lemon et al., 2002] réagit comme une interface vocale entre un contrôleur/pilote et un hélicoptère robotique autonome. TRIPS [Ferguson et al., 1998] est un assistant pour un manager humain en visant à construire le plan dans des situations crises… Les interactions dans ces systèmes en particulier, et

dans tous les systèmes actuels de dialogue, ne se déroulent qu'entre un seul utilisateur et le système.

Les analyses d'usage, que nous avons effectuées dans le cadre du projet PVE[16], des hôpitaux, des services d'administration universitaire, chez des professions libérales et des entreprises montrent que les services vocaux sont très utiles dans les applications de renseignement, de demande de service d'urgence, d'accès aux dossiers techniques, de secrétariat (la redirection d'appel, l'organisation d'un rendez-vous à distance, la réservation de salle, l'annulation d'une réunion en urgence, etc.). Ces services sont surtout une solution nécessaire pour les personnels en mobilité, hors du lieu de travail ou bien en travail à distance. Par exemple, un chef de projet, au cours d'une conférence, peut téléphoner au système vocal d'organisation de réunion pour lui demander d'arranger en urgence sa nouvelle réunion d'équipe. Le système, dans cette situation, devrait être capable de réagir comme une secrétaire virtuelle : réserver et éventuellement résoudre des conflits de salle, convoquer des participants, collecter les disponibilités… En résumé, ces analyses nous ont apporté les deux résultats principaux suivant :

- L'utilisateur a toujours besoin d'un service complet. C'est pourquoi nous souhaitons construire des notions, des théories, ainsi que des briques logicielles pour un système de dialogue homme-machine *orienté service*, et non plus seulement orienté tâche.

- La possibilité de conflits au cours d'un dialogue est quasiment omniprésente. Cela conduit donc à l'exigence de la capacité de négociation pour un système de DHM. Les conflits peuvent être intrinsèques entre un utilisateur et des limites du système (ses demandes sont trop exigeantes et hors de la capacité du système, par exemple) ou bien en situation réelle causée par d'autres utilisateurs dans le système.

Bien évidemment, les conflits intrinsèques sont triviaux, par rapport à l'objectif d'un système de DHM qui est adressé à un service concret. Les conflits apparus selon des contextes réels nous intéressent particulièrement sous l'angle de la négociation dans un système de dialogue. La résolution de conflits nécessite donc des interactions non seulement en limite entre un utilisateur et le système, mais en extension entre le système et plusieurs utilisateurs.

Ce chapitre traite des problèmes, ainsi que leur solution, dédiés au dialogue qui engage éventuellement plusieurs utilisateurs dans l'optique de débloquer tous les conflits possibles entre eux, afin d'obtenir un meilleur compromis. Dans la suite de cette thèse, un tel dialogue sera nommé « dialogue multisession ». En effet, chaque session est considérée comme un sous-dialogue entre le système et un seul utilisateur. Nous parlerons tout d'abord de la modélisation du dialogue multisession avec des définitions nécessaires, l'enchaînement de sessions dans un dialogue. Ensuite, nous aborderons le problème de la gestion du dialogue multisession qui concerne à la fois le contrôleur du dialogue et le contrôleur de la tâche. Des exemples importants seront décris au cours de ces parties pour illustrer en détail notre approche.

[16] http://www.telecom.gouv.fr/rnrt/projets/res_01_5.htm

4.1.2 Travail proche : Dialogue multiparty

L'idée d'envisager un dialogue sous l'angle multisession est originale. Cependant, il existe également des recherches relatives aux dialogues qui engagent plusieurs facteurs. Nous allons décrire une approche très intéressante concernant la notion de « dialogue multiparty ».

Dans le cadre du projet « Mission Rehearsal Exercise » [Swartout et al., 2001], qui vise à la formation de conduite de peloton d'armée en utilisant la réalité virtuelle et les humains virtuels, le dialogue engage un soldat stagiaire (lieutenant) simultanément avec plusieurs personnages virtuels (sergent, médecin militaire, d'autres membres du peloton, citoyens locaux, des unités distantes par radio) dont chacun a ses propres buts, intérêts et capacités. Un dialogue contient donc plusieurs conversations avec des contextes différents via des modalités, non seulement orales, mais aussi d'autres modales comme le geste, la vue… Dans [Traum, 2002], l'auteur a considéré un tel dialogue comme un *dialogue multiparty* et propose des sujets concernant cette notion au fil du temps dans ses travaux [Traum et al, 2003 ; Traum, 2004a ; Traum, 2004b]. Selon Traum, les rôles de participant dans un dialogue multiparty sont envisagés par les aspects suivants :

- rôles conversationnels (qui peut recevoir l'énoncé et à qui est adressé cet énoncé…),
- identification d'énonciateur (voix, position, information visuelle…),
- reconnaissance de destinataires (les destinataires sont totalement distingués par rapport aux auditeurs).

La gestion d'interaction est assurée par des modules comme gestion de tour de parole, gestion de canal communicatif, gestion de conversation… De plus, Traum précise également des problèmes concernant la fonctionnalité de l'arrière-plan, les engagements sociaux avec des obligations des destinataires.

Bien que l'idée de dialogue multiparty de Traum couvre à la fois multi-participants et multimodalité, elle ne se concentre pas essentiellement sur les dialogues oraux. C'est pourquoi nous la considérons quand même comme une idée intéressante pour une extension du domaine du DHM.

4.2 Modélisation du dialogue multisession

La modélisation de dialogue multisession sera évidemment élaborée en s'appuyant sur celle d'un dialogue ordinaire, avec le modèle stratégique du dialogue. Cependant, pour s'adapter à de nouvelles situations, nous proposons de nouvelles définitions ainsi que des attitudes, des traitements possibles concernant le dialogue multisession. Tout d'abord, nous définissons la notion de « *dialogue multisession* » ainsi que des notions concernées. Ensuite, nous envisageons des cas qui concernent les dialogues multisession. Et finalement, l'enchaînement de sessions dans un dialogue multisession sera présenté de manière explicite. Le service d'organisation de réunion dans le cadre du projet PVE est enfin considéré pour illustrer notre approche.

4.2.1 Dialogue multisession

Supposons qu'un système de dialogue oral homme-machine S soit conçu pour s'occuper un service complet ayant plusieurs tâches différentes. L'utilisateur interagit évidemment avec le système S par la parole, en langue naturelle.

Au cours d'interactions entre S et un utilisateur, ses intentions d'atteindre un certain but dans S peuvent concerner d'autres utilisateurs dans ce système, reliés éventuellement par des conflits de ressources (par exemple de matériels, de salle…), des demandes cohérentes (convocation d'une réunion, collection de disponibilité de participants…). Ces conflits nécessitent donc des négociations, des interactions entre le système et ces multiples personnes. Pour que l'utilisateur obtienne son but, la résolution sera donc constituée par un dialogue qui engage plusieurs utilisateurs, y compris lui-même.

Nous considérons maintenant la situation concrète suivante : l'utilisateur D (***demandeur***) demande S pour atteindre le but $?b_D$. Néanmoins, ce but ne peut être directement atteint à cause des conflits différents avec d'autres utilisateurs concernés par son but. Ces conflits sont logiquement en relation par des contraintes ET/OU. En souhaitant garder l'objectif de début, D demande donc à S de négocier avec ces personnes pour bien obtenir b_D. Dans le dialogue illustré par l'exemple 4.1, D veut réserver la salle Lafayette qui est déjà occupée par deux réservations dans son créneau (la contrainte de deux réservations ici est évidemment ET). En tenant compte de cette situation, le système lui propose la salle Aquarium qui sera disponible à ce moment là. Néanmoins, cette salle ne satisfait pas D, qui demande alors au système de négocier avec les deux autres utilisateurs concernés pour lui laisser la salle. Dans cet exemple, des conventions sociales doivent pragmatiquement être prises en compte, comme par exemple, la hiérarchie de fonctionnement, pour assurer le droit de chaque utilisateur. Ce dialogue fait donc apparaître des sous-dialogues différents, afin de satisfaire la demande de D.

```
D : ici Dupuis, réservez-moi la salle Lafayette pour demain de 10h
   à midi
S : désolé, cette salle est réservée demain par M. Dupont de 9h à
   11h et par M. Dupaint de 11h à 13h. Par contre, la salle Aquarium
   sera libre ce moment là.
D : Non je ne la veux pas, elle est trop petite. Dites leur que
   j'ai besoin de Lafayette et que je leur demande de changer de
   salle.
S : d'accord, je les appelle et je vous tiens au courant.
```

Exemple 4.1 : Un dialogue qui demande la négociation

Après un processus de négociation avec les patients, le système obtiendra la résolution de b_D (soit atteint $\dagger b_D$, soit abandonné $@b_D$) et il va informer D de ce résultat.

La modélisation du dialogue multisession nous amène à introduire des définitions suivantes :

- ***Patient*** : utilisateur qui a déjà interagi avec le système et obtenu son but. Pourtant, ce but n'a pas encore été effectué (le moment d'exécution n'est pas encore arrivé, par exemple) et de plus, il est en conflit avec le nouveau but b_D manifesté par un autre utilisateur D qui est engagé dans le dialogue actuel avec S.

- ***But en conflit*** : but de dialogue b_D manifesté par D est en conflit avec un autre but déjà satisfait posé par un patient P. Et on note le but en conflit comme $b^F = (?b_D, ‡b_P)$

- ***Arbre de buts en conflit*** : but b_D peut éventuellement être en conflit avec n buts déjà satisfaits de n patients qui sont en relation eux-mêmes par les conditions ET/OU et produit un ensemble de buts en conflit (b^{F1}, … , b^{Fn}). Cet ensemble constitue un arbre binaire ET/OU de buts en conflit A^F dont chaque feuille (sommet) représente un but en conflit avec un patient, et chaque nœud est un opérateur binaire ET ou OU (pour simplifier, on dit seulement A^F comme arbre de buts en conflits)

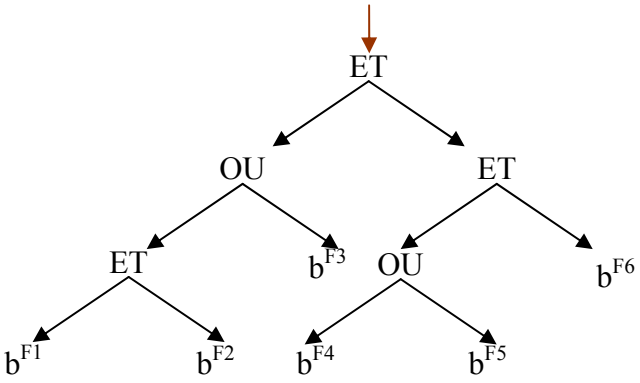


Figure 4.1 : Un exemple d'arbre de buts en conflit

- ***Chemin à parcourir*** : chemin conduisant des feuilles à la racine, en respectant bien les conditions ET/OU dans l'arbre A^F. L'ensemble de ces feuilles constitue une liste de buts en conflit $L^F = \{ b^{Fi} \wedge b^{Fj} \wedge \ldots\}$. Par exemple, dans l'arbre A^F illustré par la figure 4-1, il y a des chemins comme $\{b^{F1} \wedge b^{F2} \wedge b^{F4} \wedge b^{F6}\}$, $\{b^{F3} \wedge b^{F5} \wedge b^{F6}\}$, …

- ***Session*** : entité représentant toutes les interactions déroulées entre un seul utilisateur et le système. Elle peut donc être un dialogue complet ou bien un sous-dialogue.

- ***Dialogue multisession*** : ensemble successif de sous-dialogues discontinus et discrets, chacun représentant une propre session. Cet ensemble peut être :

  - Une session unique entre un utilisateur D et le système : c'est le cas ordinaire dans lequel b_D est directement atteint ou abandonné,
  - Au moins trois sessions : c'est le cas où b_D ne peut pas être directement atteint car il est en conflit avec d'autres utilisateurs – ou bien des patients. Dans ce cas, le dialogue contient une session d'ouverture, au moins une session pour la négociation avec des patients pour résoudre des conflits, et une session pour informer D du résultat.

4.2.2 Quand apparaît le dialogue multisession ?

Selon les définitions précédentes, un dialogue multisession est considéré comme un ensemble de sous-dialogues discontinus dont chacun représente une session entre le système et un utilisateur. En d'autres termes, un dialogue multisession représente un tour fermé de sessions entre le système et plusieurs utilisateurs. Il commence et se termine avec l'utilisateur initial, qui pose un nouveau but (demandeur).

La figure 4.2 illustre le cycle de sessions dans un dialogue multisession. Bien évidemment, il n'y a qu'un seul utilisateur initial et les autres utilisateurs sont des patients engagés dans ce dialogue.

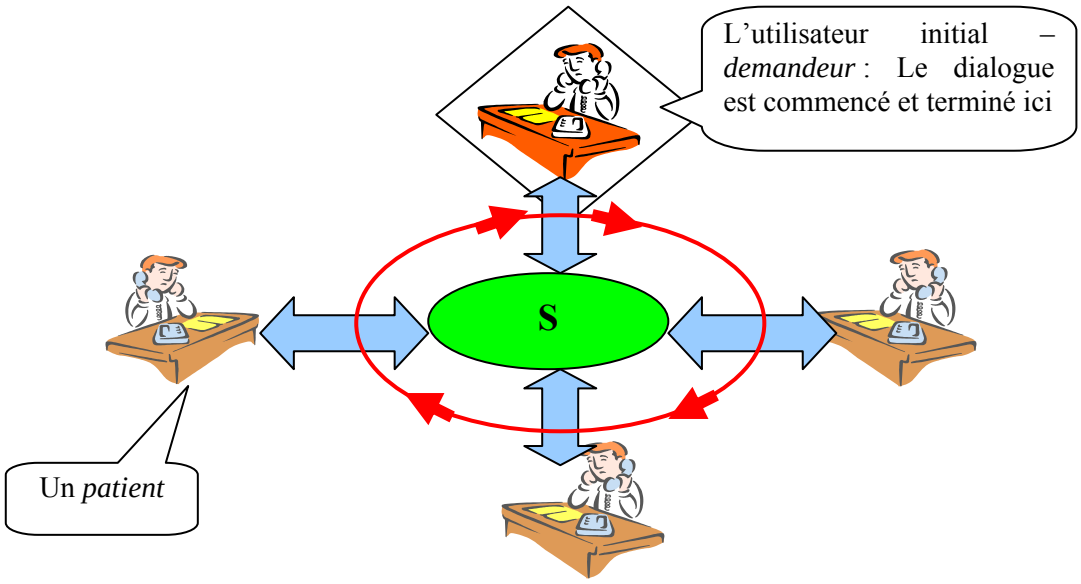


Figure 4.2 : Tour de sessions dans un dialogue multisession

A partir de cela, nous pouvons constater que le dialogue multisession peut apparaître dans les deux cas suivants :

- cas 1 : Le but posé par un utilisateur exige des informations supplémentaires d'autres utilisateurs : par exemple pour diffuser des informations, collecter des réponses… Ces sessions ne nécessitent pas de négociations pour atteindre le but.
- cas 2 : Le but posé par un utilisateur est en conflit avec d'autres buts qui sont déjà satisfaits, posés par d'autres utilisateurs. Dans ce cas, le système doit résoudre ces conflits via les négociations nécessaires.

Il est certain que le second cas est plus difficile à traiter que le premier, qui laisse au demandeur le droit actif pour décider totalement des jugements de son but. Les sessions, qui se déroulent dans le premier cas, ne sont donc que le rassemblement de dialogues ordinaires. En tenant compte de cette cause, nous avons décidé de nous concentrer fortement sur le second cas

avec des conflits qui conduisent à des négociations importantes. Nous abordons exclusivement ce cas dans les sections suivantes.

4.2.3 Enchaînement de sessions

L'enchaînement de sessions dans un dialogue multisession assure à la fois la cohérence et l'avancement du dialogue. Il permet évidemment de bien coordonner toutes les sessions et de préciser la position de chaque session. Comme illustré dans la figure 4-3, nous envisageons tout d'abord la structure d'un dialogue multisession en le divisant en trois phases principales :

- *Emergence* : Au départ, l'utilisateur D manifeste un nouveau but $?b_D$. Cependant, il y a des conflits pour atteindre ce but et le système S doit déterminer ces conflits en construisant l'arbre de buts en conflit A^F. S informe D de ces conflits par la représentation de A^F. Une fois que S reçoit la demande de résoudre ces conflits pour atteindre b_D, il doit mettre b_D en attente et planifier le processus de résolution de conflits en faisant l'exploration de l'arbre A^F,

- *Négociation* : Cette phase constitue un processus de négociation pour atteindre le but b_D. Ce processus comporte éventuellement plusieurs sessions et engage plusieurs utilisateurs. La solution obtenue est un des chemins à parcourir dans A^F. Il est certain que, avant d'obtenir le résultat final, S pourrait subir des échecs illustrés par des impasses au cours de la recherche du chemin final. Tant que le but b_D n'est pas encore atteint et s'il existe encore d'un chemin à parcourir dans A^F, le système doit contacter le patient pour négocier avec lui un compromis favorable à b_D. Ce processus sera terminé quand b_D est atteint (cas positif) ou bien quand il n'y a plus de chemin à suivre dans A^F et qu'on doit abandonner b_D (cas négatif),

- *Notification* : Une fois le résultat final obtenu concernant b_D, S va re-contacter D et l'informer de ce résultat de négociation (soit $\dagger b_D$ en cas positif ou soit $@b_D$ en cas négatif). D peut évidemment accepter ou bien refuser ce résultat mais le dialogue est considéré comme complet dans cette session.

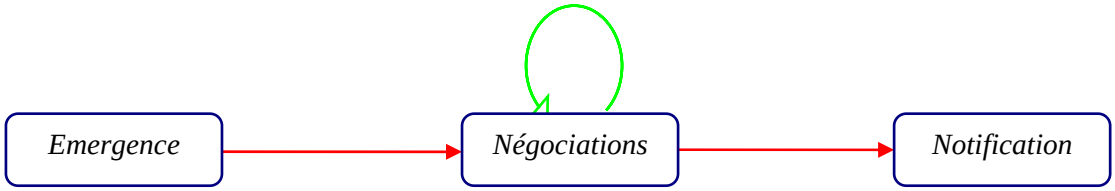


Figure 4.3 : Les trois phases d'un dialogue multisession

L'enchaînement de sessions dans un tel dialogue est illustré par la figure 4-4 en utilisant des notions de modélisation du dialogue, abordées au chapitre 2.

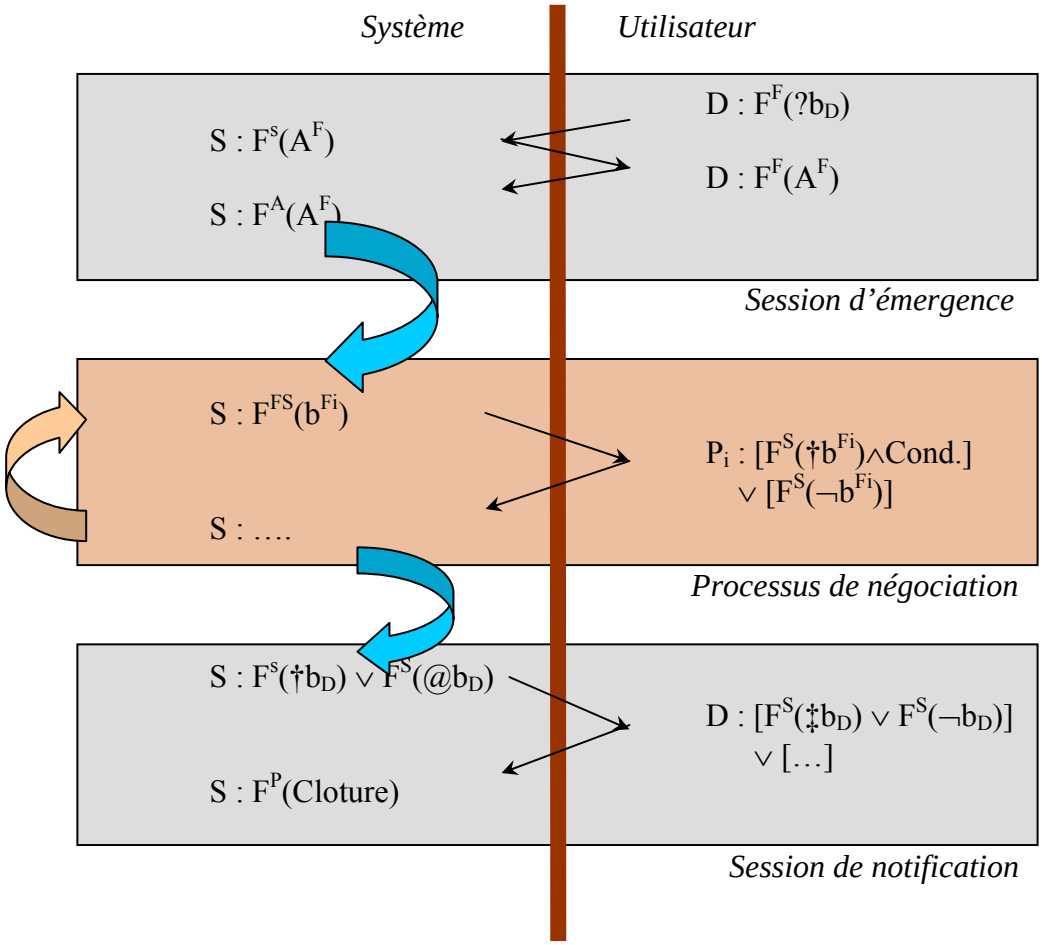


Figure 4.4 : Enchaînement de sessions dans un dialogue

Dans la phase d'émergence, le demandeur D manifeste son but en demandant le but $?b_D$ par l'acte faire-faire F^F. Le système S signale l'existence de conflits en précisant l'arbre de buts en conflit via l'acte faire-savoir $F^S(A^F)$. Pour obtenir son but, D demande donc à S de résoudre les conflits par l'acte $F^F(A^F)$. Selon de certaines conditions (nous préciserons à la section 4.3.1), S effectue la demande de D et lance le processus de négociation, c'est-à-dire la phase de négociation.

En ce qui concerne la phase de négociation, S doit résoudre le conflit en négociant avec un certain nombre de patients. La négociation avec un patient P_i est effectuée tout d'abord en l'informant du but de conflit $F^{FS}(b^{Fi})$ en faveur du demandeur D. P_i peut laisser ce but à D en donnant éventuellement certaines conditions *Cond.*, ou bien réfuter ce but. Le système doit répéter cette négociation jusqu'à atteindre la réponse complète concernant le but b_D du demandeur : b_D soit atteint $\dagger b_D$, soit abandonné $@b_D$, et le processus de négociation est terminé ici.

La phase de notification, représentée par une seule session, concerne le fait que S informe D du résultat de négociation par soit l'acte $F^s(\dagger b_D)$, soit $F^S(@b_D)$. D peut accepter ce résultat, le refuser, ou bien demander d'autres choses. Cependant, le dialogue multisession est considéré comme complet dans cette session.

4.2.4 Multisession et négociation sociale

La négociation est une activité sociale qui attire l'attention de beaucoup de philosophes, de psychologues, d'économistes, de théoriciens sociaux, et maintenant de chercheurs dans le domaine du dialogue. Dans « Sociologie de la négociation » [Bourque et Thuderoz, 2002], cette activité est considérée comme « *un processus social, cheminant d'étape en étape, à la fois symétrique et indéterminé pour résoudre des conflits ou des querelles. […] Il s'agit donc d'un processus de prise de décision entre des parties interdépendantes mais dont les intérêts sont différents ou divergents.* »

En effet, il est certain que le conflit est la poursuite de la négociation par d'autres moyens [Adam et Reynaud, 1978]. Normalement, un intérêt conjoint à régler le conflit des personnes engagées ouvre la voie à un compromis, qui représente une solution commune mutuellement acceptable pour ces personnes.

Fondé sur ces principes, un dialogue multisession est véritablement une forme de négociation, dans laquelle le système cherche à obtenir un compromis entre les utilisateurs (patients) concernant le conflit posé dans un contexte du service visé par le système de dialogue. Certes, notre approche concernant le dialogue multisession ne modélise qu'une partie de la négociation dans laquelle le conflit se limite à un utilisateur et plusieurs patients. Cependant, avec des nouvelles recherches, la négociation générale et sociale sera intégrée au système de dialogue.

4.2.5 Exemple

Pour illustrer l'enchaînement d'un dialogue multisession, le service d'organisation de réunion est utilisé comme cas d'études pour notre approche.

Supposons que M. Dupuis veuille absolument réserver la salle Lafayette pour le 17 mai 2004, de 10h à midi et qu'il téléphone au système S. Malheureusement, cette salle est déjà réservée avant par M. Dupont de 9h à 11h et par M. Dupaint de 11h à 13h. Avec l'ambition de satisfaire sa demande, S cherche dans sa base de connaissances et trouve que la salle Aquarium est disponible au temps demandé. S signale ensuite tous ces phénomènes à M. Dupuis. En ayant bien conscience que la salle Aquarium est trop petite par rapport au nombre de participants de sa réunion, qu'il a un droit supérieur (la priorité, causée par la hiérarchie de poste) à celui des deux autres utilisateurs, M. Dupuis décide donc d'ordonner au système de négocier avec les deux patients pour lui laisser la salle Lafayette. Le déroulement de ce dialogue, illustré par la figure 4.5, comporte quatre sessions différentes et séparées. Une session initiale fait apparaître les conflits à négocier, deux sessions sont dans le processus de négociation avec ces deux patients et la dernière session notifie le résultat à M. Dupuis. L'évolution des buts au cours de négociation est également montrée dans cette figure.

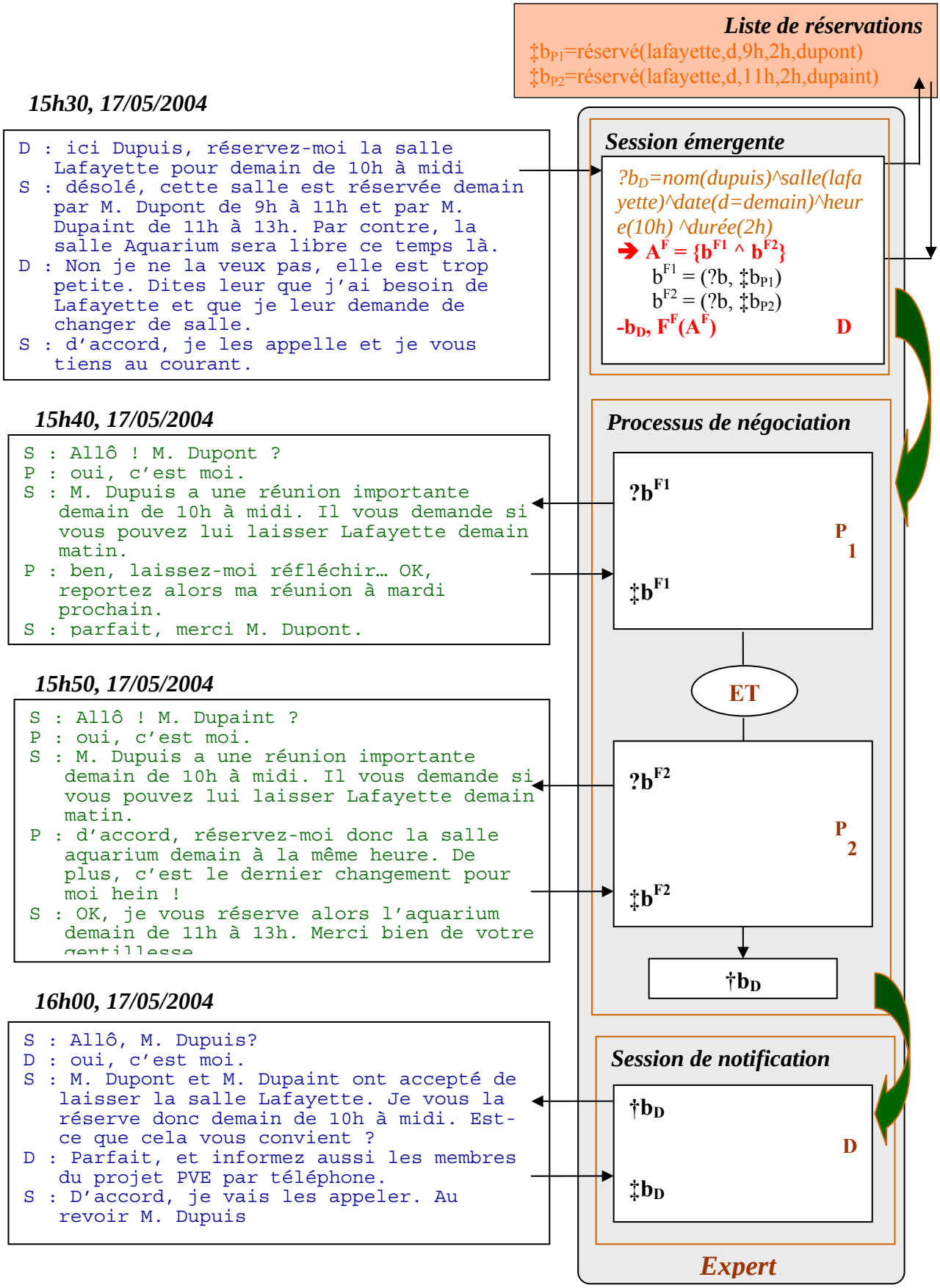


Figure 4.5 : Enchaînement d'un dialogue multisession

Cet exemple illustre parfaitement le problème soulevé dans le cas d'un dialogue qui a lieu en plusieurs étapes. Notre modélisation, avec la définition d'un dialogue multisession, s'avère parfaitement adapté à ce type de situation. Elle est effectivement la base solide à développer la gestion de dialogue multisession présentée dans la section suivante.

4.3 Gestion de dialogue multisession

Dans l'optique d'apporter la capacité de traitement de dialogue multisession, le système de DHM doit être équipé d'une gestion cohérente, dynamique et efficace d'un dialogue multisession. Il est évident que la demande cohérente permet d'assurer le déroulement correct du dialogue multisession ; l'exigence dynamique permet au système de s'adapter à des situations différentes ; et l'efficacité doit être considérée afin d'obtenir rapidement la résolution de problèmes posés autour du dialogue multisession. Les trois critères « cohérence, dynamique et efficacité » sont donc notre objectif, pour apporter de la souplesse au système de dialogue.

En précisant bien le rôle de chaque module dans l'architecture d'un système DHM, nous pouvons constater que la gestion de dialogue multisession devra être effectuée à la fois au niveau du contrôleur du dialogue (CD) et à celui du contrôleur de la tâche (CT). Lorsque l'utilisateur D demande le but $?b_D$, le CT doit déterminer s'il existe des conflits avec d'autres utilisateurs. S'il y en a, il construit immédiatement l'arbre de buts en conflits A^F et le transforme directement au CD.

Au niveau du CD, il doit assurer la gestion de l'arbre de buts en conflit A^F, ainsi que l'enchaînement de sessions. Nous élaborons un module appelé ***Expert*** qui prend le rôle de jugement des attitudes portant sur l'arbre A^F. Une fois que la demande effective de D pour débloquer les conflits est reçue, l'Expert planifie le processus de négociation avec des patients. Il doit également s'assurer de trouver un bon patient avec lequel négocier, afin d'obtenir rapidement le résultat souhaité.

Le détail des tâches et le fonctionnement de chaque module sont décrits dans les sections suivantes.

4.3.1 Au niveau du contrôleur du dialogue

Dans le dialogue multisession - en plus du rôle de calculer la stratégie adéquate, de déterminer le but de dialogue, de générer l'acte du système…, le CD doit également trouver une bonne stratégie d'exploration dans l'arbre A^F pour négocier activement avec des patients, afin d'atteindre rapidement et efficacement le but demandé par l'utilisateur. L'enchaînement des sessions, ainsi que ses cohérences, doivent être traitées à ce niveau. Différentes attitudes de l'utilisateur devant les conflits sont également envisagées, pour mieux gérer le dialogue.

4.3.1.1 Attitudes de l'utilisateur devant l'arbre de buts en conflits

Au cours de la session d'émergence, une fois que le CD découvre des conflits par l'aide du CT, il notifie ces conflits à l'utilisateur. Ses attitudes devant cet arbre sont :

- $\mathbf{F^{A}_{D}(b_D) => F^{S}_{M}(A^F)}$: S informe les conflits à D,
- $\mathbf{F^{FS}_{D}(A^F) => \delta=directive \wedge F^{S}_{M}(explications\ A^F)}$: S explique à D le phénomène de A^F,
- $\mathbf{F^{F}_{D}(A^F) => -b_D \wedge \delta=négocié \wedge F^{A}_{M}(A^F)}$: S met b_D dans la pile d'attente (en suspendant le dialogue entre S et D), applique la stratégie négociée et commence le processus de négociation.
- $\mathbf{F^{F}_{D}(\neg A^F) => \delta=coopérative \wedge F^{FS}(b'_D)}$: Si D manifeste son désaccord de négociation pour débloquer les conflits, alors S va lui proposer une autre solution représentée par le but réparé b'_D en appliquant la stratégie coopérative.

Dans cette section, nous ne nous intéressons qu'au cas dans lequel D demande à S de négocier avec les patients en souhaitant obtenir son but posé, et qui amène l'apparition du processus de négociation (les autres attitudes sont traitées comme dans un dialogue normal au CD). Il est certain que le système doit bien examiner son droit vis-à-vis des autres patients afin d'éviter des demandes excédant ses pouvoirs. Dans le cas où D a le droit d'ordonner des négociations, S suspendra le dialogue avec D, planifiera et déclenchera le processus de négociation. Dans le cas contraire, S doit proposer une autre solution à D en signalant le problème de droits.

*4.3.1.2 « **Expert** » et gestion de l'arbre de buts en conflits*

Tous les buts en conflits qui sont déjà déterminés par le CD doivent être gérés au CD. Nous introduisons un nouveau module nommé **Expert** dans l'optique de mener à bien la gestion de l'arbre A^F, ainsi que le processus de négociation. Son fonctionnement a pour objectif de :

e1 : Trouver le meilleur chemin non-parcouru à parcourir dans A^F, en bien respectant ses conditions ET/OU $L^F = \{b^{F1} \wedge b^{F2} \wedge \ldots \wedge b^{Fn}\}$. Les critères de recherche de L^F dépendent quasiment du contexte du service que le système vise. Par exemple pour le service d'organisation de réunion, pour économiser le coût d'appel, on peut utiliser un critère comme chercher le chemin non-parcouru le plus court, …

e2 : Planifier le temps à négocier avec chaque patient dans L^F. A cette étape, il doit juger quel patient S doit contacter auparavant. Pour répondre à ces exigences, il utilise stratégiquement des critères comme :

- la potentialité conduisant à une réponse négative,
- la disponibilité de ces patients,
- le choix du temps de négociation le plus efficace.

Il est évident que l'échec de négociation d'un patient dans L^F amène directement l'échec de ce chemin. Dans ce cas, ***Expert*** doit marquer ce chemin comme parcouru et revenir à l'étape *e1*.

L'algorithme de traitement de conflits dans Expert est représenté dans la figure 4.6.

```
TantQue (b_D != †b_D) Alors
    L^F = chercheChemin(A^F);// chercher un chemin non-parcouru dans A^F
    Si (L^F == vide) Alors
        Notifier(D,@b_D) ;// Informer D que son but ne peut pas être atteint @b_D
        sortir ;
    FinSi
    FlagNegociation = OUI;
    Pour chaque b^Fi dans L^F Alors
        Négocier(P_i, b^Fi) ; // créer une session « négocier avec P_i » concernant le
                but en conflits b^Fi
        MAJ_Etat(A^F) ;//mettre à jour A^F avec le nouveau état de b^Fi
        Si (b^Fi != ‡b^Fi) Alors
            FlagNegociation = NON ;
            break ;// terminer le processus de négociation dans le chemin L^F
                    sans aucun résultat obtenu.
        FinSi
    FinPour
    Si (FlagNegociation == OUI) Alors
        Notifier(D, †b_D) ; //informer D que son but est déjà atteint et le dialogue
                se déroule comme un dialogue normal.
        sortir ;
    FinSi
FinTantQue
```

Figure 4.6 : Algorithme de traitement de conflits

Pour chercher le chemin le plus court, on adapte l'algorithme de Dijkstra [Cormen et al., 2002] avec le principe suivant : le coût du chemin est le nombre de feuilles dans l'arbre A^F. L'algorithme de recherche du chemin le plus court à parcourir dans l'arbre A^F est implémenté comme décrit dans la figure 4.7.

```
    LePlusCourtChemin(in A^F, out L^F)
        //on commence par le nœud racine
        n = racine(A^F);
        //Condition d'arrêt : le nœud courant est bloqué ou bien est une feuille
        Si (n.drapeau == BLOQUE) Alors
            L^F = vide ;
            Terminer() ;
        FinSi ;
        Si (n == feuille) Alors
            L^F = {n} ;
        SiNon
        //on cherche les chemins de chaque sous-arbre dans ce nœud
            LePlusCourtChemin (n.gauche, L1) ;
            LePlusCourtChemin (n.droit, L2) ;
        //Si le contraint est ET, on doit rendre le chemin combiné tous les deux sous-arbres
            Si (n.opérateur == ET) Alors
                L^F = {L1,L2} ;
            //Sinon, on rend le chemin le plus court
            SiNon (n.opérateur == OR) Alors
```

```
                    Si (L1 == Vide) Et (L2 == Vide) Alors
                        L^F = vide ;
                    SiNon (L1 == Vide) Alors
                        L^F = L2 ;
                    SiNon (L2 == Vide) Alors
                        L^F = L1 ;
                    SiNon (L1.nombre > L2.nombre) Alors
                        L^F = L1 ;
                    SiNon
                        L^F = L2 ;
                    FinSi ;
                FinSi ;
            FinSi ;
        Fin_LePlusCourtChemin ;
```

Figure 4.7 : Algorithme pour trouver le chemin le plus court

Au cours de l'exploration de l'arbre de buts en conflit A^F, l'état de chaque feuille/nœud est contrôlé par l'Expert pour qu'il puisse déterminer quel est le chemin non-parcouru qui reste à suivre. Nous avons donc utilisé une étiquette pour chaque nœud/feuille dans A^F avec le mécanisme d'attribution comme suit : une feuille, représentant le but en conflit b^{fi} entre D et P^i, sera marquée comme « BLOQUER » si et seulement si la négociation avec P_i a échoué ; par contre, un nœud N est marqué comme « BLOQUER » si :

- soit N représente la condition ET, et un de ses deux enfants est marqué comme BLOQUER,
- soit N représente la condition OU, et tous ses deux enfants sont marqués comme BLOQUER.

Une fois que l'on obtient la liste de buts en conflit L^F dans un chemin non-parcouru de A^F, l'Expert doit appliquer des stratégies actives pour bien choisir un patient dans cette liste pour commencer à négocier avec lui. A l'étape i pour traiter le patient P_i, si P_i réfute le but en conflit b^{fi}, on doit abandonner ce chemin et marquer les drapeaux concernant ce but comme BLOQUER. Ensuite, on répète le processus de négociation en trouvant un autre chemin non-parcouru le plus court…

Dans le cas où le système arrive à rassembler tous les résultats de négociation positifs avec tous les patients de L^F (le but posé par D est déjà atteint), l'Expert va déclencher la session de notification à D. Le dialogue se déroule alors comme un dialogue normal.

En ce qui concerne le cas négatif (il s'agit que le but b_D ne puisse pas être atteint), l'Expert doit éventuellement garder le meilleur chemin qui rend le meilleur compromis entre b_D et les buts des patients dans ce chemin. Dans ce cas là, au cours de la session de notification, le CD doit proposer à D une autre solution attachée à ce compromis.

4.3.1.3 Stratégie de négociation

Afin de gérer de manière souple et efficace l'arbre de buts en conflit, nous proposons les aspects suivants, en envisageant des connaissances supplémentaires associées à chaque patient au cours du processus de négociation :

- *Ordre de négociation* : Dans l'arbre de buts en conflit, chaque patient prend un rôle différent dans le système et donc dans le processus de négociation. De plus, la relation entre les patients dans A^F est également un élément important qui pourrait conduire à des contraintes entre l'un et l'autre. Cependant, nous introduisons la notion d'« ordre de négociation », qui est affectée à chaque patient dans A^F. Cet ordre est représenté par la séquence successive de patients qui doivent négocier avant de lancer la négociation avec lui-même.

- *Coût de négociation* : Dans l'arbre de buts en conflit, il est certain que la négociation avec un patient crée et subit également des influences sur d'autres utilisateurs. A titre d'exemple, pour le service d'organisation de réunion, la résolution du conflit d'une salle pourrait conduire à un nouveau conflit pour une autre salle ou du matériel. C'est pourquoi nous définissons la notion de « coût de négociation », qui calcule l'ensemble des éléments influençant le résultat final du système si le système réalise la négociation, pour obtenir le but b_D.

Nous calculons le coût négociation C^N_{Pi} pour un patient P_i par la formule suivante :

$$C^N_{Pi} = \text{Gain}(P_i) + \sum_j \text{Gain\_conjoint}(P_i, U_j) - \text{Concession}(P_i) - \sum_j \text{Concession\_conjointe}(P_i, U_j)$$

où :

- Gain(P_i) est le gain que le système gagne directement s'il négocie avec P_i concernant le conflit entre b_D et ‡b_{Pi},
- Σ_jGain_conjoint(P_i, U_j) est l'ensemble de gains que le système obtient grâce à l'influence de P_i sur les autres utilisateurs dans le système,
- Concession(P_i) est la concession que le système perd s'il négocie avec P_i,
- Σ_jConcession_conjointe(P_i, U_j) est l'ensemble des concessions que le système doit subir à cause des influences des autres utilisateurs à P_i par rapport au résultat de négociation avec P_i.

La stratégie de négociation est donc de trouver le chemin non-parcouru ayant le coût total de négociation maximal. Dans ce chemin, selon l'ordre de négociation de chaque patient, l'Expert déterminera avec qui il doit négocier d'abord.

4.3.1.4 Règles d'initialisation de session de négociation

Au cours du processus de négociation, il est évident que le résultat de chaque session décide forcément du résultat final de négociation. Le système S doit donc être équipé des bonnes stratégies de dialogue, pour apporter la souplesse nécessaire à chaque session différente de négociation avec un patient différent, dès le début jusqu'à la fin de session. En effet, l'ouverture de négociation est toujours un élément important qui pourrait promouvoir la négociation dans des cas positifs ou bien affaiblir, même détruire, le processus de négociation dans des cas inverses. Par cette raison, les comportements du système vis-à-vis d'un patient devraient être bien codés dans chaque énoncé du système. Le choix d'une stratégie adéquate, dès l'ouverture de session et dans tous les échanges entre S et ce patient, est donc décisif.

La stratégie de dialogue appliquée au début de chaque session sera donc choisie de manière pragmatique. Nous pouvons constater les éléments qui influencent éventuellement la négociation comme le rôle du patient (son rôle social, son influence sur d'autres patients dans le processus…), la nécessité d'avoir rapidement la réponse, etc. Notre approche considère les attitudes concernant la stratégie de dialogue utilisée pour initier chaque session S_i dans le processus de négociation :

- rôle(D) > rôle(P_i) ➔ δ = **directive** : on applique la stratégie directive si le rôle du demandeur D est plus grand que le patient P_i. Dans ce cas, S annonce seulement le problème et l'origine du conflit, sans donner une solution alternative.
- rôle(D) <= rôle(P_i) ➔ δ = **coopérative** : la stratégie coopérative est utilisée dans le cas où la priorité de P_i est plus haute ou égale à celle de D. Dans ce cas, S doit négocier avec P_i de manière plus souple en lui proposant une autre solution attractive à remplacer.

Les autres stratégies sont évidemment évitées lors du début d'une session en raison d'incompatibilité. Au cours de chaque session, la stratégie est donc calculée dynamiquement selon chaque tour de parole entre le patient et le système.

4.3.1.5 Interaction entre Expert et contrôleur de la tâche

Le module Expert est stratégiquement en charge de gérer l'arbre de buts en conflits A^F afin de trouver le résultat de b_D qui est soit abandonné ($@b_D$), soit atteint ($\dagger b_D$). C'est pourquoi le moment de déclenchement de chaque session dans le processus de négociation et la session de notification doivent être déterminés par l'interaction avec le contrôleur de la tâche.

4.3.2 Au niveau du contrôleur de la tâche

Le contrôleur de la tâche gère bien évidemment des tâches dans le service qu'il vise à réaliser. De plus, pour un dialogue multisession, il doit également identifier les conflits s'il y en a et construire l'arbre binaire ET/OU de buts en conflits A^F pour que l'Expert puisse traiter ces conflits. Une fois que l'Expert trouve un patient avec qui négocier, le CT doit être invoqué par l'Expert pour déterminer le moment du déclenchement de la session concernée.

4.3.2.1 Identification de conflits

Au niveau du CT, un but de dialogue b_D est interprété par un ensemble de buts élémentaires indivisibles. Les conflits seront donc déterminés par ces buts élémentaires. Par exemple, pour le dialogue de réservation de salle, un but de dialogue b_D est décomposé comme b_D = salle(s) $\wedge$ date (d) $\wedge$ heure(h) $\wedge$ durée(dur) $\wedge$ matériel(mats) $\wedge$ membre(participants) $\wedge$ mode(m). Les conflits possibles dans cette tâche concernent donc le temps de réservation (date, heure et durée) et les matériels dans une même salle.

Dans l'agenda géré par le CT, il garde soigneusement tous les buts déjà satisfaits posés par les utilisateurs au système. Une fois qu'un utilisateur D manifeste un nouveau but b_D, le CT va consulter son agenda pour déterminer des conflits possibles. Par exemple, dans l'agenda, on a trois buts déjà satisfaits proposés par P_1, P_2 et P_3 tels que :

$\ddagger b_{P1}$ = salle(S) $\wedge$ date(10/05/2004) $\wedge$ heure(9h) $\wedge$ durée(2h) $\wedge$ membre(PVE)

$\ddagger b_{P2}$ = salle(S) $\wedge$ date(10/05/2004) $\wedge$ heure(11h) $\wedge$ durée(2h) $\wedge$ membre(GEOD)

$\ddagger b_{P3}$ = salle(R) $\wedge$ date(10/05/2004) $\wedge$ heure(8h) $\wedge$ durée(4h) $\wedge$ matériel(projecteur)

Maintenant, si D demande un but comme $?b_D$ = salle(S$\vee$R) $\wedge$ date(10/05/2004) $\wedge$ heure(10h) $\wedge$ durée(2h), il est clair que les conflits possibles concernent le problème de créneau horaire pour la salle demandée par D. Le CT se rend compte de ces conflits et détermine que b_D est en conflit avec trois autres buts $\ddagger b_{P1}$, $\ddagger b_{P2}$ et $\ddagger b_{P3}$. Finalement, il identifie ces trois conflits comme $b^{F1} = (b_D, \ddagger b_{P1})$, $b^{F2} = (b_D, \ddagger b_{P2})$ et $b^{F3} = (b_D, \ddagger b_{P3})$

4.3.2.2 Construction de l'arbre A^F

Lorsque le CT arrive à identifier des buts en conflits, il doit également les coordonner et construire l'arbre A^F de manière correcte, cohérente et facile à exploiter. Il s'agit que le CT applique l'opérateur binaire ET/OU aux buts en conflits. Par exemple, dans le cas précédent, le créneau horaire demandé par D pour la salle S est en conflit avec deux patients P_1 et P_2. L'opérateur ET doit être utilisé pour ces deux buts en conflits b^{F1} et b^{F2}. Par contre, en ce qui concerne la salle R, il n'y a qu'un conflit entre D et P3. C'est pourquoi l'arbre de buts en conflit A^F peut être représenté comme une expression logique $(b^{F1} \wedge b^{F2}) \vee (b^{F3})$. Le CT se base sur cette expression pour constituer A^F, comme décrit dans la figure 4-8.

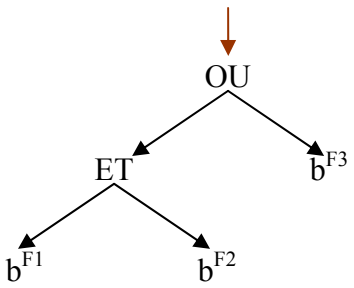


Figure 4.8 : Construction d'un arbre A^F

Une fois que le CT possède l'arbre de buts en conflit, il le transfère ensuite au CD pour informer l'utilisateur de ces conflits.

4.3.2.3 Queue de sessions

Pour l'instant, nous supposons que le système n'assure au mieux qu'une interaction entre lui-même et un utilisateur à chaque instant. Dans le processus de négociation et la session de notification, le système doit prendre le rôle actif : il s'agit d'initier chaque session dans le chemin à parcourir L^F, déterminé par le module Expert. Cette tâche est pratiquement assurée dans le module CT, en se basant sur une queue de sessions qui précise à quel moment le système doit lancer telle ou telle session.

Le moment de lancer une session est déterminé selon la base de données dans le modèle de la tâche. Lorsque l'Expert trouve le chemin le plus court à parcourir dans A^F, il interagit directement avec CT pour qu'il puisse construire la queue de sessions. Bien évidemment, une fois que ce chemin est abandonné, le CT doit impérativement supprimer cette queue et attendre un autre chemin proposé par l'Expert. Quand le moment de lancer une session concernant l'utilisateur U arrive, le CT déclenche immédiatement la nouvelle session de dialogue avec U. Eventuellement, si U n'est pas encore joignable, le CT doit reporter ce moment pour le re-contacter plus tard.

4.4 Conclusion

L'ambition d'exploiter de plus en plus la machine, afin de remplacer l'humain, sera attachée à sa capacité de traiter plusieurs tâches. Nous considérons donc que l'idée du dialogue multisession est un des éléments décisifs pour atteindre cet objectif.

En tenant compte de l'analyse d'usage des problèmes concrets, nous avons présenté dans ce chapitre un nouveau concept concernant la multisession dans un dialogue. Un dialogue multisession se compose de trois phases : une session émergente, un processus de négociation, et une session de notification. Avec les connaissances acquises et enrichies portant sur le modèle stratégique, nous avons modélisé un dialogue multisession en utilisant ses éléments principaux : actes, stratégies et buts de dialogue. En proposant la notion « **Expert** », nous avons évoqué des problèmes concernant la gestion de dialogue multisession, à la fois au niveau du contrôleur du dialogue et de la tâche. De plus, nous avons également précisé la stratégie de négociation et les règles d'initiations d'une session de négociation, en vue de conforter la souplesse au cours du processus de négociation.

Le dialogue multisession ouvre effectivement la voie de projeter la négociation entre l'utilisateur et le système dans le domaine du DHM, sous l'angle de la résolution de conflits de ressources entre deux ou plusieurs personnes. En effet, le dialogue multisession permet au système d'intervenir éventuellement à plusieurs utilisateurs, de manière discontinue, dans le but de trouver

un compromis favorable entre eux. Par conséquent, le dialogue multisession apporte de la souplesse à un dialogue homme-machine et renforce l'efficacité d'un système de DHM.

Bien que l'approche de dialogue multisession nous donne des telles privilèges, l'implémentation de cette approche dans un système de DHM demande effectivement des travaux considérables. L'expérimentation, décrite dans le chapitre 5 et dédiée au système de dialogue Mélina, constitue un cas typique pour valider notre approche.

Chapitre 5

EXPERIMENTATION

Dans ce chapitre, nous présentons les expérimentations réalisées au cours de cette thèse. Tout d'abord, nous allons parler du système de dialogue oral ***Mélina*** qui est construit comme un agent conversationnel dédié au service d'organisation de réunion. Ensuite, nous allons montrer des exemples enregistrés au cours des expérimentations que nous avons faites. Nous présenterons, finalement, un bilan pour ces expérimentations.

5.1 Introduction

En appliquant les connaissances acquises durant les chapitres précédents, nous avons réalisé des expérimentations ayant pour objectif de valider nos théories. Nous avons effectivement développé un système de dialogue oral homme-machine en prenant le service d'organisation de réunion comme cas d'application. Ce système est nommé **Mélina :** il a été utilisé également dans le cadre du projet de recherche PVE.

Mélina est conçu comme un agent conversationnel, qui a pour but de remplacer une secrétaire dans la tâche d'organisation de réunion. L'utilisateur utilise le téléphone pour dialoguer avec le système afin de réserver une salle, modifier une réservation, informer les participants… De plus, dans le cas causé par des conflits de ressources (par exemple, de la salle, du matériel…), il pourrait demander au système de négocier avec plusieurs personnes pour atteindre son but.

Les objectifs principaux que nous sommes fixés, pour la conception et le développement de Mélina, sont les suivants :

- Le système doit se comporter comme un vrai assistant dans le service d'organisation de réunion. C'est à dire qu'il doit être suffisamment coopératif et capable de négocier avec les utilisateurs via des dialogues souples,

- Le système doit satisfaire un certain nombre de conditions : il doit être efficace, robuste et extensible. L'efficacité est la première condition pour que l'utilisateur puisse atteindre rapidement son but. La robustesse a pour objectif d'assurer que le système n'est pas très sensible aux erreurs produites au cours des interactions (erreurs de reconnaissance de la parole, de compréhension sémantique, d'interprétation pragmatique,…). Quant à l'extensibilité, elle consiste en la réutilisation des composants du système en l'adaptant à de nouveaux services.

Mélina a été conçu et développé dans le cadre du projet PVE. Il ne vise actuellement que l'interface vocale par téléphone. Cependant, son extensibilité doit permettre d'y intégrer d'autres modalités comme le clavier (entrée textuelle), la vidéo (entrée/sortie avec des émotions…)

Nous allons présenter l'infrastructure de Mélina qui prend en compte les objectifs précédents pour la conception et le développement. Cette infrastructure est dédiée au système Mélina tout en tant générique, en raison de l'indépendance du contexte. Puis, nous allons présenter le corpus de dialogue qui a permis d'extraire les données nécessaires ayant alimenté tous les modules dans notre architecture. Finalement, nous terminons par la mise en œuvre et les résultats avant le résumé qui conclura ce chapitre.

5.2 Infrastructure de Mélina

5.2.1 Aperçu et principes d'architecture

5.2.1.1 Principes généraux

L'architecture de Mélina est fondée sur celle présentée à la section 1.4.2 du chapitre 1. Ce qui caractérise cette architecture est la séparation des composants d'un système de DHM, afin que nous puissions les manipuler aisément indépendamment les uns des autres. En d'autres termes, l'indépendance est notre but ultime, surtout celle entre le contrôleur du dialogue et le contrôleur de la tâche, afin de construire une infrastructure générique dédiée à la construction du système de dialogue.

Pour atteindre notre but, nous visons les caractéristiques principales correspondant aux défis présentés dans la section 1.4.3. Ces dernières sont : un système générique, extensible, adaptatif, coopératif, distribué et modulaire. La généricité, l'extensibilité et l'adaptabilité sont déjà abordées dans ladite partie. Quant aux principes de distribution et de modularité, ils sont envisagés d'un point de vue de conception logicielle. En effet, chaque composant doit être modulaire, pour éviter tout problème éventuel lié aux modifications des autres composants. Pour une raison similaire, ils sont distribués en tirant profit des capacités et des puissances des différentes machines concernées. Toutes ces caractéristiques facilitent à la fois la conception et la construction d'un système de dialogue, sous l'angle non seulement du domaine de dialogue oral homme-machine, mais aussi du domaine de génie logiciel.

5.2.1.2 *Architecture détaillée*

Après avoir précisé les principes présentés d'un système de dialogue, nous allons aborder plus en détail l'architecture de Mélina, illustrée par la figure 5.1.

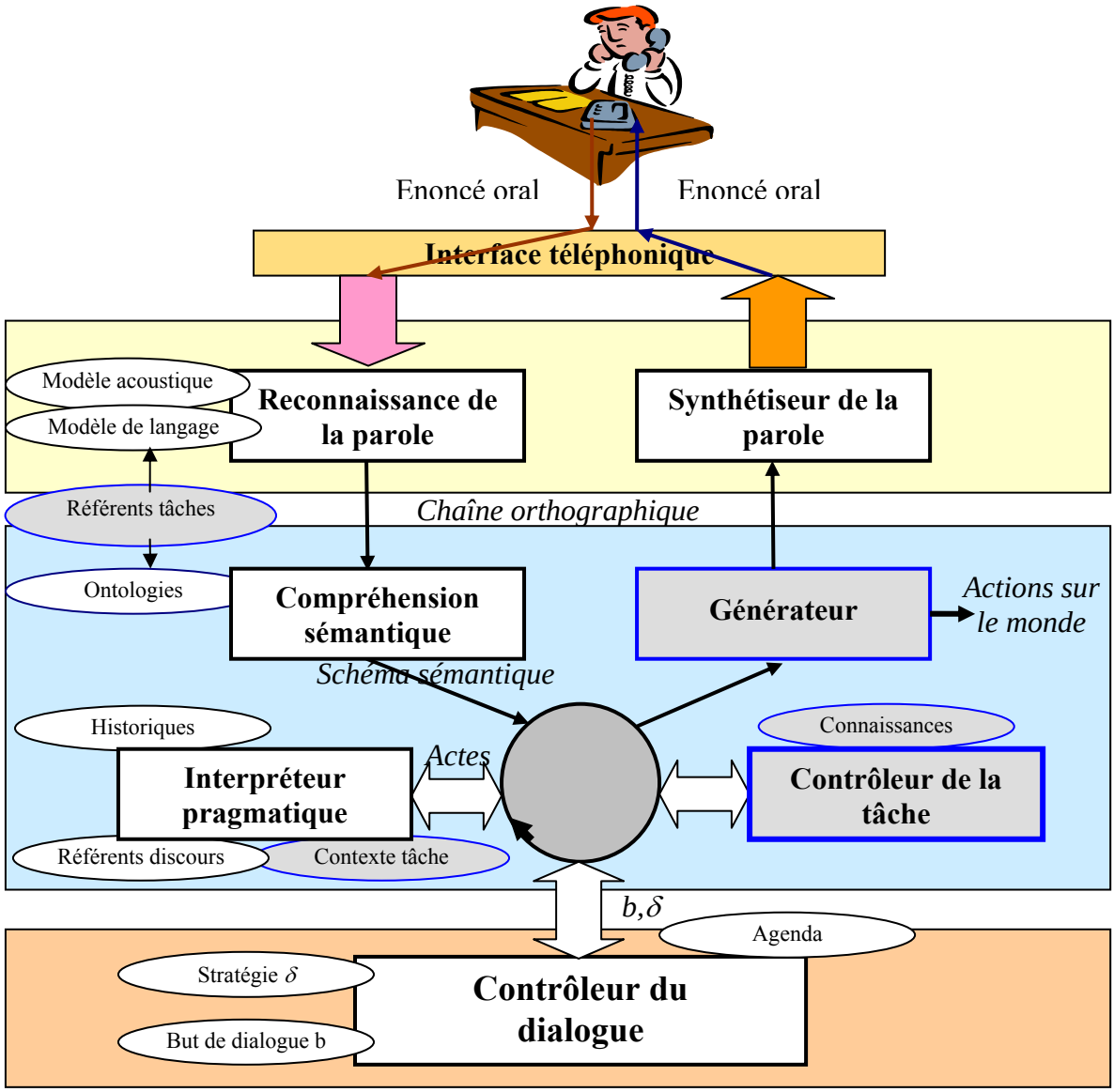


Figure 5.1 : Architecture du système Mélina

L'interaction entre l'utilisateur et le système a lieu via le téléphone. C'est pour cette raison que notre architecture a besoin d'une interface téléphonique, qui prend en charge la capture des appels téléphoniques reçus et leur diffusion. Cette interface combinée avec les réseaux téléphoniques facilite l'utilisateur, pour interagir avec le système de manière permanente. Avec le téléphone, elle constitue donc « la bouche » (qui sait parler) et « l'oreille » du système Mélina.

Notre architecture sépare ses modules de manière générique et instanciée, afin d'obtenir une architecture générique pour un système de dialogue oral homme-machine. Ainsi, au niveau de la reconnaissance de la parole, nous distinguons les données générales des données particulières de la tâche (vocabulaire spécialisé, noms propres par exemple). De même, au niveau du traitement du langage, nous distinguons le module de compréhension sémantique et celui d'interprétation pragmatique. Enfin, au niveau du dialogue, nous séparons ce qui relève de la gestion proprement dite du dialogue (tours de parole, buts dialogiques, stratégies) de ce qui relève de la gestion de la tâche (agenda, objets de la tâche, contexte d'interaction).

Cette architecture est donc divisée en trois couches principales, qui varient entre le plus haut niveau représentant le moyen de communication et le plus bas niveau orienté vers le traitement du discours.

En effet, la première couche contient deux modules : le module de reconnaissance automatique de la parole spontanée, qui correspond métaphoniquement au « *système auditif* » chez l'être humain et le synthétiseur de la parole, qui effectue la tâche des « *cordes vocales* » humaines.

La deuxième couche regroupe quatre modules qui dépendent actuellement encore du contexte du système. Le rôle du module de compréhension est notamment de traduire ce qui est dit dans un format compréhensible par l'interpréteur. Quant à l'interpréteur, il doit décoder, résoudre et raisonner en fonction du sens du résultat sorti du module de compréhension. Le contrôleur de la tâche s'occupe de la gestion des actions dans le monde réel, contenues dans le service visé par le système. Le générateur assure la conversion des signes, extraits du contrôleur du dialogue, en une chaîne de caractères, pour que le synthétiseur puisse les prononcer. Le générateur doit également gérer les « *bras et pieds* » pour effectuer l'action visée par le système, une fois reçue de l'ordre d'exécution provenant du contrôleur du dialogue.

La dernière couche, dédiée particulièrement au contrôleur du dialogue, doit contrôler toutes les activités du système : trouver la bonne stratégie de communication, coordonner les activités entre l'utilisateur et le système, générer les signes du système, pour que le générateur puisse les interpréter en actions sur le monde.

Dans cette architecture, nous définissons donc le flux de données de façon non linéaire comme suit :

- Le locuteur prononce un énoncé oral.
- Le système de reconnaissance produit une chaîne orthographique candidate.
- La chaîne orthographique est ensuite prise en charge par un composant linguistique (dit de compréhension) qui fournit un schéma sémantique représentant le sens littéral de l'énoncé.
- On entre ensuite dans un processus itératif contrôlé par un mécanisme de gestion de dialogue. Le schéma sémantique s'enrichit au fur et à mesure des apports des autres modules pour devenir un acte de langage, en particulier grâce au module d'interprétation et au module

de gestion de la tâche. Cet acte de langage est pris en charge par le contrôleur du dialogue qui le soumet sous forme de but au gestionnaire de tâches.

- La réponse de ce dernier conduit à formuler une réaction sous forme de changement dans le « *monde de la tâche* » et/ou de réponse vocale synthétisée. En cas d'échec dans la résolution d'une tâche, le contrôle est redonné au contrôleur du dialogue qui décide de la stratégie à tenir. Il peut alors redistribuer les rôles et les traitements à l'un des modules.

5.2.2 Infrastructure générique

Dans cette section, nous n'abordons que les primitives qui constituent l'infrastructure générique permettant de construire un système de dialogue en visant deux objectifs : modularité et distribution. Il faut souligner également que les actions sur le monde (celles qui sont rattachées aux tâches précises du système) ne nous intéressent pas ici. En effet, les approches innovantes dans le domaine du génie logiciel, de la gestion de bases de données, etc., peuvent être ignorées dans notre proposition, du fait de la faible quantité de données d'échanges entre les composants du système.

Nous présentons maintenant notre approche constituée de l'infrastructure de Mélina en particulier, et d'un système de dialogue en général.

5.2.2.1 Modules distribués

Nous caractérisons un module dans l'architecture précédente en terme de « distribué et isolé » s'il ne partage pas de connaissances dynamiques (celles qui évoluent au cours de l'exécution) avec d'autres modules. Il est éventuellement conçu comme un serveur/un agent dans un réseau qui reçoit des demandes de clients en écoutant sur un port spécifique et renvoie les réponses correspondant à ces demandes.

Tout d'abord, nous pouvons dire que le synthétiseur de la parole est un module pouvant être totalement indépendant de la tâche. Nous avons donc conçu ce module comme un serveur, qui peut gérer indépendamment l'écoute sur un port des demandes de synthèses pour la chaîne de caractères d'un client, et le renvoi du fichier sonore (sous format WAV) au client.

Nous constatons ensuite que certaines connaissances sont partagées entre le module de reconnaissance de la parole et celui de compréhension, représentées par les référents de tâche. Néanmoins, ces référents ne concernent que les entités statiques (vocabulaires, noms propres…) qui servent forcément au module de reconnaissance dans la phase de construction du modèle statistique de langage. C'est pour cette raison que nous avons décidé de mettre en place le module de reconnaissance comme un serveur indépendant par rapport aux autres modules de l'architecture. Cependant, quand il reçoit les signaux sonores prononcés par l'utilisateur, il les convertit en des chaînes orthographiques et les transfère au *noyau* du système (cf. section 5.2.2.2).

Finalement, le module de compréhension est également conçu comme un module isolé. L'idée de choisir d'implémenter ce module comme un agent isolé découle du fait qu'il ne gère pas les

connaissances dynamiques. En effet, ce module fonctionne sans tenir compte de la déduction et du traitement pragmatique. Il ne traduit que l'énoncé de l'utilisateur en schémas sémantiques, en utilisant la base des ontologies, des concepts et peut donc s'exécuter de façon distribuée et autonome.

5.2.2.2 Noyau, agents et leurs interactions

En considérant les modules de reconnaissance et de synthèse de la parole comme les systèmes auditif et vocal chez l'être humain, les autres modules dans notre architecture constituent donc le « cerveau » du système de dialogue. Chacun est conçu comme un *agent* qui peut être distribué. Les agents sont coordonnés par l'agent central appelé *« noyau »* du système (symbolisé par le cercle en gris dans l'architecture). Il faut souligner que la notion « agent » est éventuellement abusive et ne respecte pas tout à fait sa définition dans le domaine de système multi-agent. En effet, Demazeau, dans [Demazeau, 2003], définit : « *un agent est une entité réelle ou virtuelle qui évolue dans un environnement, qui est capable de le percevoir, d'agir dans cet environnement, de communiquer avec d'autres agents et qui montre un comportement autonome* ». Selon cette définition, chaque module de l'architecture ne respecte pas un/des critères, par exemple le critère « comportement autonome » n'est pas respecté si l'on considère l'« interpréteur pragmatique » comme un agent. Cependant, en négligeant certains critères, nous considérons chaque module comme un agent, et tous ces agents constituent un système multi-agent de dialogue.

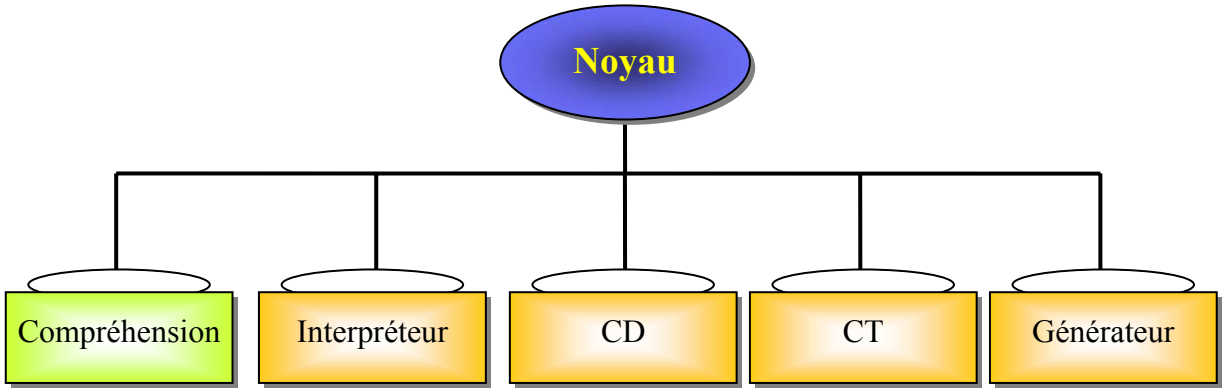


Figure 5.2 : Hiérarchie des agents dans le système Mélina

En s'inspirant du modèle d'architecture logicielle PAC (**P**résentation, **A**bstraction et **C**ontrôle) de [Coutaz, 1990], nous avons choisi le mécanisme d'interactions entre les agents de ce modèle avec l'organisation hiérarchique arborescente des agents illustrée comme dans la figure 5.2. Dans cette organisation, chaque module «*Compréhension, Interpréteur, Contrôleur du dialogue (CD), Contrôleur de la tâche (CT) et Génénateur* » est un fils de l'agent ***Noyau***, qui est considéré comme un agent ciment en maintenant la cohérence entre ses agents fils.

La communication entre ces agents via des messages doit être arbitrée par le noyau, c'est-à-dire que les messages d'échanges entre deux agents ne peuvent jamais être directement transférés. La structure d'un message contient deux paramètres (***Msg*** ; ***Data***) : ***Msg*** spécifiant le nom de

l'agent destinataire et l'action que cet agent doit effectuer, et ***Data*** étant les données transférées. Lorsqu'un agent veut communiquer une information, il envoie un message complété de ces deux paramètres à son « père » Noyau. Ici, Noyau décode l'information dans ***Msg*** et connaît donc l'agent destinataire auquel transférer ce message. L'agent destinataire effectue ensuite l'action codée dans ce message et le processus d'interaction entre deux agents s'achève alors.

Nous avons implémenté l'infrastructure de notre architecture en langage de programmation JAVA, en tenant bien compte de l'approche présentée ci-dessus. La facette d'interaction de tous ces agents est déjà bien développée sous forme de classes en Java. Nous avons groupé les agents Interpréteur, Contrôleur du dialogue, Contrôleur de la tâche et Générateur en les mettant en place dans une machine. Deux raisons nous ont amené à grouper ces agents. Premièrement, tous ces agents doivent garder en mémoire l'historique du dialogue : les tours de parole, les actes du dialogue, les stratégies, les buts dialogiques précédents qui constituent les connaissances dynamiques de chacun d'entre eux. La deuxième raison est celle que nous avons déjà abordée : la quantité de données manipulées au cours d'un dialogue n'est pas volumineuse. Nous allons décrire tous les agents de notre architecture à la section 5.4.

5.3 Collecte du corpus de dialogue

5.3.1 Corpus de dialogue

Les corpus de dialogue jouent un rôle important dans le processus de construction d'un système de dialogue. L'utilisation de corpus de dialogue est nécessaire pour étudier, analyser et déterminer les différents termes véhiculés dans un dialogue, au niveau de la sémantique, de la pragmatique, et même au niveau de la reconnaissance de la parole.

Dans le domaine du dialogue, le corpus de dialogue est divisé en deux catégories selon le type de collection :

- Corpus de dialogue réel qui collecte des dialogues homme-homme dans des situations réelles. L'analyse de ce corpus permet de mettre en évidence les phénomènes linguistiques et dialogiques inhérents naturellement à la tâche considérée [Caelen et al. 1997].
- Corpus de dialogue simulé qui est enregistré en utilisant la méthode « *Magicien d'Oz* ».

La technique dite du « Magicien d'Oz », ou bien MOZ, consiste à simuler le fonctionnement d'un système interactif à développer par des experts humains (ou bien des « compères », des magiciens) et permet dans notre cas de recueillir des corpus de dialogue homme-machine. Cette technique permet également d'observer les comportements de l'utilisateur dans son interaction avec le système simulé. Au cours de cette phase, les composants simulés sont gérés par le magicien en utilisant des moyens spécifiques, et les utilisateurs sont incités à croire qu'ils interagissent avec un système réel. Dans le cas où l'utilisateur invoquerait une fonction qui n'est pas disponible dans le système observé, le magicien doit alors simuler l'effet de cette fonction. Par l'observation des

comportements induits par cette technique, les concepteurs peuvent identifier les éventuels besoins des utilisateurs pour réaliser un ensemble particulier de tâches et peuvent évaluer l'interface particulière employée pour accomplir ces tâches.

Le Magicien d'Oz est toujours une méthode adéquate pour la simulation des systèmes de dialogue oral homme-machine. Cependant, sa réalisation est relativement coûteuse, car elle requiert un expert humain qui doit être bien formé et bien encadré. Le MOZ est donc approprié au développement d'un système complexe (comme dans le cadre du projet PVE)

Ces deux méthodes présentent effectivement des intérêts, ainsi que des lacunes spécifiques. Le corpus de dialogue réel constitue un outil fiable pour bien comprendre et circonscrire le « domaine naturel » relevant de la tâche considérée. Par contre, les dialogues homme-homme constituent une idéalisation peu opérationnelle pour la conception des systèmes de dialogue homme-machine [Caelen et al, 1997]. De son côté, la technique du magicien d'Oz participe tôt dans le processus de conception et demande un travail de conception très fin, pour arriver à une simulation totalement réaliste. De plus, les scénarios manipulés par cette technique doivent être proches du réel, pour justifier et valider le système à développer. C'est donc un défi difficile de concevoir une simulation complète, rendant compte de toutes les situations d'interaction.

5.3.2 Collecte de corpus dans le projet PVE

Dans le cadre du projet PVE, en été 2002, nous avons collecté des dialogues homme-homme dans le but de construire le corpus réel de dialogue, devant servir à la construction des modules nécessaires du système de dialogue.

Thèmes	Dialogues
Réservation de salle	28
Réservation de matériels	5
Réservation de chambre	14
Renseignement	17
Rendez-vous	34
Re-direction	45
Standard	67
Total	210

Tableau 5.1 : Thèmes du corpus de dialogues réels

Le tableau 5.1 présente les thèmes de notre corpus de dialogue réels. Nous avons enregistré 801 dialogues homme-homme dans différents contextes. Ces dialogues ont été analysés, puis triés par thèmes. Et finalement, 210 dialogues correspondant à sept thèmes différents ont été retenus. Parmi ces dialogues, 35 ont été annotés en acte de dialogue, but et stratégie de dialogue (cf. section

5.3.3). Leurs transcriptions ont été également utilisées pour construire le modèle statistique de langage dédié au module de reconnaissance de la parole.

L'analyse de dialogues réels nous a permis d'avoir une vue globale concernant les demandes de l'utilisateur face à un service visé par le système de dialogue. Cependant, ce corpus n'illustre que les propriétés communicatives entre les humains et d'où l'absence des attitudes de l'utilisateur devant une machine gérée par le système de dialogue. C'est pour cette raison que nous avons effectué la simulation « Magicien d'Oz ».

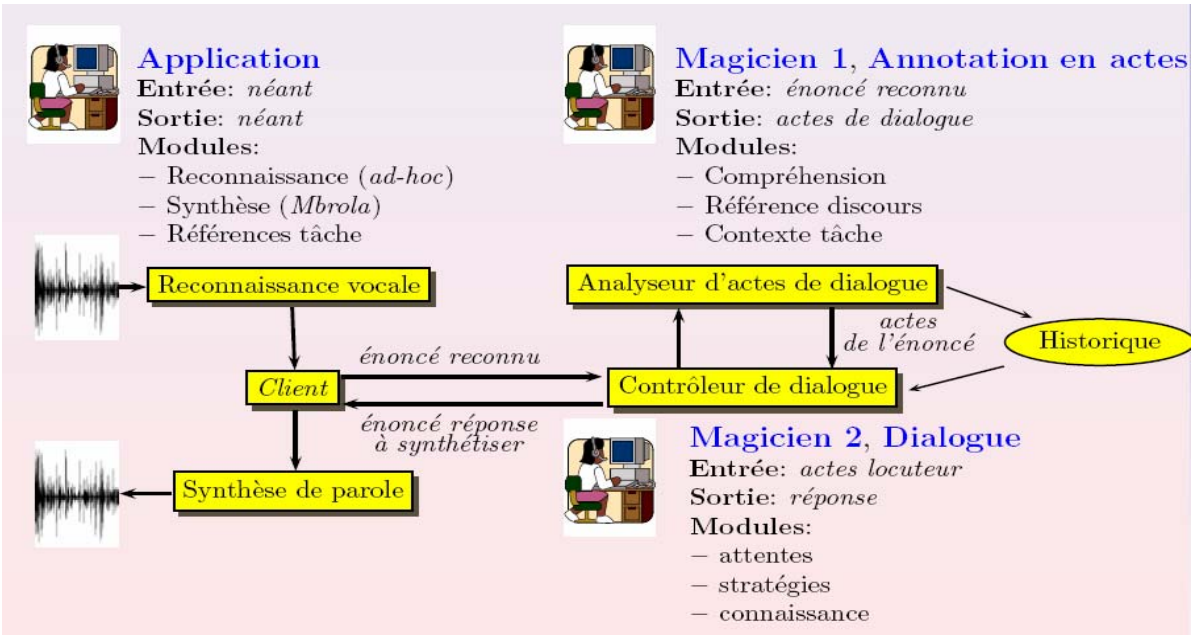


Figure 5.3 : La plate-forme Magicien d'Oz [Fouquet, 2004]

Dans le cadre du projet PVE, la simulation « Magicien d'Oz » a été effectuée par deux magiciens différents : l'un est restreint à la partie de compréhension et d'interprétation et l'autre simule le contrôleur du dialogue et de la tâche. Cette plate-forme de simulation a été réalisée par [Fouquet, 2003 ; Fouquet, 2004], et illustrée par la figure 5.3.

Au cours de cette simulation, nous avons collecté 86 dialogues homme-machine correspondant à six tâches différentes : réservation de salle et de matériel, prise de rendez-vous, envoi de documents, fonction de standard et redirection d'appel, tenue d'agenda, communication d'information. [Fouquet, 2004]. Ces dialogues constituent effectivement le corpus de simulation « Magicien d'Oz » que nous avons utilisé lors de la phase de conception, de construction de la base de concepts du module de compréhension, et de validation de notre modèle de gestion du dialogue dans le contrôleur du dialogue…

5.3.3 Annotation de dialogue

L'annotation est normalement définie comme toutes les activités qui consistent à caractériser les dialogues dans un corpus [Jacobson, 2000]. Elle est considérée également comme une étape qui illustre les apports d'information de nature interprétative aux données brutes.

En reprenant les notions concernant le modèle stratégique, l'annotation de dialogue illustrée dans cette thèse se fonde sur l'enchaînement de dialogue et sur les trois éléments essentiels : acte, but et stratégie de dialogue. Toutes les activités dans un dialogue sont modélisées autour de ces trois éléments.

L'exemple 5.1 montre un dialogue exemplaire avec les annotations du dialogue en trois éléments principaux – acte avec le contenu propositionnel P, but b et stratégie δ de dialogue. L'évolution de ce dialogue se déroule en bien respectant les trois étapes présentées dans la section précédente.

	Enoncé	Acte	P	b	δ	Evolution
S	Bonjour,	F^P	Salutation		D	*Ouverture*
	ici Mélina, le service de réservation du laboratoire CLIPS, je vous écoute	F^S	Présentation			
U	Bonjour,	F^P	Salutation			
	J'aimerais réserver	F^F	Réservation	$?b_0$	C	*Echange*
	la salle Lafayette	F^S	Salle			
	pour le lundi 20 à 14h	F^F	Temps			
S	*Puis-je demander votre nom ?*	F^{FS}	Usager	$-b_0, ?b_1$	C	
U	*Je suis Jean Caelen*	F^S	Nom	$†b_1$	C	
S	*Bonjour M. le directeur,*	F^S	Rôle	$‡b_1$	C	
	la capacité de cette salle est de 25 places.	F^S	Capacité	$†b_0$		
	Voulez-vous des matériels ?	F^{FS}	Matériel	$?sb_0$	C	
U	Non,	F^S	Matériel	$†sb_0,‡b_0$	D	
	mais prévenez aussi les membres du projet PVE	F^F	Convocation	$?sb_1$	D	
S	Je les contacte par téléphone…	F^S	Réservation	$†sb_1$	R	
U	Parfait,	F^P	Confirmation	$‡sb_0, ‡sb_1$	D	
	merci bien et au revoir	F^P	Au revoir		D	*Clôture*
S	Je vous en prie, au revoir monsieur le directeur	F^P	Au revoir final		R	

Exemple 5.1 : Annotation d'un dialogue oral homme-machine

Dans ce dialogue, l'utilisateur et le système sont engagés dans un « jeu » dans lequel le système gère la liste de salles à réserver avec l'objectif de satisfaire la demande de réservation,

l'utilisateur tente de trouver une salle disponible au bon moment. Ce jeu s'est bien déroulé avec la convergence de leur souhait et le dialogue est donc terminé de manière efficace avec succès.

5.4 Mise en œuvre

La mise en œuvre des modules dans notre architecture est effectuée à partir de trois idées principales : tirer parti des systèmes existants de manière efficace et robuste, respecter le rôle de chaque module dans l'architecture et maintenir le défi de « généricité ». Les travaux de réalisation de chaque module sont différents, ce qui explique la disparité de présentation. De plus, la bonne intégration des modules dans un système, qui est identifiée comme étant la clé du succès du système de dialogue [McTear, 2002], est également un objectif à atteindre.

Dans le cadre de cette thèse, nous nous concentrons essentiellement sur les phénomènes concernant le contrôleur du dialogue. Des approches nouvelles et originales, proposées dans les chapitres précédents, seront mises en pratique afin de prouver leur validité, ainsi que leur efficacité… En ce qui concerne les autres modules, la solution que nous avons choisie n'est éventuellement pas la meilleure. Les principaux critères pris en compte dans la mise en œuvre sont l'intégrabilité, la facilité, la rapidité. L'intégrabilité est privilégiée, afin de satisfaire tout d'abord l'exigence de notre architecture qui est modulaire, distribuée et générique. L'harmonie et la cohérence de tous les composants dans notre architecture jouent également leur rôle dans l'intégrabilité. La facilité est considérée, non seulement du point de vue de l'implémentation, mais également dans l'ensemble des défis d'un système de dialogue. Le dernier critère devrait prouver l'efficacité et la pertinence de nos approches via un vrai système de dialogue avec le « cerveau » représenté par le contrôleur du dialogue, en ignorant si possible les contraintes provenant d'autres modules. Bien évidemment, d'autres critères sont également pris en compte comme l'efficacité, la robustesse, la fiabilité, etc. mais ne sont pas mis au même niveau que les trois précédents. Tous les autres modules, sauf le contrôleur du dialogue, sont construits à partir de cette idée.

5.4.1 Reconnaissance automatique de la parole

Le module de reconnaissance automatique de la parole continue (RAP) a pour le rôle de traduire les signaux sonores d'entrée en une chaîne orthographique en sortie. Généralement, un module de RAP contient au moins deux modules principaux : un module de décodage acoustico-phonétique et un module de modélisation du langage [Baggia et al., 1999].

Pour le module de RAP de Mélina, nous avons utilisé le module de décodage acoustico-phonétique en français de *Nuance*[17], capable de traiter les signaux téléphoniques (normalement de basse qualité) et possédant des primitives pour bâtir un système de RAP générique. Le processus de reconnaissance de la parole de Nuance est illustré par la figure 5.4.

[17] Site web : http://www.nuance.com

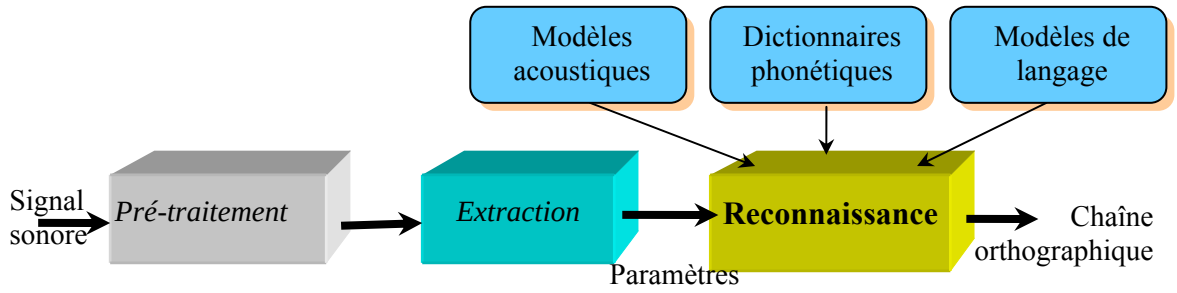


Figure 5.4 : Schéma du processus de reconnaissance de la parole par Nuance

Dans ce processus, afin de renforcer le taux de reconnaissance, le signal capturé est prétraité par deux étapes importantes : « *annulation d'écho* » et « *endpointing* ». L'annulation d'écho vise à améliorer la qualité du signal sonore, en diminuant les échos causés par la ligne téléphonique et l'étape « endpointing » permet au processus d'extraction de bien distinguer le silence ou le bruit ambiant, de l'énoncé.

Le processus d'extraction transforme les signaux prétraités sous forme d'onde sonore (*waveform*) en des paramètres représentant l'énoncé prononcé par l'utilisateur. Ici, les données audio sont échantillonnées à une fréquence de 8000Hz et segmentées à 10-milliseconde frames (80 échantillons par *frame*). La sortie de ce processus est un ensemble de paramètres qui contiennent des informations dérivées du signal comme l'énergie du signal, MFCC (Mel-scaled Frequency Cepstral Coefficients) [Davis et Mermelstein, 1980], taux de passage par zéro…

Le module de reconnaissance analyse ensuite cet ensemble de paramètres afin de produire une transcription de l'énoncé (ou bien la transformation du flux audio en texte). Cette analyse est effectuée en utilisant premièrement des modèles acoustiques qui identifient les phonèmes individuels, deuxièmement des dictionnaires phonétiques qui spécifient les séquences de phonèmes pour un mot, et troisièmement des modèles statistiques de langage qui déterminent un ensemble de chaînes de mots possibles de l'énoncé. L'analyse acoustico-phonétique est effectuée en se fondant sur les modèles de Markov cachés (Hidden Markov Models) [Rabiner et Juang, 1989] et sur les dictionnaires phonétiques, qui sont déjà construits dans *Nuance*.

En ce qui concerne les modèles statistiques de langage, nous avons utilisé le corpus de textes collecté à partir des articles du journal « ***Le Monde*** » en l'adaptant à un corpus collecté au cours de la phase « Magicien d'Oz » du projet PVE pour entraîner le module de RAP. Dans cette étape, nous avons sélectionné (utilisation de la technique des mots les plus fréquents, couramment utilisés…) un vocabulaire d'environ vingt mille mots avec un corpus de 200Mb de pur texte, dont chaque ligne représente un énoncé écrit.

Le taux de reconnaissance obtenu dans le contexte d'organisation de réunions est acceptable pour les énoncés complexes. Le tableau 5.2 ci-dessous présente les résultats concernant le taux de reconnaissance de ce module pour l'ensemble des énoncés enregistrés au cours de l'étape MOZ.

Estimation du module de reconnaissance		
Nombre d'énoncés	120	
Temps total	250,54s	
Score de confiance moyen	53	
Nombre total de mots	533	
Erreur totale d'insertion de mots	7	1,31%
Erreur totale de suppression de mots	58	10,88%
Erreur totale de substitution de mots	64	12,00%
Taux d'erreur	**=129**	**=24,20%**
Nombre d'énoncés faux	40	33,33%
Nombre d'énoncés rejetés	8	6,67%
Temps moyen de reconnaissance	**0,80s/énoncé**	

Tableau 5.2 : Une estimation du module de reconnaissance de la parole

A partir de ce tableau, pour notre module de reconnaissance en particulier et pour tout système de reconnaissance de la parole en général, nous constatons que l'efficacité n'est pas parfaite. La reconnaissance automatique de la parole est toujours confrontée à des problèmes, tels que l'utilisateur employant des mots courants mais hors de son vocabulaire, le bruit couvrant l'énoncé, la qualité de transmission n'étant pas impeccable… et de plus, la statistique n'a pas été bien évaluée en raison de cette caractéristique intrinsèque. Tout cela conduit à des problèmes d'incompréhension des autres modules et nécessite ainsi des solutions adéquates, afin de diminuer les erreurs portant sur le système de dialogue.

5.4.2 Compréhension sémantique

Le module de compréhension joue un rôle important dans un système de dialogue. Il fournit un schéma sémantique compréhensible par le module d'interprétation pragmatique à partir de la chaîne textuelle produite par le module de RAP. Il doit donc être robuste et avoir une faible sensibilité aux erreurs venant du module de RAP. Sa qualité a donc un impact direct sur la performance du système.

5.4.2.1 Schéma sémantique

Le processus d'analyse des énoncés se déroule normalement selon deux étapes : d'abord identifier le sens d'un énoncé en s'appuyant uniquement sur les mots sans tenir compte de l'historique du dialogue, et puis décoder les informations contextuelles. Dans notre architecture, le module de compréhension sémantique effectue la première étape et le décodage est réalisé par le module d'interprétation pragmatique. En quelque sorte, on pourrait considérer la première étape comme la compréhension littérale [Bousquet-Vernhettes, 2002].

L'objectif principal de la compréhension sémantique est d'extraire les sens utiles d'un énoncé prononcé par l'utilisateur [Pérennou, 1996]. Ceux-ci seront enrichis par le module d'interprétation, afin de fournir les connaissances les plus pertinentes au contrôleur du dialogue. L'extraction du sens utile d'un énoncé est ainsi vue comme la transformation de la chaîne orthographique en une représentation interprétable par le contrôleur du dialogue. Nous appelons cette représentation *« schéma sémantique »* de l'énoncé.

Dans la littérature, plusieurs formalismes permettant la représentation d'une phrase ont été proposés avec des approches différentes. L'approche logique utilise actuellement la logique des propositions, la logique des prédicats ou la logique modale pour représenter une phrase. Une autre approche, développée par Sowa [Sowa, 1988], représente une phrase par des graphes, dits *graphes conceptuels* (appelés aussi réseaux sémantiques ou graphes Sowa). D'autres approches de représentation, que nous pouvons citer ici, sont les grammaires hors-contexte, les grammaires de cas, les réseaux de neurones, les grammaires d'unifications, etc.

5.4.2.2 Analyse par règles et système Phoenix

L'une des caractéristiques importantes de la langue naturelle parlée est la spontanéité. Elle caractérise la production orale comme ne respectant pas toujours les règles de l'écrit, d'où certains phénomènes tels que les répétitions, les ruptures, les hésitations, etc. L'analyse grammaticale proposée par les grammaires hors-contexte, axée sur la structure syntaxique de l'énoncé, n'est donc pas suffisante. La représentation sémantique devrait être envisagée selon un ensemble de cas, de règles au sein du contexte d'application. Le formalisme de grammaire de cas [Fillmore, 1968] est une des approches pour l'extraction du sens de l'énoncé. L'idée principale de cette approche est de constituer des concepts à partir des mots reliés par un sens élémentaire en considérant cinq cas : le cas sémantique, le cas de surface, le cas conceptuel, le marqueur de cas et la structure de cas [Bruce, 1975].

L'analyse sémantique utilisant des grammaires de cas ou des règles grammaticales est une approche qui offre plusieurs avantages au processus de compréhension dans un dialogue oral homme-machine. Premièrement, c'est un formalisme qui est fortement orienté vers la sémantique liée essentiellement à l'application. Il présente un modèle de la structure profonde d'une phrase, en tenant éventuellement compte des contraintes syntaxiques. Deuxièmement, il permet d'effectuer le traitement automatique des phrases agrammaticales, qui est l'un des problèmes majeurs du DHM, surtout pour les systèmes de dialogue grand public.

Nous présentons maintenant le système Phoenix, présenté par Ward [Ward, 1991 ; Ward, 1994][18], qui est un analyseur sémantique utilisant la grammaire de cas sous forme de règles. Notre module de compréhension sera développé en s'appuyant sur ce système avec des adaptations importantes destinées à nos besoins.

[18] A l'époque, il travaillait à l'université Carnegie Mellon – CMU, et maintenant, ce système est disponible au site de l'université Colorado (http://communicator.colorado.edu/phoenix/download.html) où il est professeur

La représentation sémantique dans Phoenix est composée de schémas (frame), dont chacun contient un ensemble de paires attribut/valeur (slots) représentant une partie de l'information. Chaque *slot* est organisé par un automate à états finis, qui spécifie toutes les possibilités de prononciation, ce qui caractérise ce slot. Un tel automate représente donc la transcription des règles de la grammaire hors-contexte de ce slot. La grammaire est écrite de manière à ce que les phrases puissent être isolées ou intégrées dans une autre phrase. Les morceaux d'une phrase, qui ne respectent pas les règles grammaticales, peuvent aussi être analysés par le système. La grammaire est compilée à un ensemble de ces automates sous forme des réseaux de transition récursifs RTNs (Recursive Transition Networks). La partition est effectuée en prenant en compte l'idée principale : un gros automate pourrait être fractionné en plusieurs petits automates. Chaque automate peut en appeler un autre, pour réduire la taille de la grammaire (et de la mémoire du système également). Ces automates sont utilisés au cours de l'analyse de l'énoncé.

Les figures 5.5 et 5.6 montrent un exemple de schéma sémantique avec ses automates RTNs et le résultat d'analyse de l'énoncé « *show fares of flights from Denver to Boston on United* ». Phoenix fournit en sortie un schéma sémantique dont chaque slot contient l'arbre d'analyse sémantique d'un groupe de mots. Le nom de l'arbre reprend le nom de slot.

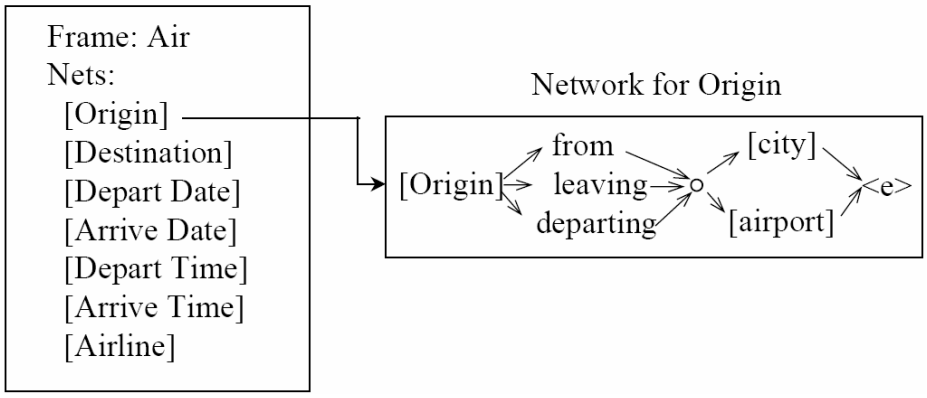


Figure 5.5 : Exemple d'un schéma - frame

Input: Show fares of flights from Denver to Boston on United

Parse:

```
Frame: Air

[Field]( show [_fares](fares of flights))
[Origin]( from [City]( Denver ))
[Destination]( to [City]( Boston ))
[airline]( on [AirlineName]( United ))
```

Figure 5.6 : Résultat d'analyse d'un énoncé

Le processus d'analyse, ainsi que l'algorithme de recherche de slot, et de remplissage du schéma sémantique en sortie, sont décrits en détail dans le manuel d'utilisation de ce système

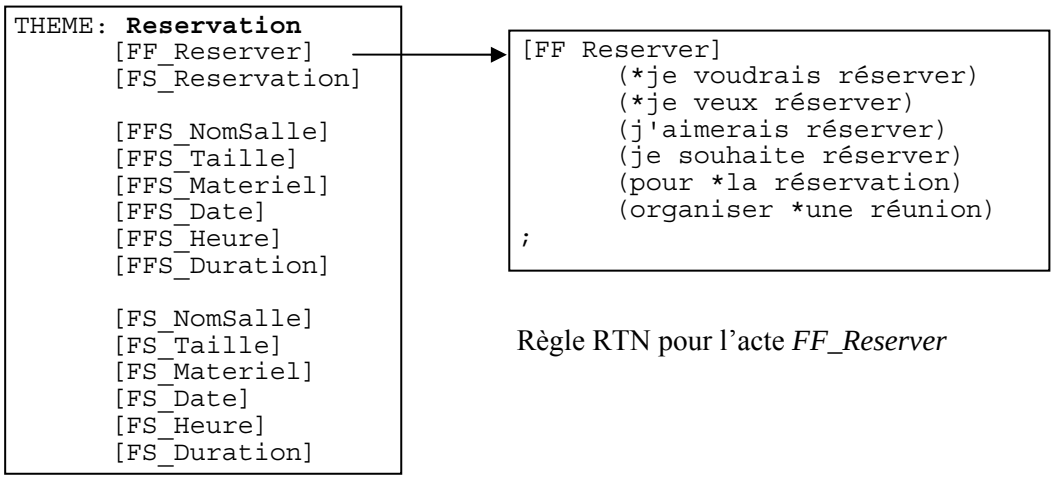
[Phoenix, 2002]. Quant à son efficacité, lors du test officiel présenté dans [Pallett et al., 1995], le système affichait un taux d'erreurs de réponse de 3,8%, mesuré sur les transcriptions de requêtes.

5.4.2.3 De Phoenix au module de compréhension sémantique

L'efficacité du système Phoenix attire beaucoup notre intérêt dans l'optique de construire un analyseur robuste, extensible. De plus, comme nous l'avons signalé au début de la section 5.4, notre sujet de thèse concerne surtout le contrôleur du dialogue et seuls les critères « intégrabilité, harmonie et rapidité » sont considérés pour l'implémentation. En tenant compte de ces critères, nous avons décidé d'adapter le système Phoenix à nos besoins pour le module de compréhension.

Tout d'abord, l'idée principale de Phoenix a été intégrée et implémentée dans notre module. Cependant, les schémas sémantiques sont organisés sous forme de thèmes. Chaque thème est représenté par un ensemble d'actes de dialogue différents, pouvant apparaître dans ce thème. L'acte de dialogue est ensuite reconnu et rempli en faisant des recherches dans les automates RTNs, qui expriment les règles, les prononciations possibles du sens de cet acte.

Nous pouvons constater que l'identification de l'acte de dialogue est déjà codée dans la grammaire du module de compréhension. En effet, cette idée part du fait que la reconnaissance de l'acte de dialogue est en cours d'élaboration et selon nos connaissances actuelles, il n'existe pas encore un système capable d'identifier automatiquement tous les actes de dialogue de manière correcte et efficace. De plus, la force illocutoire de l'acte de dialogue est naturellement reliée aux mots de l'énoncé. Par exemple, l'énoncé « *je voudrais réserver une salle* » présente une demande et nous pouvons constater que cet énoncé exprime l'acte de faire-faire F^F une réservation (cf. section 2.1.3.2) .C'est pourquoi nous avons codé la force illocutoire de chaque acte de dialogue dans les schémas sémantiques, dans le but de diminuer la charge du module d'interprétation. La figure 5.7 illustre un exemple d'un thème spécifiant les actes de dialogue possibles concernant ce thème. Chaque acte est représenté par un automate RTN.



Règle RTN pour l'acte *FF_Reserver*

Figure 5.7 : Exemple d'un thème avec ses actes de dialogue

Il s'avère que les codes sources de Phoenix sont disponibles sur Internet pour l'usage non-commercial ou de recherche. Néanmoins, ils ne sont pas capables d'analyser les mots ayant des accents comme le français et la structure de la grammaire n'est pas conforme à notre besoin. Cela nous a donc conduit à adapter ces codes sources en réutilisant des parties comme la structure des automates RTNs, la structure et l'analyse de la grammaire.

La base de connaissances du module de compréhension sémantique est élaborée dans deux fichiers : l'un contient des schémas sémantiques, c'est-à-dire les thèmes concernant l'application réelle ; l'autre représente la base d'ontologies pour ces thèmes. Cette base doit spécifier tous les concepts caractérisés, tous les actes de dialogue contenant ces thèmes. Chaque concept est également représenté par des automates RTNs. Nous gardons la syntaxe pour construire cette base d'ontologies comme celle de Phoenix.

Le concept est défini entre les deux crochets et dans une ligne complète. Elle est suivie par un ensemble de règles dont chacune est considérée comme un pattern, une par ligne, chacune entre parenthèses. Des macros (nœud non-terminal dans le réseau RTN) peuvent être introduites dans une règle. Chaque macro est écrite comme un concept, sauf qu'elle n'apparaît pas dans le résultat d'analyse. Elle représente donc un moyen d'écrire des expressions dans la base d'ontologies sans ajouter des concepts inutiles. La structure d'un concept est organisée comme la figure 5.8.

```
# Commentaire/ remarque optionnelle
[Concept]
        (pattern_a)
        (pattern_b)
        (… Macro1 …)
        (… Macro1 … Macro2…)
Macro1
        (règle_1)
Macro2
        (règle_2)
;
```

Figure 5.8 : Structure d'un concept

La spécification d'un pattern est respectée sous la forme suivante :

- Des mots en minuscule sont des terminaux dans les réseaux,
- Mot en majuscule est une macro,
- Noms entre deux [] sont non-terminaux (référence à un autre concept),
- Expressions régulières : * *mot* indique 0 ou 1 répétition de ce *mot* ; + indique 1 ou plus de répétitions ; +* signifie 0, 1 ou plus de répétitions,

Les ontologies sont ensuite compilées afin de les transformer en des réseaux sous forme de RTN (qui ressemble à un automate à états finis non déterministe sauf qu'un arc peut être un symbole terminal ou bien le nom d'un autre état) qui seront stockés dans un fichier. Pour réduire la taille de la mémoire, chaque mot est assigné à un entier et tous les mots apparus dans la base d'ontologies seront stockés dans un fichier de vocabulaire [Phoenix, 2002].

Le fichier de thèmes spécifie contextuellement tous les schémas qui sont utilisés par l'analyseur. Chaque thème représente des actes de dialogue que l'utilisateur peut manifester devant le système. La structure d'un thème est illustrée comme la figure 5.9.

```
# Commentaire
THEME : Nom_Thème
        [Acte_1]
        [Acte_2]
        [Acte_3]
        …
;
```

Figure 5.9 : Structure d'un thème

Les trois fichiers des ontologies compilées, du vocabulaire et des thèmes sont chargés dans la mémoire, dès le démarrage de l'analyseur. L'analyse est effectuée de gauche à droite en essayant de coupler des concepts avec des mots dans l'énoncé en entrée. Le résultat de ce couplage constitue donc les schémas sémantiques de l'énoncé. La figure 5.10 montre les schémas sémantiques, qui sont le résultat d'analyse sémantique d'un énoncé.

```
Enoncé : je voudrais réserver une salle lundi prochain vers dix heures

Schémas sémantiques :
     {Reservation:[FF_Reserver] ([_reserver]( je voudrais réserver ))}
     {Reservation:[FS_NomSalle] (une salle )}
     {Reservation:[FS_Date] ([Jour]( lundi ) [Relative]( prochain ))}
     {Reservation:[FS_Heure] ( vers [Heure] ( dix heures ))}
```

Figure 5.10 : Schémas sémantiques d'un énoncé

Le schéma sémantique en sortie représente donc des informations hiérarchiquement précises, concernant l'acte de dialogue, ainsi que son thème, qui servent à les interpréter en des actes de dialogue dans l'interpréteur.

5.4.2.4 Construction de thèmes et de base de concepts

La base de concepts, ainsi que les thèmes d'un service visé par le système de dialogue, sont clairement les éléments qui décident directement de la qualité et de l'efficacité du module de compréhension. Pour construire ces éléments dédiés au service d'organisation de réunion, au cours de la phase « Magicien d'Oz » du projet PVE, nous avons collecté un corpus de dialogues qui s'est déroulé avec tous les scénarios concernant ce service. L'analyse de ce corpus nous permet donc de bien spécifier tous les thèmes concernant ce service, ainsi que toutes les entités de ces thèmes, qui sont présentés dans l'annexe B.

5.4.2.5 Communication d'entrée/sortie

La communication entre le module de compréhension et les autres dans l'architecture est établie en se fondant sur le réseau. Ce module est donc construit comme un serveur : recevoir une chaîne de caractères provenant d'un client en entrée et lui rendre le schéma sémantique en sortie.

5.4.2.6 Problème à améliorer

L'analyse effectuée dans ce module s'appuie essentiellement sur tous les mots sortis du module de reconnaissance de la parole spontanée (dans le cadre du dialogue oral homme-machine, nous ignorons pour l'instant les problèmes d'intonation, de mimes, etc.). Pourtant, aucun système actuel de reconnaissance de la parole ne produit exactement en texte ce que l'on prononce. L'incompréhension est donc un grand problème posé à ce niveau là.

De plus, l'analyse morpho-syntaxique apporte également des connaissances utiles qui peuvent aider le groupement des mots ayant le même sens, par exemple « je voudrais réserver » - « je veux réserver ». L'efficacité de ce module peut être donc améliorée, si nous effectuons l'analyse syntaxique avant celle sémantique [Rouault et Manes-Callo, 2003].

5.4.3 Interpréteur pragmatique

A partir des schémas sémantiques produits par le module de compréhension, l'interpréteur pragmatique a pour rôle essentiel de résoudre les références (par exemple : noms propres, expressions de désignation, anaphores, déictiques, etc.) en utilisant des connaissances du contexte et l'historique des échanges précédents entre l'utilisateur et le système. Il doit ensuite traiter les présuppositions et les implicatures conversationnelles, lorsqu'il y en a (par exemple : *Est-ce que l'Aquarium est libre demain ?* Il faut comprendre que *Aquarium* est une salle et que *libre* veut dire non réservé et que *demain* signifie pendant les heures ouvrables). Une fois que tous ces problèmes sont résolus, l'interpréteur doit rendre au module du contrôleur du dialogue les actes de dialogue relativement aux schémas sémantiques reçus en entrée.

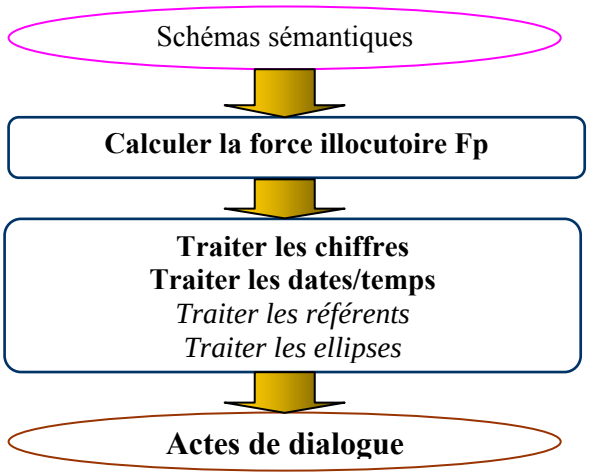


Figure 5.11 : Schéma de construction des actes de dialogue

La figure 5.11 montre les étapes essentielles de traitements dans l'interpréteur. Tout d'abord, il doit analyser les schémas sémantiques entrés, afin de trouver des marqueurs pragmatiques. A partir des ces marqueurs, il détermine le type d'acte de dialogue, c'est-à-dire la force illocutoire exprimée dans l'énoncé de l'utilisateur. Ensuite, en balayant tous les concepts dans les schémas sémantiques

entrés, il doit rattacher tous les chiffres, les dates, les heures/minutes aux concepts correspondants. Les étapes suivantes s'efforcent de résoudre les ellipses (par exemple « réservez moi l'Aquarium alors » - ici Aquarium doit effectivement être interprété comme le nom de la salle), les référents, les anaphores (par exemple « c'est moi qui vient d'appeler »…), les présuppositions, les implicatures conversationnelles. Finalement, l'interpréteur construit les actes de dialogue sous forme compréhensible par le contrôleur du dialogue, à partir des analyses de tous ces traitements.

5.4.3.1 Structure d'un acte de dialogue

Détaillons maintenant la structure d'un acte de dialogue. Un acte de dialogue est codé par les attributs illustrés en tableau 5.3 suivants :

Attributs	Description
Type F_p	Type de force illocutoire : faire faire F^F, faire savoir F^S, faire faire savoir F^{FS}… (cf. section 2.3.3.1)
Thème	Elément qui précise la position cet acte portant sur l'activité d'un dialogue.
Concept	Paire concept/valeur représentant le contenu propositionnel P. La valeur d'un acte peut être donnée par l'utilisateur ou le système.
Valeur	
Modalité	Modalité de l'acte : déclarative (celle qui est au mode indicatif), interrogative (questions), impérative (ordres), conditionnelle, exclamative (expression des états mentaux) et optative (expression des souhaits).

Tableau 5.3 : Structure d'un acte de dialogue

L'énoncé de l'utilisateur est décomposé en un ou plusieurs actes de dialogue de manière élémentaire et indivisible. Reprenons l'exemple précédent illustré dans la figure 5.10, l'interpréteur pragmatique doit produire les actes de dialogue illustrés par la figure 5.12 :

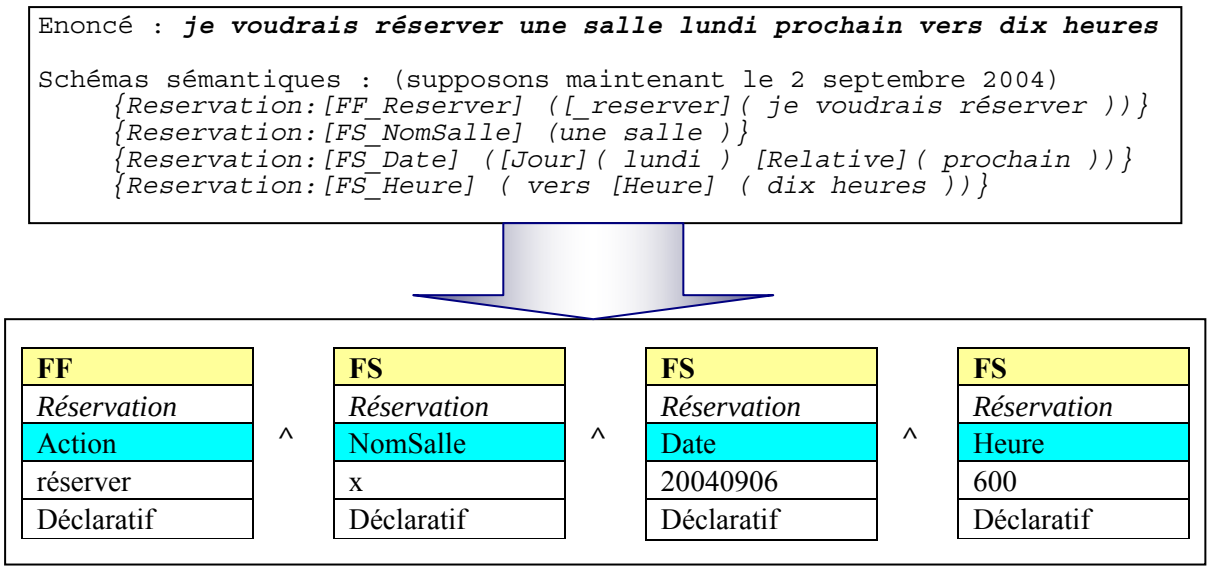


Figure 5.12 : Représentation en acte de dialogue d'un énoncé

Dans cet exemple, en se fondant sur des marqueurs pragmatiques bien représentés dans les schémas sémantiques, l'énoncé de l'utilisateur a été donc interprété en quatre actes : F^F(Action(réserver)) & F^S(NomSalle(x)) & F^S(Date(20040906)) & F^S(Heure(600)).

5.4.3.2 Traitement de chiffres et de temps

La sortie du module de reconnaissance ne fournit qu'une chaîne de caractères purement textuelle, y compris les chiffres (par exemple le chiffre 22 est représenté par « vingt deux »), la date (ex. « lundi le six septembre »), l'heure… L'interpréteur doit donc traduire ces éléments en valeurs ayant le type adéquat.

La conversion d'un pattern représentant un nombre en chiffres est triviale et nous n'abordons donc pas ce point ici. Néanmoins, la conversion des dates, heures est plus compliquée en raison des référents possibles pour ce type de données.

En effet, l'interpréteur doit résoudre tous les référents concernant la date, par exemple : « maintenant », « lundi prochain », « mardi dernier », « le dernier dimanche de ce mois », « le premier lundi du mois prochain », etc. De plus, des patterns concernant la date peuvent être une date ayant le nom déjà défini – date particulière, par exemple : « l'Ascension », « Noël », etc. L'analyseur sémantique ainsi que l'interpréteur doivent prendre en compte ces termes et bien traiter chaque cas précis. Dans le service d'organisation de réunion, nous avons traité tous les problèmes de référents de date, mais les dates particulières sont laissées de côté pour l'instant. Les dates de référence sont évidemment calculées en se fondant sur la date du moment de traitement.

Les référents relativement à l'heure sont également un problème à résoudre. Au lieu de donner directement l'heure précise, l'utilisateur peut également prononcer le temps de manière indirecte, par exemple : « dès que possible », « maintenant », etc.

5.4.3.3 Traitement de référents, d'ellipses

Le problème référentiel est considéré comme une tâche principale et très difficile pour l'interpréteur pragmatique. Du point de vue linguistique, les références consistent à un ensemble de marqueurs comme l'article défini (le, la, les), indéfini (un, une, des), partitif (du, de la, des) ; l'adjectif démonstratif (ce, cette, ces) ; l'adjectif possessif (son, sa, ses, etc.) ; le pronom personnel (il, elles…) ; le pronom possessif (le mien, la mienne…) ; etc. On distingue ces références par les références anaphoriques (concernant la co-référence), les références spatiales, les références temporelles (traitées dans la section ci-dessus)…

Les ellipses sont définies comme l'effacement d'une unité linguistique dans un contexte donné pour réduire un vocabulaire ou éviter une répétition. Par exemple dans l'énoncé « est ce que l'Aquarium est disponible ? », l'interlocuteur doit comprendre que « Aquarium » est le nom d'une salle même si le sens sémantique « salle » n'est pas abordé dans cet énoncé.

En attente de résultats prometteurs d'une autre thèse appliquant la théorie SDRT (Segmented Discourse Representation Theory) [Asher et Lascarides, 2002] au dialogue oral, nous ne traitons

que des références, des ellipses simples reliées directement à la tâche de réservation de salle (noms de salle, noms de matériel..). L'extension de ce module est donc encore ouverte.

5.4.3.4 Construction des marqueurs dialogiques d'incompréhension

Comme nous avons présenté la notion de marqueur dialogique d'incompréhension MDI dans la section 3.3.3 au chapitre 3, l'interpréteur pragmatique se situe effectivement à la dernière étape du processus de détection de ces marqueurs.

Bien évidemment, la construction des MDIs se fonde notamment sur le résultat du module de compréhension et sur l'historique des énoncés précédents de l'utilisateur. En effet, dans le cas d'incompréhension complète (aucun schéma sémantique n'est produit), l'interpréteur génère l'acte Faire-Savoir du marqueur MDI-1 et l'envoie au contrôleur du dialogue. L'identification des marqueurs ayant le type MDI-2 est plus compliquée que MDI-1 : elle doit être fondée sur les actes du système dans le tour de parole précédent et certainement sur des morceaux qui ne sont pas encore consommés par le module de compréhension. L'intégration de la théorie SDRT à l'interpréteur dans un premier temps complétera cette tâche et améliorera les problèmes concernant l'incompréhension partielle.

5.4.3.5 Extension : relation rhétorique et SDRT

La théorie SDRT est initialement proposée par N. Asher dans [Asher, 1996] : « *La SDRT intègre et étend la sémantique dynamique, telle qu'elle est développée dans la DRT, en introduisant une analyse plus fine de la structure du discours. En SDRT, l'analyse du discours fournit non une seule DRS, mais une structure complexe où des DRS sont reliées entre elles par des relations de discours pour former des « Segmented Discourse Representation Structures » ou SDRS. Des SDRS peuvent elles-mêmes être reliées par des relations de discours pour former des structures hiérarchiques.* »

De façon formelle, la SDRT est une extension de la DRT, qui permet la référence à des segments de discours, et qui établit une hiérarchie entre ces segments de discours par des *relations rhétoriques*. Les relations rhétoriques enrichissent le contenu sémantique, et permettent de représenter la structure logique du discours. Elles améliorent le traitement des liens anaphoriques par rapport à la DRT. En SDRT, on peut représenter les anaphores propositionnelles (référence à un segment du discours). Il n'existe pas de liste définitive des relations rhétoriques. Pour l'analyse des discours narratifs, [Asher & Lascarides, 2002] proposent les relations *narration, résultat, arrière-plan, explication, conséquence, parallèle, contraste, élaboration, etc*.

Des recherches dans l'optique d'appliquer cette théorie dans le cadre du dialogue oral homme-machine ont déjà été commencées dans notre équipe [Xuereb et Caelen, 2004]. La mise en œuvre de ce modèle et son intégration sont en cours. Le résultat prometteur de ce travail permettra véritablement d'améliorer la capacité d'interprétation pragmatique de ce module.

5.4.4 Contrôleur du dialogue

Le contrôleur du dialogue (CD) est considéré comme le « cerveau » du système de dialogue. Ses tâches principales sont de transmettre les messages aux autres modules, de calculer la stratégie adéquate, de gérer les buts et de préparer les données pour le générateur d'énoncés. Pour réaliser toutes ces fonctions, il est équipé de règles de calcul des stratégies et de règles de gestion des buts de dialogue (cf. section 3.2). De plus, pour gérer des sessions discontinues dans un dialogue multisession, le contrôleur du dialogue est également équipé d'un sous module, dit ***Expert***, assurant l'avancement du dialogue à partir de chaque session [Nguyen et Caelen, 2004b]. Nous allons voir maintenant en détail la mise en œuvre de ce module.

5.4.4.1 Interaction entre le contrôleur du dialogue et les autres modules

La communication entre les modules dans Mélina est effectuée par des messages, qui sont contrôlés par le NOYAU. Dans l'architecture générique présentée à la section 5.2.1.2, le contrôleur du dialogue CD interagit avec les trois autres modules : Interpréteur pragmatique I, contrôleur de la tâche CT et Générateur G. Nous envisageons maintenant les interactions entre eux :

- 1. Interaction entre I et CD : Les résultats de la sortie du module I sont les actes de dialogue détaillés construits à partir de l'énoncé de l'utilisateur. Ces actes doivent être amenés au CD afin de commencer un nouveau tour de parole,
- 2. Interaction entre CD et CT : Au début du dialogue, le CT doit spécifier l'agenda qui représente la tâche visée par le système au niveau dialogique (voir la section suivante). Ensuite, dans un tour de parole, le CD demande au CT de vérifier la cohérence, l'exactitude des actes de l'utilisateur et ordonne d'exécuter les actions illocutoires dans ces actes. Les effets dialogiques de ces actions seront stockés dans l'agenda. Nous détaillons ces interactions dans les parties qui suivent,
- 3. Interaction entre CD et G : Une fois que le CD génère les actes du système, il va les envoyer au Générateur, ainsi que la stratégie, le but et le thème de dialogue dans le tour actuel de parole.

Ces interactions sont illustrées par la figure 5.13.

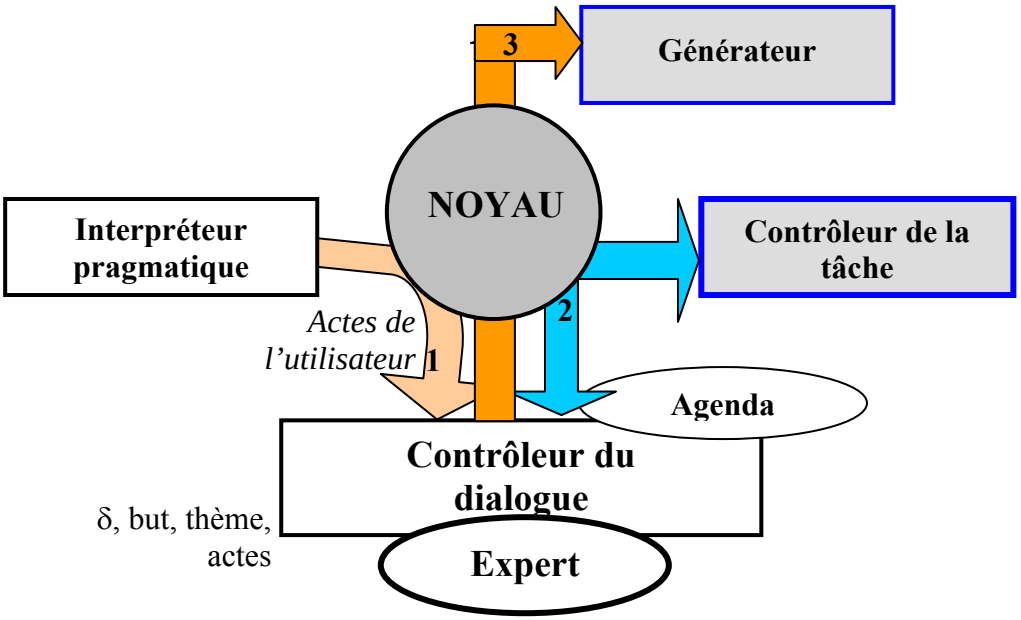


Figure 5.13 : Interactions entre le contrôleur du dialogue et les autres modules

5.4.4.2 Structure de l'agenda

L'agenda est hiérarchiquement organisé par les buts et les sous buts de dialogue (cf. section 3.2.2). Nous présentons donc la structure d'un but avant d'aborder celle de l'agenda. Un but se compose des attributs présentés en tableau 5.4 suivant :

Attributs	Description
Nom	Nom de but,
Valeur	Valeur actuelle de ce but,
Etat	Etat de ce but : nouveau, atteint, satisfait…
Poids	Priorité de ce but : impératif, négligeable, caché…
Thème	Thème de dialogue,
ListeSousButs	Liste de sous buts de ce but,
Structurel	Ses sous buts sont décomposés de façon structurelle : ET, OU, ALT
Temporel	Ses sous buts sont décomposés de manière temporelle : SEQ, PAR, SIM
TypeDeDonnée	Type de donnée de ce but : Oui/Non, entier, texte…
ValeurParDéfaut	Valeur par défaut de ce but.
Parent	Lien avec le parent de ce but dans l'agenda

Tableau 5.4 : Attributs d'un but

Cette structure permet de constituer un agenda de dialogue de manière hiérarchique et efficace. La figure 5.14 illustre un agenda détaillé pour la tâche de réservation de salle qui se compose de trois thèmes différents : « Identification » qui consiste en données personnelles de l'utilisateur afin

de l'identifier avant d'effectuer une réservation ; « Réservation » représente tous les éléments concernant la salle, le temps, le matériel ; et « Convocation » concerne le fait que l'utilisateur peut demander au système d'informer les participants à sa réunion et ce thème est donc négligeable. La relation temporelle de ce thème est séquentielle : le thème « identification » doit être satisfait avant de traiter le thème « Réservation ». La relation structurelle des sous buts dans cet agenda est conjointe (signifié par « ET ») : un but est atteint si ses sous buts impératifs sont atteints.

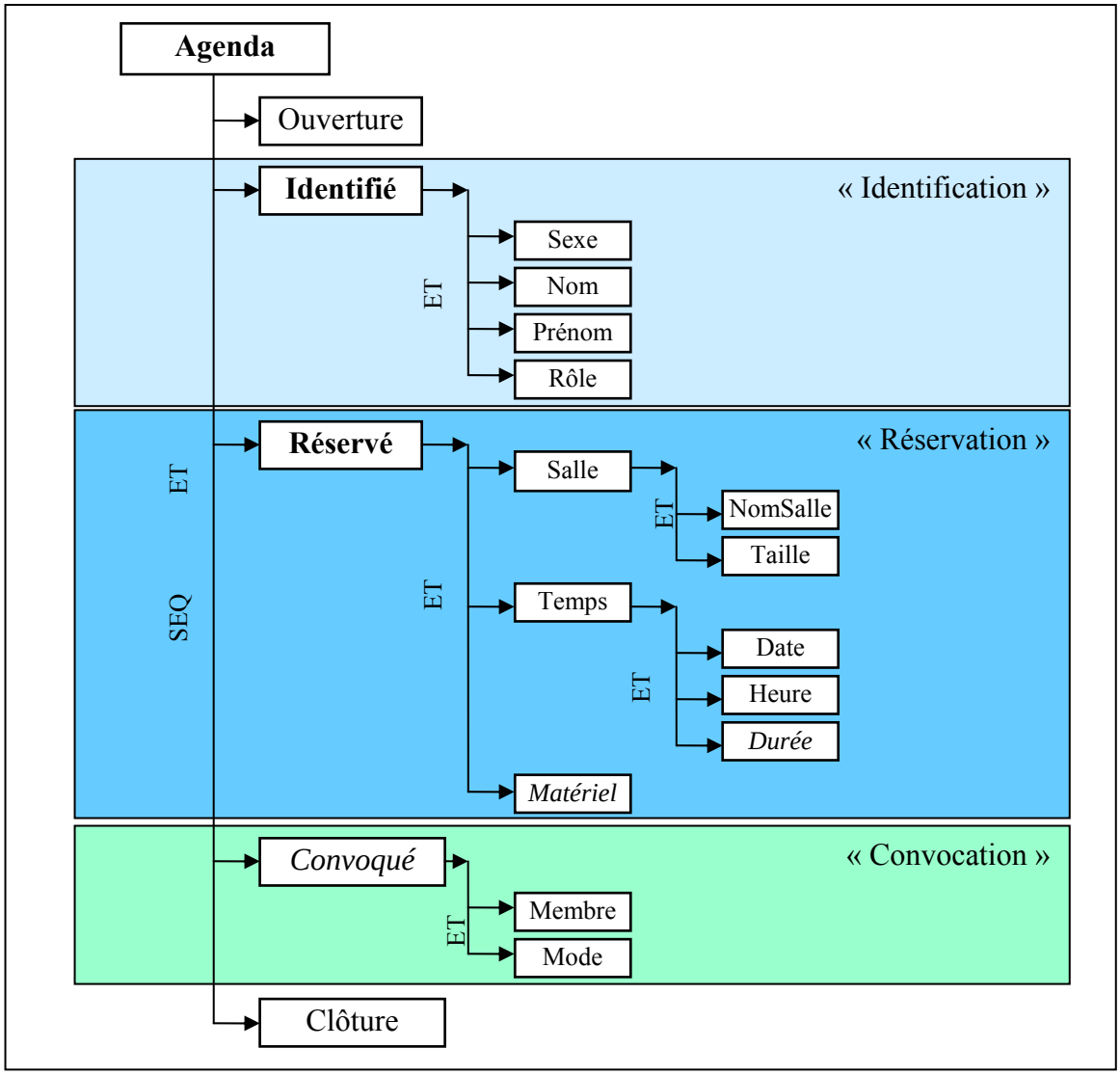


Figure 5.14 : Agenda détaillé de la tâche de réservation de salle

(Dans cet agenda, les buts en italique signifient ceux négligeables, en gras pour les buts impératifs)

En ce qui concerne la relation dans un agenda, les buts ayant un même « père » sont « frères » et un but qui est avant un autre but est considéré comme son « petit frère ».

5.4.4.3 Algorithme générique de traitement d'un tour de parole

Au cours d'un dialogue, un tour de parole est caractérisé explicitement par les actes de l'utilisateur et du système, et implicitement par la stratégie et le but de dialogue dans ce tour.

Dans un tour de parole, une fois que le contrôleur du dialogue reçoit les actes de l'utilisateur, il doit déterminer tout d'abord la stratégie, le but de dialogue et ensuite il génère les actes du système. L'algorithme générique de traitement d'un tel tour est présenté dans la figure 5.15 :

```
Entrée : liste des actes de dialogue de l'utilisateur ADU
Sortie : liste des actes de dialogue du système ADM
Arrière plan : but actuel b, stratégieδ, thème T, agenda, Expert

DEBUT
Si ADU == MDI Alors //marqueur d'incompréhension est signalé au CD
    TraiterIncompréhension(b) ;
    Retour() ;
FinSi

Agenda.MiseAJour(ADU) ;//mettre à jour l'agenda
EnvoyerAVerifier(ADU, agenda) ;//envoyer ces actes au CT pour vérifier leurs cohérences
b1 = Agenda.VérifierCohérence() ; // vérifier la cohérence dans l'agenda,
Si b1 != vide Alors // la cohérence est violée, on traite ce cas avant
    δ = coopérative ;
    ADM.ajouter(FFS, b1) ;//demander l'information de b1
    Retour() ;
FinSi

δ = CalculerStratégie() ;//calculer la stratégie

T = IdentifierTheme(T, δ, Agenda) ;//déterminer le thème actuel

OrdonnerCTEffectuer(ADU, Agenda) ; // demander au CT d'effectuer ADU

b = Agenda.DerterminerBut((T, δ, Agenda) ;//identifier le but actuel à résoudre

Si b = Conflit Alors
    Si b.état < atteint Alors //le conflit est en train d'émerger
        chemin = Expert.PlusCourtChemin() ;//chercher le chemin le plus court
        ADM.ajouter(FS,chemin) ;//informer la nature du conflit
    SiNon b.état == atteint
        ADM.ajouter(FS, «suspendre à négocier ») ;//stopper la session actuelle
        Expert.StartProcessusNegociation() ;//lancer le processus de négociation en suspendant
le dialogue actuel
    FinSi
SiNon
    ADM = GénérerActes(T, δ, b, ADU) ;
FinSi
FIN
```

Figure 5.15 : Algorithme générique d'un tour de parole

Dans un tour de parole, si l'interpréteur signale qu'il y a une incompréhension, le contrôleur du dialogue doit traiter ce cas grâce à l'algorithme mentionné dans la section 3.3.3 au chapitre 3.

Dans le cas contraire, le CD met à jour tout d'abord l'agenda avec les données fournies par les actes de l'utilisateur ADU. Ensuite, il va envoyer ces actes au contrôleur de la tâche CT, pour qu'il puisse vérifier la cohérence, juger l'exactitude, traiter les pré-conditions, … de ces actes (cf. section 5.4.5). Les résultats de cette action seront enregistrés dans l'agenda.

Une fois que le système a pris en compte la nouvelle situation correspondant aux nouveaux actes de l'utilisateur, CD vérifie par la suite la cohérence, la régularité de l'agenda. Dans cette étape, la condition temporelle est surtout considérée : si un but et ses frères sont en relation séquentielle SEQ, mais son petit frère n'est pas encore atteint tandis que l'utilisateur attaque ce but, alors l'agenda doit signaler ce problème et le CD doit traiter son petit frère avant d'attaquer ce but là en appliquant la stratégie coopérative.

Dans le cas normal, le CD continue à déterminer la stratégie actuelle en appliquant les règles présentées dans la section 3.2.4. Par la suite, il identifie le thème actuel dans l'agenda en se fondant sur l'agenda, la stratégie ainsi que le thème du tour précédent.

Les actions illocutoires codées dans les actes de l'utilisateur doivent être ensuite effectuées, dans l'optique de prise en compte de la nouvelle situation influencée par ces actions. Cela est réalisé par le CT, qui reçoit l'ordre d'exécuter provenant du CD. Les effets de ces actions d'un point de vue du but de dialogue seront pris en compte dans l'agenda.

Ensuite, le CD détermine le but actuel à résoudre avec l'aide des trois éléments : l'agenda, le thème et la stratégie actuelle. L'algorithme de gestion de buts est déjà présenté dans la section 3.2.5. Ce but peut être un but normal dans l'agenda ou bien un but concernant le phénomène de conflits dans l'arrière-plan du système.

Dans le cas où des conflits sont apparus, le CD demande au module Expert de trouver le chemin le plus court à traiter et informe sur la nature des conflits en détaillant ce chemin. Une fois qu'il reçoit la demande de traiter ces conflits de l'utilisateur, le but « conflit » devient le but atteint, le CD va suspendre le dialogue actuel (session émergente) et lancer le processus de négociation.

Finalement, dans le cas normal, le CD génère les actes du système en s'appuyant sur tous les éléments : stratégie, but, thème et aussi les actes de l'utilisateur. Tous ces éléments sont ensuite transférés au module Générateur afin de construire l'énoncé du système.

Dans le CD, le module Expert est également implémenté sous forme d'un agent avec le modèle de gestion de dialogue multisession (cf. section 4.3, chapitre 4). Il a en charge de coordonner des sessions différentes en cas de conflit, et par la suite de le résoudre par des négociations importantes dans un dialogue multisession.

5.4.4.4 Remarques

Vu l'algorithme présenté ci-dessus, nous constatons que le contrôleur de la tâche n'interagit avec le CD que via trois actions : spécifier l'agenda avec des relations structurelles et temporelles ; vérifier la cohérence des actes de l'utilisateur et exécuter les actions illocutoires dans ces actes. Cependant, tout cela est réalisé par un protocole normalisé, qui est représenté par des messages : le CD envoie ses ordres au CT et reçoit des effets dialogiques dans l'agenda. La gestion de stratégies, de buts et de l'agenda est implémentée avec ses propres règles qui ne concernent pas le modèle de la tâche. Cette remarque est très importante et nous apporte des fonctionnalités, qui peuvent diminuer et surmonter les défis du système de dialogue, que nous avons abordés dans la section 1.4.3 au premier chapitre.

5.4.5 Contrôleur de la tâche

5.4.5.1 Interaction avec le contrôleur du dialogue

Au point de vue dialogique, le contrôleur de la tâche CT doit assurer les trois tâches suivantes, dont tous les effets dialogiques seront stockés dans l'agenda de dialogue :

- Spécifier l'agenda de dialogue : dans cet agenda, il doit spécifier tous les buts ainsi que les sous buts concernant une tâche précise dans le service visé par le système. Le poids, la relation structurelle, temporelle, la valeur par défaut, le thème… d'un but doivent être bien déterminés dans cette étape,
- Vérifier la cohérence, l'exactitude des actes de l'utilisateur. Bien évidemment, cette étape est principalement le travail du CT,
- Exécuter les actions illocutoires codées dans les actes de l'utilisateur.

De plus, en cas de conflit, le CT doit construire l'arbre de buts en conflit en le transférant au module Expert (cf. section 4.3.1.2).

L'interaction entre le CT et le CD se déroule à l'aide des messages comme la communication normale entre les modules dans l'architecture que nous avons présentée.

5.4.5.2 Gestion de tâches

Du côté de la tâche, naturellement, le CD doit gérer toutes les tâches visées par le système : des objets, la base de données…

Pour le service d'organisation de réunion, nous avons élaboré quatre sous tâches différentes : une tâche représentant la demande de réservation de salle, une pour le processus de négociation, une pour la notification et la dernière consacrée à la tâche de convocation des participants. Dans ce service, le CT doit gérer des bases de données contenant les données ad hoc : droits des participants, salles et leurs descriptifs, liste de réservations, liste des utilisateurs, etc.

Vu que notre contrôleur du dialogue est générique, il est utilisé dans toutes ces tâches sans aucune modification.

5.4.6 Générateur

Le module de génération prend en entrée les consignes (acte de système, but de dialogue, stratégie de dialogue) du contrôleur du dialogue, et fournit en sortie une suite d'instructions exécutables et/ou produit une chaîne orthographique textuelle.

La génération d'une telle chaîne dépend fortement non seulement de la force illocutoire, mais également de la stratégie actuelle dans le tour actuel de parole. Par exemple, si le système génère les actes : F^{S}[Salle(Aquarium) & F^{FS}[Date(20041008)], la chaîne orthographique obtenue doit varier selon la stratégie :

- Réactive : « Je vous réserve donc la salle Aquarium vendredi 8 octobre 2004 ? »
- Coopérative : « Voulez vous réserver la salle Aquarium le vendredi 8 octobre 2004 ? »

La chaîne orthographique textuelle est générée en utilisant des segments d'énoncés prédéfinis (relevés au cours de la phase « Magicien d'Oz » du projet PVE). Dans une première version, le générateur raccorde les segments pour fournir un énoncé cohérent au regard de la réponse à fournir. La chaîne orthographique générée est ensuite envoyée au synthétiseur de parole.

5.4.7 Synthétiseur de la parole

La tâche du synthétiseur de parole est de convertir une chaîne orthographique de caractères en signal sonore (TTS – Text To Speech).

La mise en œuvre de ce module a été effectuée en développant un serveur qui reçoit des textes générés par le module Générateur et répond en renvoyant le fichier sonore au système. Bien évidemment, le temps de synthèse TTS doit être suffisamment rapide pour ne pas augmenter le temps d'interaction entre l'utilisateur et le système. Le synthétiseur, proposé par Mbrola[19] (qui est un synthétiseur de parole fondé sur la concaténation des diphones), répond à cette contrainte et a une qualité acceptable pour la première version de notre système.

5.5 Résultat : Mélina – un système de dialogue

La mise en œuvre de tous les modules présentés ci-dessus nous amène au système de dialogue Mélina. De manière à faciliter l'intégration de ses modules et profiter des outils existants, la construction de ses modules est effectuée en utilisant divers langages de programmation : l'interface vocale (reconnaissance et synthèse de la parole) et le module de compréhension sont

[19] Mbrola : http://tcts.fpms.ac.be/synthesis/mbrola.html

construits en langage C ; l'infrastructure du système, le contrôleur du dialogue, celui de la tâche, ainsi que l'interpréteur, sont développés en Java afin de profiter des points forts de traitements de chaînes de caractères, des objets communicants de ce langage…

Le domaine étudié concerne le service d'organisation de réunion : l'utilisateur interagit avec le système pour réserver une salle de réunion. En cas de conflit de salle provoqué par des réservations effectuées précédemment, l'utilisateur peut demander à Mélina de contacter les personnes concernées afin de résoudre éventuellement le conflit entre eux.

Nous présentons dans cette section quelques exemples enregistrés au cours de la phase testée du système de dialogue Mélina.

5.5.1 Evolution d'un tour de parole

Dans un tour de parole géré par Mélina, le résultat d'analyse de tous ses modules est illustré par la figure 5.16 ci-dessous :

```
Enoncé = « réservez moi la salle lafayette jeudi prochain de onze heures
à treize heures pour la réunion de geod »

Compréhension = {Reservation:[FF_Reserver] ( [_reserver] ( réservez moi )
) }{Reservation:[FS_NomSalle] ( la salle [NomSalle] ( lafayette ) )
}{Reservation:[FS_Date] ( [Jour] ( jeudi ) [Relative] ( prochain ) )
}{Reservation:[FS_Temps] ( de [Heure] ( onze heures ) à [Heure] ( treize
heures ) ) }{Reservation:[FF_Reserver] ( [_reserver] ( pour la réunion )
) }

Interpréteur = FF[Action(reserver)] & FS[NomSalle(lafayette)] &
FS[Date(20041007)] & FS[Heure(660)] & FS[Duration(120)]

Contrôleur du dialogue :
      Stratégie : COOPERATIVE
      Thème actuel : Reservation
      But atteint : réservé
      Actes du système : FS[Taille(45)]

Génération : la capacité de cette salle est de 45 places.
```

Figure 5.16 : Evaluation d'un tour de parole de Mélina

Le résultat d'analyse dans un tour de parole est enregistré dans un fichier de traces, qui sert à un double objectif : évaluer le déroulement et l'efficacité du système, et mettre au point le système Mélina.

5.5.2 Exemple d'un dialogue normal

Voici un dialogue extrait de l'historique de Mélina concernant une demande de réservation d'une salle de réunion en gardant seulement les éléments principaux dans un tour de parole. Dans cet exemple, l'utilisateur manifeste sa demande à Mélina pour la réservation d'une salle quelconque qui a au moins dix places. Le déroulement de ce dialogue est illustré, avec des annotations comme l'acte, le but et la stratégie de dialogue, en exemple 5.2.

	Enoncés et actes de dialogue	Stratégie & but
S0	ici le service de réservation du laboratoire CLIPS, je suis Mélina et vous ? *FP[Ouverture]*	δ = directive ?Ouverture
U1	bonjour c'est solange *FP[Ouverture()] & FS[Prénom(solange)]*	δ = coopérative ‡Ouverture †Usager
S1	*FS[Sexe(2)] & FS[Nom(hollard)] & FS[Role(chercheur)] & FS[Groupe(PVE)]* bonjour madame Hollard, je vous écoute…	
U2	je voudrais réserver une salle demain l'après midi vers quinze heures *F[Action(reserver)] & FS[NomSalle(x)] & FS[Date(20031105)] & FS[Heure(900)]*	δ = coopérative ‡Usager ?Réservé †Date ?Salle
S2	*FFS[Salle(x)]* quelles sont vos préférences de salle ?	
U3	il me faut éventuellement une salle de dix places *FS[Taille(+10)]*	δ = cooperative †Salle
S3	*FS[NomSalle(aquarium)] & FS[Taille(20)]* la salle aquarium serait à votre disposition, sa capacité est de 20 places.	
U4	très bien et j'ai besoin aussi d'un vidéo projecteur *FS[Reservation(OK)] & FS[Materiel(vidéo projecteur)]*	δ = coopérative †Matériel †Réservé ?Convoqué
S4	*FFS[Convocation(x)]* voulez vous que je prévienne les participants ?	
U5	ça serait génial prévenez les membres du projet pve alors *FF[Convocation(x)] & FS[Membre(pve)]*	δ = réactive ‡Réservé †Covoqué
S5	*FFS[Convocation(téléphone)]* je vais les contacter par téléphone ?	
U6	oui bien sûr *FS[Convocation(OK)*	δ = coopérative ‡Convoqué
S6	*FS[Convocation(OK)]* très bien, je vais prévenir les participants	
U7	merci beaucoup et au revoir *FS[Cloture(x)]*	δ = réactive ‡Clôture
S7	*FS[Cloture(x)]* au revoir madame hollard à bientôt.	

Exemple 5.2 : Un dialogue d'une réservation normale

5.5.3 Exemple d'un dialogue multisession

Nous présentons maintenant un dialogue multisession, afin de débloquer un conflit, enregistré dans la phase testée de Mélina.

Supposons que nous sommes maintenant mardi 5 octobre 2004. Dans l'historique de Mélina, il y a deux réservations effectuées par deux utilisateurs différents pour la salle Lafayette le jeudi 7 octobre : l'une par M. H. Nguyen de 9h à 11h et l'autre par M. J.F. Serignat de 11h à 13h. Pourtant, ce jour là, M. J. Caelen interagit également avec Mélina pour réserver la salle Lafayette le jeudi 7 octobre, de 10h à midi. Vu que cette salle est déjà prise par M. H. Nguyen de 9h à11h et par M. J.F. Serignat de 11h à 13h, et que la priorité de M. Caelen est plus haute que celles des deux autres, il décide de demander à Mélina de négocier avec eux dans le but de lui laisser Lafayette. Mélina

prend en compte sa demande, lance le processus de négociation et fait apparaître un dialogue multisession. Le déroulement de ce dialogue est illustré par 4 sessions successives dans trois exemples. L'exemple 5.3 illustre la session initiale qui fait émerger le conflit à négocier. L'exemple 5.4 montre le processus de négociation avec les deux patients Nguyen et Serignat, et l'exemple 5.5 présente la session de notification du résultat à M. Caelen.

	Enoncés et actes de dialogue	**Stratégie et but**
S0	*FS[Ouverture(commencer)]* ici le service de réservation du laboratoire clips, je suis Mélina et vous ?	δ = directive ?Ouverture
D1	c'est jean caelen à l'appareil *FS[Prenom(jean)] & FS[Nom(caelen)]*	δ = coopérative ‡Ouverture †Usager
S1	*FS[Sexe(0)] & FS[Role(directeur)] & FS[Groupe(geod)]* bonjour monsieur le directeur ! je vous écoute...	
D2	est ce que tu peux me réserver la salle lafayette jeudi prochain de dix heures à midi *FF[Action(reserver)] & FS[NomSalle(lafayette)] & FS[Date(20041007)]&FS[Heure(600)]&FS[Duration(120)]*	δ = coopérative ‡Usager ?réservé †Salle †Date ?Conflit
S2	*FS[Erreur(DATE_ERREUR)] & FS[Conflit(2)]* désolé, cette salle est déjà réservée ! Plus précisément, la salle lafayette est déjà retenue par monsieur jean françois serignat de 11 heures à 13 heures et par monsieur hoa nguyen de 9 heures à 11 heures le jeudi 7 octobre 2004	
D3	dis leur que j'en ai besoin et qu'ils doivent me la laisser *FF[Action(negocier)]*	δ=négociée -réservé ‡Cloture
S3	*FS[Action(suspendre à négocier)]* d'accord je vais les appeler et je vous tiens au courrant...	

Exemple 5.3 : Session émergente

	Enoncés et actes de dialogue	**Stratégie et but**
S0	*FS[Ouverture(commencer)] & FFS[Patient(1)]* bonjour ici Mélina, le service de réservation du laboratoire clips ! êtes vous bien monsieur hoa nguyen ?	δ = réactive, ?Ouverture †Patient
P1	oui *FS[Patient(OK)]*	δ = réactive ‡Ouverture ‡Patient †Négociation
S1	*FFS[Negociation(x)]* monsieur jean caelen a une réunion importante à 10 heures pendant 2 heures le jeudi 7 octobre 2004. Il vous demande si vous pourriez lui laisser la salle lafayette à ce moment là ?	
P1	c'est vrai… ok j'accepte *FS[Negociation(OK)]*	δ = coopérative ‡Négociation †Date
S2	*FS[Action(demander_reporter)] & FFS[Temps(+1)]* voulez vous reporter votre réunion à un autre jour ? par exemple, le vendredi 8 octobre 2004	
P2	non réservez moi donc l'aquarium *FS[Temps(NON)]&FF[Action(reserver)]&FS[NomSalle(aquarium)]*	δ = réactive ‡Date ‡Salle
S3	*FS[Taille(20)] & FS[Date(20041007)] & FS[Heure(540)] & FS[Duration(120)]* sa capacité est de 20 places, je vous la réserve donc le jeudi 7 octobre 2004 à 9 heures pendant 2 heures	

P3	très bien… *FS[Reservation(OK)]*	δ = réactive ‡Réservé ?Convoqué
S4	*FFS[Convocation(x)]* voulez vous que je prévienne les participants ?	
P5	non merci au revoir *FS[Convocation(NON)] & FS[Cloture(x)]*	δ = réactive ‡Convoqué ‡Clôture
S5	*FS[Cloture(OK)]* au revoir monsieur nguyen à bientôt	

	Enoncés et actes de dialogue	**Stratégie et but**
S0	*FS[Ouverture(commencer)] & FFS[Patient(2)]* bonjour ici Mélina, le service de réservation du laboratoire clips ! êtes vous bien monsieur jean françois serignat ?	δ = réactive, ?Ouverture †Patient
P1	oui, c'est bien moi *FS[Patient(OK)]*	δ = réactive ‡Ouverture ‡Patient †Négociation
S1	*FFS[Negociation(x)]* monsieur jean caelen a une réunion importante de 10 heures pendant 2 heures le jeudi 7 octobre 2004. Il vous demande si vous pourriez lui laisser la salle lafayette à ce moment là?	
P2	Ah bon… d'accord mais reportez ma réunion à vendredi prochain *FS[Negociation(OK)]&FF[Action(reporter)]&FS[Date(20041008)]*	δ = coopérative ‡Négociation †Réservé
S2	*FS[NomSalle(lafayette)]&FS[Taille(45)]&FS[Heure(660)]&FS[Duration(120)]&FFS[Action(reserver)]* la salle lafayette serait à votre disposition, sa capacité est de 45 places, à 11 heures pendant 2 heures	
P3	parfait *FS[Reservation(OK)]*	δ = réactive ‡Réservé ?Clôture
S3	*FFS[Cloture(x)]* Je vous remercie pour votre générosité, j'espère que vous êtes satisfait…	
P4	à bientôt *FS[Cloture(OK)]*	δ = réactive ‡ Clôture
S4	*FS[Cloture(OK)]* au revoir monsieur serignat à bientôt	

Exemple 5.4 : Sessions du processus de négociation

	Enoncés et actes de dialogue	**Stratégie et but**
S0	*FS[Ouverture(commencer)] & FFS[Demandeur(1)]* bonjour ici Mélina, le service de réservation du laboratoire clips ! êtes vous bien monsieur jean caelen ?	δ = réactive, ?Ouverture †Demandeur
D1	Oui, c'est bien moi *FS[Demandeur(OK)]*	δ = réactive ‡Ouverture ‡Demandeur ?Notification
S1	*FFS[Notification(5)]* monsieur jean françois serignat et monsieur hoa nguyen ont accepté de vous laisser la salle lafayette à 10 heures le jeudi 7 octobre 2004. Voulez vous accepter ce résultat ?	
D2	bien sûr *FS[Notification(OK)]*	δ = coopérative ‡Notification
S2	*FS[Reservation(4)]*	

	je vous réserve la salle lafayette à 10 heures le jeudi 7 octobre 2004 pendant 2 heures	†Réservé
D3	et préviens aussi les membres du clips *FF[Annonce(x)] &FS[Membre(clips)]*	δ = réactive ‡Réservé ?Convoqué
S3	*FFS[Convocation()]* je vais les contacter par téléphone ?	
D4	par téléphone bien sûr *FS[Mode(téléphone)] & FS[Annonce(OK)]*	δ = coopérative †Convoqué
S4	*FS[Annonce(OK)]* très bien, je vais prévenir les participants	
D5	merci bien au revoir *FS[Cloture())*	‡Convoqué δ = réactive ‡Clôture
S5	*FS[Cloture(OK))* au revoir monsieur caelen à bientôt.	

Exemple 5.5 : Session de notification

5.5.4 Exemple d'un dialogue avec des incompréhensions

Nous présentons maintenant un exemple d'un dialogue oral avec Mélina en y intégrant l'interface vocale. Au cours de ce dialogue, des erreurs du module de reconnaissance de la parole provoquent donc des incompréhensions qui nécessitent de les traiter au niveau du contrôleur du dialogue. Le détail de ce dialogue est illustré dans l'exemple 5.6.

	Enoncés et actes de dialogue	**Stratégie et but**
S0	*FS[Ouverture(commencer)]* ici le service de réservation du laboratoire clips, je suis Mélina et vous ?	δ = directive, ?Ouverture
U1	**je si elle vient** <je suis nguyen> *FS[Incomprehension(MDI-1)]*	δ = directive ‡Ouverture ?Identifié
S1	*FS[Ouverture(répéter)] & FFS[Usager(x)]* ici Mélina, le service de réservation du clips ! puis je vous demander votre nom ?	
U2	nguyen *FS[Nom(nguyen)]*	δ = coopérative ‡Identifié ?Réservé
S2	*FS[Groupe(geod)] & FFS[Reservation(x)]* quelles sont vos préférences de réservation ?	
U3	j' aimerais réserver la salle lafayette *FF[Action(reservé)] & FS[NomSalle(lafayette)]*	δ = coopérative †Salle ?Temps
S3	*FS[Taille(45)] & FFS[Temps(x)]* sa capacité est de 45 places, c'est pour quel jour, s'il vous plaît ?	
U4	**est elle disponible le genre jeudi prochain** <est ce qu'elle est disponible le jeudi prochain> *FS[Date(20041014)]*	δ = coopérative ‡Salle ?Heure
S4	*FFS[Heure(x)]* à quelle heure s'il vous plaît ?	
U5	**non de dix heures à midi**	δ = coopérative

	<ben de dix heures à midi> *FS[Heure(600)] & FS[Duration(120)]*	?Matériel
S5	*FS[Reservation(2)]* je vous réserve la salle lafayette à 10 heures le jeudi 14 octobre 2004 pendant 2 heures	
U6	**j' aimerais aussi sinon vidéo projecteur** <j'aimerais aussi un vidéo projecteur> *FS[Materiel(vidéo projecteur)]*	δ = coopérative ‡Temps †Matériel †Réservé ?Convoqué
S6	*FFS[Convocation(x)]* voulez vous que je prévienne les participants ?	
U7	oui tous les membres du projet pve *FS[Convocation(OK)] & FS[Membre(pve)]*	δ = coopérative ‡Matériel ‡réservé †Convoqué ?Mode
S7	*FFS[Mode(x)]* par email ou par téléphone ?	
U8	**pas téléphone** <par téléphone> *FS[Incomprehension(MDI-1)]*	δ = directive ?Mode
S8	*FFS[Mode(téléphone)]* je vais les contacter par quel moyen, par téléphone ?	
U9	oui par téléphone *FS[Mode(téléphone)]*	δ = coopérative ‡Mode ‡Convoqué
S9	*FS[Convocation(OK)]* très bien, je vais prévenir les participants	
U10	merci bien au revoir FS[Cloture(x)]	δ = réactive ‡Cloture
S10	FS[Cloture(OK)] au revoir monsieur nguyen à bientôt.	

Exemple 5.6 : Un dialogue oral avec des incompréhensions

5.6 Evaluation

Nous présentons maintenant des résultats d'évaluation que nous avons effectués avec le système Mélina.

L'évaluation d'un système de DHM est essentielle pour réaliser un système de dialogue efficace. Dans le domaine du DHM, elle est éventuellement considérée de différentes façons : soit en effectuant un ensemble de tests et en évaluant le résultat, soit par un questionnaire que l'utilisateur remplit après avoir utilisé le système, etc. Les travaux de [Antoine et al, 2000], [Kurdi et al, 2002] portant sur la méthode DCR (Demande, Contrôle, Référence) ouvre une voie sur l'évaluation générique qui est considérée comme indépendante par rapport au modèle de la tâche.

Dans le cadre de la thèse de M. Ahafhaf [Ahafhaf, 2004], celui-ci a présenté des travaux concernant l'extension de cette méthode et des résultats obtenus en faisant des tests du système Mélina dès les premières étapes de son développement. Nous présentons dans cette section l'idée principale de cette méthode ainsi que les résultats d'évalution après avois l'appliquée.

5.6.1 DCR et l'évaluation dans un tour de parole

La méthode DCR a été présentée la première fois dans le cadre du projet européen FRACAS [Fracas, 1996]. Dans le DHM, elle se définit par la constitution d'une série de tests spécifiques destinés à tester tout « phénomène » d'un dialogue oral homme-machine, non seulement sous l'aspect linguistique et pragmatique, mais également sous l'aspect dialogique. Ce paradigme met l'accent sur les inattendus de la parole spontanée (hésitations, répétitions, etc.) et la résolution des co-références (anaphores, ellipses, déictiques). Chaque test DCR se compose de trois éléments [Ahafhaf, 2004] :

- *Déclaration (D)* : appelée aussi " Demande " contient l'ensemble des informations d'une requête de l'utilisateur (ou phrases déclaratives),
- *Contrôle (C)* : correspond à un énoncé comportant une information présente dans la requête D. L'énoncé (C) est une phrase interrogative,
- *Référence (R)* : correspond à la réponse attendue à la question de C. Elle est aussi une opération effectuée soit automatiquement par le système, soit générée manuellement par l'évaluateur. Elle consiste en une comparaison de D et C. Celle-ci est positive si les deux énoncés sont compatibles et négative dans le cas contraire. On appelle le premier *test positif* et le second *test négatif*.

Test d'hésitation
[D] : je voudrais réserver une salle **euh** avec un vidéo projecteur
FF[Action(reserver)] & FS[NomSalle(x)] & FS[Materiel(vidéo projecteur)]
[C] : réserver une salle avec un vidéo projecteur
FF[Action(reserver)] & FS[NomSalle(x)] & FS[Materiel(vidéo projecteur)]
[R] : OUI

Test de répétition
[D] : je voudrais réserver une salle **une salle** avec un **un** vidéo projecteur
FF[Action(reserver)] & FS[NomSalle(x)] & FS[Materiel(vidéo projecteur)]
[C] : réserver une salle avec un vidéo projecteur
FF[Action(reserver)] & FS[NomSalle(x)] & FS[Materiel(vidéo projecteur)]
[R] : OUI

Test d'autocorrection
[D] : j'aimerais réserver pour **jeudi prochain non mercredi prochain**
FF[Action(reserver)] & FS[Date(20041006)] (mercredi le 6 octobre 2004)
[C] : réserver pour mercredi prochain
FF[Action(reserver)] & FS[Date(20041006)] (mercredi le 6 octobre 2004)
[R] : OUI

Figure 5.17 : Exemples de tests DCR au niveau sémantique et pragmatique

La figure 5.17 montre trois exemples concernant les tests d'hésitation, de répétition et d'autocorrection en utilisant la méthode DCR au niveau sémantique et pragmatique. Nous avons également appliqué cette méthode à l'évaluation dialogique, qui envisage le dialogue sous l'ange non seulement de stratégie, mais également de but dialogique. Au cours d'un dialogue, le testeur suspend le dialogue en posant la question C portant sur les phénomènes dialogiques : soit la stratégie, soit le but actuel du système.

5.6.2 Evaluation dialogique

L'évaluation dialogique est effectivement une étape importante pour juger de la performance non seulement du modèle de dialogue utilisé, mais spécialement du système de dialogue. Dans cet exercice, nous mettons l'accent sur les évaluations de la stratégie et sur celles de dialogue multisession.

5.6.2.1 Evaluation des stratégies de dialogue

Au cours d'un dialogue, dans chaque tour de parole, le système génère une stratégie en se fondant sur les tours précédents. En appliquant la méthode DCR, nous avons fait des tests concernant la stratégie afin d'évaluer sa pertinence. Certes, suite à des incidentes, des incompréhensions, les règles de calcul de la stratégie deviennent inopérantes.

Le test DCR visant à l'évaluation stratégique est effectué au cours d'un dialogue. Cependant, la stratégie directive est toujours utilisée au début d'un dialogue en raison de la présentation du système, nous avons donc réalisé ces tests à partir du deuxième tour de parole.

A un certain tour, le testeur suspend le dialogue courant en posant une question concernant la stratégie actuelle du système et compare le résultat obtenu par rapport au résultat attendu afin de donner un jugement sur l'exactitude des calculs stratégiques. Voici un exemple de test :

```
[S] : sa capacité est de 20 places. c'est pour quel jour, s'il
    vous plaît ?
[D] : lundi prochain
[C] : est ce que cette stratégie est coopérative ?
[R]- OUI
```

Exemple 5.7 : Exemple d'un test stratégique

Du fait que la stratégie attendue dans ce tour est coopérative, ce test donne effectivement une réponse positive.

Nous avons re-utilisé le corpus de dialogue oral collecté au cours de la phase Magicien d'Oz [Fouquet, 2004] qui contient 12 dialogues déjà annotés dédiés à la tâche de réservation de salle. En partant de ce corpus, nous avons effectué des tests importants, qui se déroulent au début, au milieu et même à la fin du dialogue. Dans cette expérimentation, nous avons enregistré un nouveau corpus de 14 dialogues pertinents entre l'utilisateur et Mélina, qui comportent des dialogues multisession et des dialogues normaux. A base de notre corpus, nous avons analysé 88 stratégies de dialogue au

total, qui sont réparties en trois stratégies principales : coopérative, réactive et directive, et illustrées en tableau 5.5.

Stratégie	Nombre des stratégies correctes	Nombre des stratégies fausses	%
Coopérative	40	0	100%
Réactive	34	4	89,47%
Directive	10	0	100%

Tableau 5.5 : Distribution de stratégies de dialogue dans notre corpus

Totalement, le taux de calcul de stratégies est de : 100 - (0+4+0)*100/88 = 94,45%. La figure 5.18 montre un histogramme qui représente le résultat portant sur l'évaluation des stratégies. Dans cette figure, nous constatons que les stratégies coopératives et directives sont parfaitement calculées, à base des règles que nous avons élaborées (cf. section 3.2.4). Néanmoins, la stratégie réactive souffre des influences causées par les incompréhensions éventuelles au cours d'un dialogue (cf. section 3.3). Cela explique le résultat de 89,47% de la stratégie réactive que le système a bien calculé.

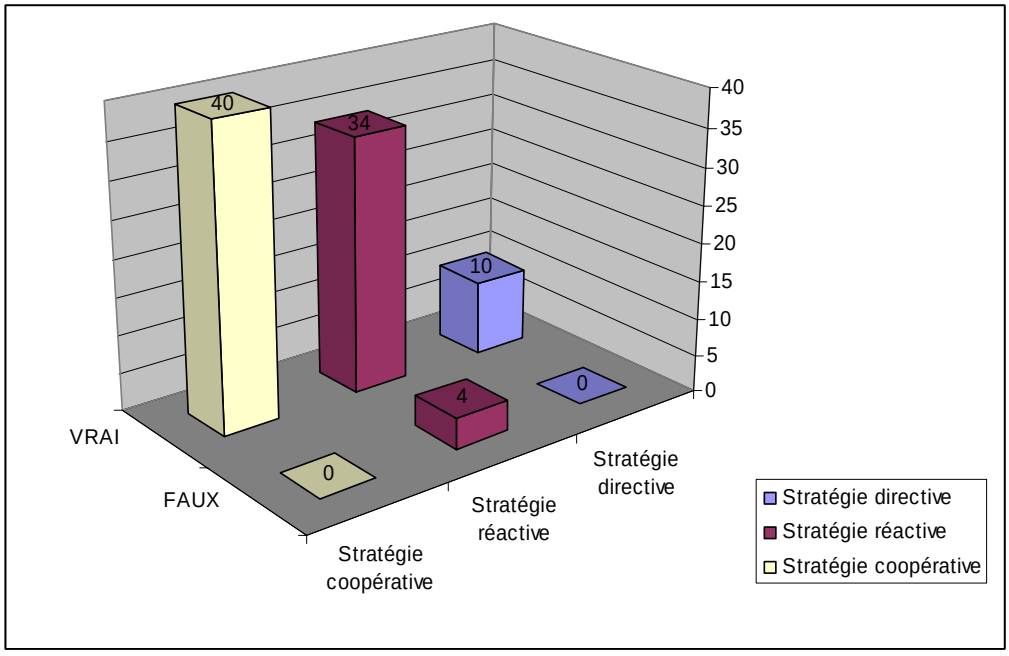


Figure 5.18 : Résultat d'évaluation stratégique

5.6.2.2 Evaluation de dialogue multisession

Le dialogue multisession est bien évidemment un dialogue qui exige une négociation et des engagements entre plusieurs utilisateurs. Un tel dialogue est évalué en tant que capacité de négociation du système, effectuée via des tests concernant le conflit apparu au cours d'un dialogue. Cette capacité dépend en effet de plusieurs éléments : le nombre de patients dans un conflit, la

solution obtenue… Nous distinguons deux types de négociations : la négociation globale qui est envisagée sous l'angle d'un dialogue multisession et la négociation intrinsèque dans chaque session du processus de négociation.

Pour l'instant, Mélina est capable de résoudre le conflit de salle et l'évaluation a donc été réalisée avec des scénarios différents concernant ce conflit. Nous effectuons deux catégories de test suivantes :

- Tests positifs de négociation : Dans ce cas là, le processus de négociation donne un bon résultat : le système atteint un chemin dans l'arbre de buts en conflit dont tous les patients manifestent leur accord portant sur la solution proposée par lui-même. Ce genre de tests peut être effectué comme pour l'exemple de dialogue multisession présenté ci-dessus.
- Tests négatifs de négociation : La demande de résoudre le conflit est prise en compte mais dans le processus de négociation, le système obtient un échec : un patient ne lâche pas sa réservation et le système ne trouve plus le chemin à suivre dans l'arbre de buts en conflit.

Envisageons maintenant l'exemple suivant : dans la base de données de Mélina, il y a eu deux réservations de la salle Aquarium, l'une par M. Y. Fouquet de 14h à 16h et l'autre par Mme S. Hollard de 16h à 18h mercredi 20 octobre 2004. Maintenant, à lundi 18 octobre, M. J.F Serignat appelle Mélina et demande la salle Aquarium au même jour de 15h à 17h. Par ordre du demandeur, Mélina lance le processus de négociation avec deux patients concernés. Néanmoins, le résultat obtenu n'est pas bon : Y. Fouquet accepte mais S. Hollard refuse le changement de sa réservation.

	Enoncés
S0	bonjour ici Mélina, le service de réservation du laboratoire clips ! êtes vous bien monsieur yannick fouquet?
P1	oui
S1	monsieur jean françois serignat a une réunion importante à 15 heures pendant 2 heures le mercredi 20 octobre 2004 . Il vous demande si vous pourriez lui laisser la salle aquarium à ce moment là ?
P2	d'accord
S2	voulez vous reporter votre réunion à un autre jour ? par exemple, le jeudi 21 octobre 2004
P3	non je veux la salle lafayette alors
S3	sa capacité est de 45 places, je vous la réserve donc le mercredi 20 octobre 2004 à 14 heures pendant 2 heures
P4	très bien
S4	voulez vous que je prévienne les participants ?
P5	non je vais le faire
S5	bon je vous laisse de prévenir les participants
P6	merci et au revoir
S6	au revoir monsieur fouquet, à bientôt.

	Enoncés
S0	bonjour ici Mélina, le service de réservation du laboratoire clips ! êtes vous bien madame Solange Hollard ?
P1	oui c'est bien moi
S1	monsieur jean françois serignat a une réunion importante à 15 heures pendant 2 heures le mercredi 20 octobre 2004 . Il vous demande si vous pourriez lui laisser la salle aquarium à ce moment là ?
P2	je regrette ce n'est pas possible
S2	pourriez vous reporter votre réunion à un autre jour ?
P3	Désolé je ne veux pas changer la date de la réunion
S3	Ce n'est pas grave, je vais lui proposer une autre solution ! au revoir madame Hollard à bientôt.

	Enoncés
S0	bonjour ici Mélina, le service de réservation du laboratoire clips ! êtes vous bien monsieur jean françois serignat ?
D1	oui
S1	monsieur yannick fouquet a accepté de laisser la salle aquarium à 15 heures le mercredi 20 octobre 2004 pour vous. Mais madame solange hollard ne voudrait pas modifié sa réservation ! Veuillez trouver une autre date s'il vous plaît ?
D2	ben reporte ma réunion à jeudi prochain de dix heures à midi alors
S2	la salle aquarium serait à votre disposition, sa capacité est de 20 places
D3	Parfait
S3	voulez vous que je prévienne les participants ?
D4	oui tous les membres de l'équipe geod
S4	par email ou par téléphone ?
D5	par téléphone bien sûr
S5	très bien, je vais prévenir les participants
D6	merci bien et au revoir
S6	au revoir monsieur serignat à bientôt.

Exemple 5.8 : Test négatif de négociation

5.7 Conclusion

Au long de ce chapitre, nous avons présenté toutes les étapes nécessaires qui constituent la mise en place de l'infrastructure générique dédiée au système de dialogue en général et à Mélina en particulier.

L'architecture de Mélina est divisée en trois couches différentes qui sont constituées par sept modules principaux, le contrôleur du dialogue jouant le rôle de coordination de toutes les activités du système et de l'utilisateur. Les principes distribués et modulaires sont également pris en

considération sous l'aspect de la conception logicielle : chaque composant doit être modulaire pour que des changements de l'un n'affectent pas les autres. Pour la même raison, ils sont distribués en tirant profit des capacités et des puissances de plusieurs machines différentes. Ces principes nous amènent donc à considérer chaque module comme un *agent* distribué contrôlé par un agent central nommé *« noyau »*. Ces caractéristiques facilitent à la fois la conception et la construction d'un système de dialogue, non seulement sous l'angle du domaine du dialogue oral homme-machine, mais également sur le terrain du génie logiciel. Tout cela constitue donc notre infrastructure générique, dédiée à la construction d'un système de dialogue.

Une infrastructure satisfaisante étant adoptée, nous avons effectué l'étape de collection de corpus ainsi que l'analyse de ces corpus, afin de construire la base de connaissances nécessaire aux divers modules dans cette infrastructure. Cette étape s'est déroulée dans le cadre du projet de recherche PVE.

Par la suite, nous avons construit au fur et à mesure tous les modules dans l'infrastructure, en nous concentrant notamment sur le contrôleur du dialogue. Avec le modèle stratégique, ainsi que la gestion générique de dialogue normal et multisession, notre contrôleur du dialogue répond parfaitement aux exigences posées : la généricité, l'extensibilité, l'adaptabilité, la réutilisabilité…. Cela lui permet donc de couvrir un ensemble de tâches différentes dans un service visé par un système de dialogue, ainsi que sa portabilité en vue d'un nouveau système de dialogue.

En intégrant l'approche de gestion de dialogue multisession au contrôleur du dialogue, Mélina est maintenant capable de résoudre certains conflits par l'intermédiaire de la délégation de pouvoir au système Mélina.

Les résultats obtenus en expérimentant Mélina nous ont permis d'atteindre un double objectif : montrer la pertinence de notre approche théorique et mettre en place le squelette d'un système de dialogue. De plus, les six grands défis que nous avons abordé au premier chapitre : « exactitude, fiabilité, flexibilité, adaptabilité, extensibilité et généricité » ont été vérifiés au cours de cette expérimentation.

CONCLUSIONS ET PERSPECTIVES

Dans la présente thèse, nous avons abordé certains problèmes dans le cadre du dialogue oral homme-machine. En effet, en envisageant le modèle stratégique de dialogue au sein du contexte d'interaction homme-machine, nous avons étudié plusieurs aspects de ce modèle, en gardant une double vision : théorique et pratique. La gestion de dialogue dans un système de dialogue a été par la suite mise en valeur en visant à la généricité d'un tel système. De là, nous avons étendu la notion de dialogue oral homme-machine sous l'angle de la négociation en considérant la multisession dans un dialogue. Et enfin, nous avons validé notre approche par l'expérimentation effectuée au cours de notre travail.

Nous présentons maintenant ci-dessous nos apports dans cette thèse.

Contributions

Les contributions que nous avons faites dans le cadre de la présente thèse sont réparties en cinq points ci-après :

I. **Spécification du modèle stratégique à une implémentation**

Partant des exigences du contrôleur du dialogue concernant les six défis mentionnés au début de cette thèse, « *exactitude, fiabilité, flexibilité, adaptabilité, extensibilité et généricité* », nous avons envisagé le modèle stratégique de dialogue proposé initialement par [Caelen, 1992], en considérant le dialogue oral homme-machine comme un jeu engageant le système de dialogue et au moins un utilisateur. Un coup de jeu est effectué par un acte de dialogue, qui représente la force illocutoire (l'aspect dialogique) et un double effet, dialogique et de la tâche, contenant dans le contenu propositionnel de l'énoncé. Le jeu de dialogue est en réalité réglé par la gestion de stratégies et un mécanisme de gestion du but de dialogue. Dans ce contexte là, la stratégie de dialogue est définie comme le moyen de gérer les tours de parole en visant à la convergence du but de dialogue, via sa meilleure direction d'ajustement. En ce qui concerne le but de dialogue, il représente un état que tous les participants (système et utilisateur) du jeu de dialogue visent à

atteindre et qui évolue au cours de ce jeu. La stratégie, le but et l'acte de dialogue constituent notamment les éléments principaux de ce modèle.

II. Gestion générique de dialogue

En partant du modèle stratégique de dialogue, nous avons élaboré la gestion générique dans un système de dialogue. Un dialogue est décomposé en différents thèmes, qui ont certaines contraintes : une fois que les participants (le système et l'utilisateur) acceptent, de façon implicite ou explicite, de jouer dans un thème donné, il leur faut accomplir ce thème avant d'attaquer un autre thème. De plus, le système doit donner des règles à l'avance (précisées dans le contrôleur de la tâche) pour préciser la dépendance, la relation et l'ordre de l'enchaînement des thèmes. Dans un thème, des règles sont également précisées, afin de bien gérer la stratégie, ainsi que le but de dialogue.

Dans un système de dialogue, le contrôleur du dialogue joue effectivement un rôle important et essentiel, afin de coordonner toutes les activités dans un système de dialogue. Il exige donc un mécanisme de gestion répondant aux critères suivants : efficacité, souplesse et généricité. L'efficacité vise essentiellement à assurer la convergence rapide de but de dialogue, tandis que la souplesse a pour but de diminuer les incompréhensions apparaissant souvent au cours d'un dialogue. La dernière exigence permet véritablement de répondre aux défis « *portabilité, adaptabilité et réutilisabilité* » du contrôleur du dialogue.

En tenant compte notamment des trois éléments principaux du modèle stratégique et des thèmes d'un dialogue, nous avons défini la notion d'« agenda de dialogue » dans le but de guider les activités du dialogue. Un agenda de dialogue est alors organisé comme un arbre hiérarchique de buts, dont le squelette représente les connaissances statiques du service visé par le système de dialogue et dont l'information stockée dans chaque élément constitue donc ses connaissances dynamiques. Les thèmes d'un dialogue sont bien évidemment spécifiés dans cet agenda.

Dans un tour de parole, avec les règles de gestion de stratégies, ainsi que le mécanisme de gestion de but de dialogue, le contrôleur du dialogue doit assumer les tâches suivantes : tout d'abord mettre à jour son agenda ; puis calculer la stratégie adéquate ; déterminer le thème et le but à résoudre ; et finalement générer les actes du système. Les trois éléments principaux (acte, but et stratégie de dialogue) sont ensuite transférés au générateur et ce tour de parole s'accomplit en attendant un autre tour de parole.

Nous avons vu que l'acte de dialogue est produit par le module d'interprétation pragmatique et il n'est pas relié, de façon explicite, au modèle de tâche dans le contrôleur du dialogue ; le but de dialogue est déterminé à l'aide de l'agenda de dialogue ; la stratégie de dialogue est calculée par le contrôleur du dialogue et elle ne concerne donc pas le modèle de tâche. Pour ces raisons, notre modèle de gestion de dialogue est évidemment générique et sa dépendance au modèle de tâche est réduite au maximum.

De plus, dans notre modèle de gestion de dialogue, le problème de traitements des erreurs concernant le système de dialogue est véritablement pris en considération. Nous avons soigneusement abordé les facteurs, qui conduisent à des erreurs du système de dialogue en les divisant en deux catégories : incompréhension et malentendu. Le malentendu est le problème le plus difficile à résoudre, non seulement par le système de dialogue, mais également en communication quotidienne entre les hommes. Pour le premier temps, nous nous concentrons alors sur le traitement des incompréhensions au niveau du contrôleur du dialogue - le « cerveau » du système de dialogue. L'incompréhension est effectivement détectée à la fois par le module de compréhension et d'interprétation en distinguant deux marqueurs dialogiques d'incompréhension : MDI-1 signifie l'incompréhension complète et MDI-2 la partielle. Des tactiques de jugement du système sur ces MDIs sont également présentées : demande de confirmation explicite et implicite, désambiguïsation, suggestion, demande de répéter, notification d'incompréhension. Par conséquent, une fois que l'incompréhension est signalée par ces marqueurs, le contrôleur du dialogue peut essayer de trouver la solution adéquate, afin d'assurer l'avancement efficace du dialogue.

III. **Négociation et dialogue multisession**

La négociation dans un système de dialogue est véritablement un des grands intérêts de notre travail. Notre approche portant sur ce terrain part du fait qu'un conflit entre les utilisateurs peut être résolu via des négociations importantes entre le système et chacun d'entre eux. De là, le dialogue multisession a été proposé en traitant le cas dans lequel le système joue le rôle intermédiaire, pour résoudre le conflit entre deux ou plusieurs utilisateurs concernant les ressources gérées par le système. Nous avons modélisé un dialogue multisession en le distinguant en trois phases différentes. La première phase, dite *Emergence,* représente la session entre un utilisateur (appelé également *demandeur*) et le système, avec l'apparition de conflit entre son but et un ou plusieurs buts déjà satisfaits, manifestés par les autres utilisateurs (dit *patients*). La deuxième phase représente le *processus de négociation* contenant plusieurs sessions différentes entre le système avec ces patients, afin de trouver un compromis favorable au demandeur. La dernière phase, dite *Notification*, contient une session unique entre le système et le demandeur, dans le but de informer celui-ci du résultat de la négociation.

En ce qui concerne la gestion de dialogue multisession, nous avons élaboré un module, dit **Expert**, dans le contrôleur du dialogue. Son rôle principal est de gérer l'arbre de buts en conflit avec un double objectif : trouver le meilleur chemin non encore parcouru dans cet arbre et planifier la négociation avec chaque patient dans ce chemin, de manière efficace et rapide. La gestion d'arbre de buts en conflit, ainsi que l'enchaînement des sessions dans le dialogue multisession, ont été explicitement présentés. De plus, nous avons également précisé la stratégie de négociation, de même que les règles d'initiation d'une session de négociation, en vue de mettre l'accent sur la souplesse au cours du processus de négociation.

IV. **Mélina – un véritable système de dialogue oral homme-machine**

Dans l'intention de conforter notre approche théorique, nous avons construit une infrastructure de système de dialogue. Partant d'une architecture modulaire, distribuée et multi-agent, divisée en trois couches avec sept modules principaux : celui de la reconnaissance de la parole, celui de la compréhension sémantique, l'interpréteur pragmatique, le contrôleur du dialogue, le contrôleur de la tâche, le générateur textuel et le synthétiseur de la parole, cette infrastructure organise effectivement ses modules sous forme d'agents communicants dont l'interaction est effectuée via des messages.

En se fondant sur cette infrastructure, ainsi que nos contributions théoriques, dans le cadre du projet PVE, nous avons développé un système de dialogue dédié au service d'organisation de réunion, nommé Mélina. En effet, après l'analyse d'usage et de corpus collecté au cours de la phase de Magicien d'Oz, tous les modules dans cette infrastructure ont été implémentés, notamment le contrôleur du dialogue équipé d'un *Expert* dédié à la gestion de négociations. Avec la gestion générique de dialogue, Mélina offre effectivement des dialogues souples avec l'utilisateur, tolère des erreurs provenant du module de reconnaissance de la parole, etc. De plus, en cas de conflits, l'utilisateur peut demander à Mélina de négocier avec les utilisateurs concernés, afin de trouver un bon compromis entre eux. L'expérimentation que nous avons faite autour de Mélina a donné des premiers résultats très encourageants, validant et prouvant l'intérêt de nos approches théoriques.

Perspectives

Du point de vue du dialogue, nous espérons dans un premier temps pouvoir offrir une meilleure capacité de négociation dans un système de dialogue. L'amélioration, ainsi que le renforcement des rôles de l'Expert, devraient effectivement être pris en considération, pour que le système dialogue puisse négocier efficacement avec un utilisateur de façon naturelle.

Un autre aspect de la négociation serait également intéressant à envisager dans le futur : la coordination des négociations entre plusieurs participants et le système simultanément. Dans ce cas là, plusieurs utilisateurs et le système seront engagés dans un dialogue. Il est certain que la gestion de ce dialogue sera très complexe, avec des problèmes difficiles à résoudre comme l'identification du locuteur (afin de distinguer qui parle), la compréhension simultanée (deux ou plusieurs utilisateurs parlent d'une chose en même temps), etc.

De plus, nous pensons à l'intérêt de combiner la parole avec d'autres modalités d'interaction, comme la vision dans un système de dialogue. En effet, au point de vue dialogique, la vision apporte véritablement des données complémentaires nécessaires via les gestes, les attitudes de visage, etc., de l'utilisateur. Ces données enrichissent évidemment la capacité de compréhension sémantique, ainsi que d'interprétation pragmatique du système de dialogue.

En termes de robustesse du système de dialogue, nous proposons d'étendre le rôle du contrôleur du dialogue au traitement des malentendus. La détection, ainsi que le diagnostic de

malentendus, devraient être pris en considération, afin de trouver la solution adéquate destinée à les éviter. Ainsi, l'amélioration du traitement des incompréhensions, au niveau du contrôleur du dialogue, sera également à envisager, afin de donner davantage de souplesse au système de dialogue.

L'amélioration des modules autres que le contrôleur du dialogue dans notre infrastructure pourrait également être une autre perspective. Au sein de notre équipe, la compréhension sémantique robuste est en cours de recherche (la thèse en cours de Ana-Patricia Dominguez-Sanchez). De plus, les résultats concernant l'interprétation pragmatique se fondant sur la théorie SDRT (travaux en cours de Anne Xuereb) amélioreront nettement la capacité d'interprétation en général pour le système de dialogue et le système Mélina en particulier.

Potentiels d'application industrielle

En tirant profit de toutes nos contributions, l'extension de Mélina vers des nouveaux services est tout à fait envisageable. Nous citons dans cette section quelques potentiels de réaliser cela ci-dessous :

- Portail vocal d'entreprise : Ne pas se limiter au service d'organisation de réunion, Mélina peut évidemment s'orienter vers tous les services qui peuvent être gérés par le portail vocal d'entreprise abordé dans le projet PVE : gestion de l'agenda de tous les personnels, redirection d'appels, renseignement… La différence entre les personnels distants et ceux étant sur place aura tendance à diminuer grâce aux services d'un tel portail.
- Secrétaire virtuel : En profitant des points forts provenant d'autres modalités d'interaction, nous pourrons construire un avatar animé qui assurera les tâches de secrétariat pour un personnel, dans le but d'ignorer les points gênants d'un téléphone. En utilisant un casque-micro avec une Webcam, l'utilisateur interagira probablement avec son secrétaire personnel virtuel par la parole, des mimes et des gestes pour réaliser des différentes tâches comme : consulter/envoyer le courriel électronique, gérer son agenda personnel, effectuer un appel téléphonique, rédiger un document, etc.
- Robot communicant : L'intégration d'un système de dialogue à un robot permet effectivement de lui apporter la parole naturelle. En effet, adapter Mélina à un nouveau service visé par un robot amène donc la possibilité de le mettre en place dans un unique ordinateur. Par la suite, nous pourrons intégrer ce petit ordinateur au sein d'un robot et celui-ci serait capable de parler, de dialoguer avec les hommes.

BIBLIOGRAPHIES

Publications personnelles

[Caelen et Nguyen, 2005] J. Caelen, H. Nguyen, « *Vers un dialogue collectif avec la machine* », soumis aux Troisièmes Journées Francophones sur les « Modèles Formels de l'Interaction », Caen, mai 2005.

[Nguyen, 2005] H. Nguyen, « *MELINA – Un agent conversationnel multisession* », accepté à la conférence internationale en informatique « Recherche, Innovation & Vision du futur », février 2005, Vietnam.

[Nguyen et Caelen, 2004] H. Nguyen, J. Caelen, « *Multi-session management in spoken dialogue system* », The IX Ibero-American Conference on Artificial Intelligence (IBERAMIA), Puebla - Mexico, Novembre 2004.

[Caelen et Nguyen, 2004] H. Nguyen, J. Caelen, « *Gestion de buts de dialogue* », 11ème conférence TALN (Traitement Automatique des Langues Naturelles), Fès – Maroc, 19-22 avril 2004, pp.345-350.

[Nguyen et Caelen 2003] H. Nguyen et J. Caelen, *"Generic manager for spoken dialogue systems",* presented at DIABRUCK 2003: 7th Workshop on the Semantics and Pragmatics of Dialogue, Saarbrucken - Deutchland, September 4-6, 2003.

[Nguyen 2003a] H. Nguyen, "*Gestion générique du système de dialogue oral homme-machine*", Rencontres Jeunes Chercheurs en parole RJC, Grenoble - France, 23-25 septembre, 2003.

[Nguyen 2003b] H. Nguyen, *"Vers une architecture générique de système de dialogue oral homme-machine",* 10ème conférence TALN/RECITAL (Traitement Automatique des Langues Naturelles), Batz sur Mer, 11-14 juin 2003, pp. 539-545.

Références bibliographiques

[Adam et Reynaud, 1978] G. Adam, J.D. Reynaud, *Conflits du travail et changement social*, colloque "Sociologies", ed. PUF, Paris, 1978, 389 p.

[Ahafhaf, 2004] Mohamed Ahafhaf, « Evaluation des systèmes de dialogue orale homme-machine : Quelques éléments linguistiques appliqués au paradigme DCR », Thèse de l'Université Stendhal - Grenoble 3, 2004.

[Allen et Perrault, 1980] Allen, J., Perrault, C., *Analysing intention utterances*. Artificial intelligence, V°15, p143-178, 1980.

[Asher, 1996] N. Asher. « L'interface pragmatique-sémantique et l'interprétation du discours ». *Revue langage n° 163*. Septembre 1996.

[Asher et Lascarides, 2002] Asher N., Lascarides A., *Logics of conversation*. Cambridge University press, 2002.

[Baggia et al., 1999] Baggia P., Gauvain J.L., Kellner A., Perennou G., Popovici C., Sturm J., Wessel F., "Language Modelling and Spoken Dialogue Systems - the ARISE experience", in Proc. *Sixth European Conference on Speech Communication and Technology*, September 1999.

[Baker, 1994] Michaël Baker, "A Model for negotiation in Teaching-Learning Dialogues", In : Journal of Artificial Intelligence in Education n°5(2), pp. 199-254, 1994.

[Baker, 1999] Baker, M., *Argumentative interactions and cooperative learning*. Escritos. 1999.

[Bilange, 1991] E. Bilange, "An approach to oral dialogue modelling", in *Proceeding of Second Venaco Workshop on the structure of multimodal dialogue*, Acquafredda di Maratea, pp. 1-12, 1991.

[Bilange, 1992] Bilange E., *Dialogue personne-machine, modélisation et réalisation informatique*, Langue, Raisonnement, Calcul, Hermès, Paris, 1992.

[Bonnet, 1980] A. Bonnet, « Les grammaires sémantiques, outils puissants pour interroger les base de données en langage naturel », RAIRO Informatique, vol. 14, n°2, pp. 137-148, 1980.

[Bourque et Thuderoz, 2002] Reynald Bourque, Christiant Thuderoz, *Sociologie de la négociation*, Edition La Découverte & Syros, Paris, 2002.

[Bousquet-Vernhettes, 2002] Caroline Bousquet-Vernhettes, « Compréhension robuste de la parole spontanée dans le dialogue oral homme-machine – Décodage conceptuel stochastique », Thèse informatique de l'Université Toulouse III, 2002.

[Bresnan, 1982] J. Bresnan, "*The mental representation of grammatical relations*", MIT Press, Cambridge, MA, ISBN 0-262-02158-7, 1982.

[Brian, 1980] Chellas Brian, *Modal Logic: An Introduction*, Cambridge University Press, 316 p., ISBN: 0521295157, 1980.

[Bruce, 1975] B. Bruce, "Case Systems for Natural Language", Artificial Intelligence, vol. 6, pp. 327-360, 1975.

[Brun 1998] Brun, C., "A Terminology Finite-State Preprocessing for Computational LFG". 36th International meeting of the Association for Computational Linguistics & 17th International Conference on Computational Linguistics, Montreal, Quebec, Canada, August 1998.

[Caelen, 1992] Jean Caelen, « Attitudes cognitives et actes de langage ». In *Du dialogue, Recherches sur la philosophie du langage*, n° 14, Vrin éd., Paris, p. 19-48, 1992.

[Caelen, 1997] Caelen J.(1997), *Interaction Verbale*, Editions CEPADUES, 1997.

[Caelen et al, 1997] J. Caelen, J. Zeiliger, M. Bessac, J. Siroux, G. Perennou, « Les corpus pour l'évaluation du dialogue homme-machine. », JST Francil 1997 proceedings, Aupelf-Uref, April 15-16, 1997, University of Avignon (F)

[Caelen 2002] Caelen, J., « Modèles formels de dialogue », Actes des 2èmes assises du *GdR I3, Information, Interaction Intelligence*, CEPADUES Editions, p. 31-58, 2002.

[Caelen, 2003] Caelen, J. , « Stratégies de dialogue », Conférence *MFI'03 (Modèles Formels de l'Interaction)*, Lille, CEPADUES éd, 2003.

[Carberry, 1990] S. Carberry, *"Plan recognition in natural language dialogue",* MIT Press, 1990.

[Carpenter et al., 2001] Carpenter, P., Jin, C., Wilson, D., Zhang, R., Bohus, D., Rudnicky, A.,"*Is This Conversation on Track?"*, in Proceedings of Eurospeech 2001, Aalborg, Denmark, 2001.

[Chevallier, 1992] Robert Chevallier, « *Mise en oeuvre d'un modèle dynamique de dialogue dans un Tuteur Intelligent* », Thèse de l'Université du Mans, 1992.

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

[Cohen et Levesque, 1990a] Cohen, P., Levesque, H., “Persistence, intention and commitment”. p.33-69 dans Cohen, P., Morgan, J., & Pollack, M. (eds), *Intentions in communication*. Cambridge : MIT Press, 1990.

[Cohen et Levesque, 1990b] Cohen, P., & Levesque, H. “Rational interaction as the basis for communication”. p221-255 of : Cohen, P., Morgan, J., & Pollack, M. (eds), *Intentions in communication*. Cambridge : MIT Press, 1990.

[Cohen et Levesque, 1991] Cohen, R., Levesque, H. *Teamwork*. Nous, 25(4), 487-512, 1991.

[Cormen et al., 2002] T. Cormen, C. Leiserson, R. Rivest, C. Stein, *Introduction à l'algorithmique*, ISBN : 2-10-003922-9, Editeur Dunod, 2e édition 2002.

[Coutaz, 1990] J. Coutaz, *Interface Homme-Ordinateur*, Ed. Dunod, 1990.

[Davis et Mermelstein, 1980] Davis S., Mermelstein P., *Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences*, IEEE Transactions ASSP, ASSP-28(4), pp. 357-366, 1980.

[Davies et al., 2001] Davies, J.R.; Gertner, A.; Lesh, N.B.; Rich, C.; Rickel, J.; Sidner, C.L., "Incorporating Tutorial Strategies into an Intelligent Assistant", *ACM International Conference on Intelligent User Interfaces*, ISBN: 1-58113-325-1, pps 53-56, January 2001.

[De Mori, 1998] Renato De Mori, *Spoken dialogues with Computers*, Academic Press, Inc., Orlando, FL, 702p., 1998.

[Demazeau, 2003] Yves Demazeau, « Multi-Agent System », support du cours DEA “Système multi-agents, 2003.

[Ferguson et al., 1998] Ferguson, G., Allen, J., *“TRIPS: An Intelligent Integrated Problem-Solving Assistant”*, in Proceedings of AAAI-98, Madison, WI, July 1998.

[Fillmore, 1968] Ch.J Fillmore, *The case for case*, in Universals in Linguitic Theory, Bach Emmon et Harms Robert T., pp. 1-90, NewYork, 1968.

[Fouquet, 2003] Yannick Fouquet, « Le Magicien d’Oz pour dialogue oral : expérience avec un assistant virtuel en entreprise », Actes de Rencontre des Jeunes chercheurs en Parole, Grenoble, septembre 2003.

[Fouquet, 2004] Yannick Fouquet, « Modélisation des attentes en dialogue oral », thèse de doctorat informatique de l’université Joseph Fourier, 2004.

[Fracas, 1996] FRACAS Consortium, 1996. *Using the framework*. Fracas project LRE 62-051. Deliverable D16, chap. 3.

[Gaussier et al., 1997] Gaussier, E., Grefenstette, G., Schulze, M., « Traitement du langage naturel et recherche d' informations : quelques expériences sur le français ». *Premières Journées Scientifiques et Techniques du Réseau Francophone de l' Ingénierie de la Langue de l'AUPELF-UREF*, Avignon, Avril 1997.

[Girard, 2000] Patrick Girard, « Ingénierie des systèmes interactifs : vers des méthodes formelles intégrant l'utilisateur », Thèse, Habilitation à diriger les recherches, Université de Poitiers, 2000.

[Grice, 1975] Grice, H.-P. *Logic and conversation*. In P. Cole (ed.) Syntax and Semantics. Vol. 3. New York: Academic Press. 41-58, 1975.

[Grosz et Sidner, 1990] Grosz, B. G., Sidner, C. L. "Plans for discourse". Pages 417-444 of : Cohen, Philip R., Morgan, Jerry, & Pollack, Martha E. (eds), *Intentions in communication*. Cambridge : MIT Press.

[Grosz et Kraus, 1996] Barbara Grosz and Sarit Kraus. "*Collaborative Plans for Complex Group Action*" In Artificial Intelligence. 86(2), pp. 269-357.1996.

[Hammouche, 1995] Hammouche H., « De la modélisation des tâches utilisateurs au prototype de l'interface homme-machine ». Thèse de Doctorat – Université Paris 7, 1995.

[Harel, 1979] Harel, D. 1979. *First-order dynamic logic*. New York : Springer-verlag.

[Hazen et al., 2002] Hazen, T-J., Burianek, T., Polifroni, J., Seneff, S., "Recognition Confidence Scoring for Use in Speech Understanding Systems", in Computer Speech and Language, vol. 16, no. 1, pp 49-67, January, 2002.

[Hobbs, 1990] Hobbs, J. "Artificial intelligence and collective intentionality". pp.445-459 of : Cohen, Philip R., Morgan, Jerry, & Pollack, Martha E. (eds), *Intentions in communication*. Cambridge : MIT Press, 1990.

[Hoc, 1987] Jean-Michel Hoc, *Psychologie cognitive de la planification*, Ed. Presses Universitaire de Grenoble, 1987.

[Hunt, 2000] Andrew Hunt, *JSpeech Grammar Format*», W3C Note, June, 2000. http://www.w3.org/TR/jsgf/

[Jacobson, 2000] Jacobson M., « Les outils modernes pour la notation de corpus de parole », dans la Colloque transcription de la parole normale et pathologique, Tours, 2000.

[Jelinek, 2001] Jelinek F., Aspects of the Statistical Approach to Speech Recognition. In proceedings of *IEEE International Symposium on Information Theory*, 06-2001.

[Larsson et Traum, 2000] Larsson, S. and Traum, D., "Information state and dialogue management in the TRINDI dialogue move engine toolkit", Natural Language Engineering 6, 323–340, 2000.

[Lea, 1980] W.A Lea, "The value of speech recognition systems, Trends in speech recognisition", Prentice Hall Signal Processing Series, Saddle River, NJ, 1980.

[Lehuen et al, 1996] Lehuen, J., Nicolle, A. & Luzzati, D., « Un modèle hypothético-expérimental dynamique pour la gestion des dialogues homme-machine », RFIA, Rennes, éditeur Hermès Paris, 1996

[Lehuen, 1997] Lehuen J., « Un modèle de dialogue dynamique et générique intégrant l'acquisition de sa compétence linguistique - Le système COALA », thèse du laboratoire d'Informatique, Université du Maine, 1997.

[Lemon et al., 2002] Lemon, O., Gruenstein, A., Cavedon, L., Peters, S., "Multi-Tasking and Collaborative Activities in Dialogue Systems", in Proceedings of SIGDIAL-2002, pp. 113-124, Philadelphia, PA, 2002.

[Levin et Moore, 1980] J. Levin, J. Moore, "Dialogue Games: meta-communication structure for natural language interaction". Cognitive sciences, N°1, p.395-420, 1980.

[Levinson, 1979] Levinson, S. C. "Activity types and language". Linguistics, N°17, 1979.

[Lewin, 2000] Lewin, I. "A formal model of conversational game theory". In : Proceedings of the 4th workshop on the semantics and pragmatics of dialogue (gotalogoo), 2000.

[Litman et Allen, 1990] Litman, D.J., Allen, J.F. "Discourse processing and commonsense plans". p. 365-388 of Cohen, P. R., Morgan, J., & Pollack, M. E. (eds), *Intentions in communication*. Cambridge : MIT Press, 1990.

[Lochbaum, 1994] Lochbaum, K. E., "Using collaborative plans to model the intentional structure of discourse". Ph.D. thesis, Harvard University, Cambridge, 1994.

[Luzzati, 1995] Daniel Luzzati (1995), *Le dialogue verbal homme-machine*, Masson, Paris.

[Mann, 1988] Mann, W.C., « Dialogue games: conventions of human interaction », Argumentation 2, pp511–532, 1988.

[Maudet, 2001] Nicolas Maudet, « Modéliser les conventions des interactions langagières : la contribution des jeux de dialogue », Thèse informatique de l'Université Paul-Sabatier, Toulouse 3, 2001.

[McTear, 2002] Micheal F. McTear, “Spoken Dialogue Technology: Enabling the Conversational User Interface”, ACM Computing Surveys, Volume 34 , Issue 1 (March 2002), pp. 90 – 169.

[Miège et al., 2003] B. Miège, J. Caelen, V. Chanal, M. Favier, P. Pajon, J. Rouault, F. Séguy, J. Trahand, D. Vernant, *Communication personnes systèmes informationnels*, ISBN 2-7462-0680-3, Ed. Lavoisier – Hermes, 2003.

[Minker et Bennacef, 2001] W. Minker, S. Bennacef, *Parole et dialogue home-machine*, Eds Eyrolles, CNRS, 2001.

[Mitre, 2002] The DARPA Communication Team, « Galaxy Communicator Documentation », 2002.

[Mœschler, 1989] Jacques Mœschler (1989), *Modélisation du dialogue*, Hermès, Paris.

[Morel, 1989] Mary-Annick Morel, *Analyse linguistique d’un corpus de dialogues hommemachine. Premier corpus : centre de renseignements SNCF à Paris* , Publications de la Sorbonne Nouvelle, 1989.

[Myerson, 1991] R. B. Myerson. *Game Theory: Analysis of Conflict*. Harvard University Press, 1991.

[Néel et Minker, 1999] F. Néel, W. Minker « Computational Models of Speech Pattern Processing», Springer Verlag, Berlin/Heidelberg (Germany), 1999.

[Nerzic, 1993] P. Nerzic. « Erreurs et échec dans le dialogue oral homme-machine: détection et réparation ». Thèse de doctorat de l’univeristé de Rennes, 1993.

[Nicolle 1997] Anne Nicolle et Jean Vivier (1997), "Apprentissage et Dialogue humain/humain, humain/machine, machine/machine", In : Actes du congrès CAPS’97 sur l’apprentissage personne/système, Caen, Europia Production, Paris.

[Nuanc, 2001] Nuance Speech Recognition System v7.0, « Grammar Developper’s Guide », Nuance, 2001.

[Ogden et Bernick, 1996] Ogden, W.C. et Bernick, P. “*Using Natural Language Interfaces*”. Computing Research Laboratory, Box 30001/3CRL, New Mexico State University, Las Cruces, New Mexico 88003, May 1996.

[Pallett et al., 1995] D.S. Pallett, J.G. Fiscus, W.M. Fisher, and J.S. Garofolo, "1994 Benchmark Tests for the DARPA Spoken Language Program.", DARPA Workshop Spoken Language System Tech, 1995.

[Pérennou, 1996] Guy Pérennou, « Compréhension du dialogue oral – Le rôle du lexique dans l'approche par segments conceptuels », Lexique et communication parlée - GDR PRC, p. 169-178, 1996.

[Phoenix, 2002] "The Phoenix parser manual", The Center for Spoken Language Research University of Colorado, 2002.
http://communicator.colorado.edu/phoenix/Phoenix_Manual.pdf

[Pollack, 1990] Pollack, M. E. "Plans as complex mental attitudes". Pages 77-104 of : Cohen, Philip R., Morgan, Jerry, & Pollack, Martha E. (eds), *Intentions in communication*. Cambridge : MIT Press.

[Rabiner et Juang, 1989] Rabiner L. R., Juang B., *Tutorial on hidden Markov models and selected applications in speech recognition*, Proceedings of the IEEE, vol. 77, no. 2, pp. 257-285, 1989.

[Rickel et al., 2002] Rickel, J.; Lesh, N.B.; Rich, C.; Sidner, C.L.; Gertner, A., "*Collaborative Discourse Theory as a Foundation for Tutorial Dialogue*", *International Conference on Intelligent Tutoring Systems (ITS)*, Vol 2363, pps 542-551, June 2002 (Lecture Notes in Computer Science)

[Rickel et al., 2001] Jeff Rickel, Neal Lesh, Charles Rich, Candace L. Sidner, and Abigail Gertner. "*Using a model of collaborative dialogue to teach procedural tasks*". Workshop on Tutorial Dialog Systems, AIED - 2001.

[Rouault et Manes-Callo, 2003] Jacques Rouault, Maria-Caterina Manes-Gallo, *Interlligence linguistique – le calcul du sens des énoncés élémentaires*, Ed. Hermes-Lavoisier, 2003.

[Rouillard, 2000] Rouillard J. , « Hyperdialogue sur Internet – Le système HALPIN », thèse de doctorat de l'Université Joseph Fourier, CLIPS-IMAG, 2000.

[Roulet, 1985] Eddy Roulet, Antoine, *L'articulation du discours en français contemporain*, Ed. Peter Lang, Berne, 1985.

[Rudnicky et al., 1999] Rudnicky, A., Thayer, E., Constantinides, P., Tchou, C., Stern, R., Lenzo, K., Xu, W., Oh, A., 1999 – "*Creating natural dialogs in the Carnegie Mellon Communicator System*", in Proceedings of Eurospeech, 1999, pp 1531-1534

[Sabah, 1988] G. Sabah, *L'intelligence Artificielle et le langage*, Vol. 1, Représentation des connaissances, Ed. Hermès, 1988.

[Sabah, 1989] G. Sabah, *L'intelligence Artificielle et le langage*, Vol. 2, Processus de compréhension, Ed. Hermès, 1989.

[Schmandt, 1994] C. Schmandt, “Voice Communication with Computer”, Van Nostrand Reinhold, New York, 1994.

[Seneff, 1992] S. Seneff, "TINA: A natural language system for spoken language applications," *Computational Linguistics*, vol. 18, no. 1., pp. 61-86, March 1992.

[Sidner et al., 2003] Sidner, C.L.; Lee, C.H.; Lesh, N., "The Role of Dialog in Human Robot Interaction", *International Workshop on Language Understanding and Agents for Real World Interaction*, July, 2003.

[Singh, 1998] Singh, M.P. “An ontology for commitments in multi-agent systems: towards a unification of normative concepts”. *Artificial intelligence and law*, 1998.

[Sowa 1988] J. Sowa, “*Using a lexicon of canonical graphs in a semantic interpreter. Relational Models of Lexicon”*, Cambridge, Evens, 1988.

[Sperber et Wilson, 1986] D. Sperber et D. Wilson. *Relevance: Communication and cognition*. Basil Blackwell Ed., Oxford, 1986.

[Sperber, 2000] D. Sperber. *La communication et le sens*. Dans Yves Michaud (ed.) “Qu'est-ce que l'humain? Université de tous les savoirs”, volume 2. (Paris: Odile Jacob, 2000. 119-128). http://www.dan.sperber.com/

[Swartout et al., 2001] Swartout, W., Hill, R., Gratch, J., Johnson, W., Kyriakakis, C., Labore, K., Lindheim, R., Marsella, S., Miraglia, D., Moore, B., Morie, J., Rickel, J., Thiebaux, M., Tuch, L., Whitney, R. and Douglas, J., “*Toward the holodeck: Integrating graphics, sound, character and story*”, Proceedings of 5th International Conference on Autonomous Agents, 2001.

[Thórisson, 1995] Thórisson K.R. “Multimodal interface agents and the architecture psychosocial dialogue skills”, thèse de philosophie, Massachusetts, 1995.

[Traum, 2002] David R. Traum, “*Ideas on Multi-layer Dialogue Management for Multi-party, Multi-conversation, Multi-modal Communication*”: Extended Abstract of Invited Talk, in Computational Linguistics in the Netherlands: Selected Papers from the Twelth CLIN Meeting , Editions RODOPI B.V., pages 1-7, 2002.

[Traum et al, 2003] David Traum, Jeff Rickel, Jonathan Gratch, and Stacy Marsella, “*Negotiation over Tasks in Hybrid Human-Agent Teams for Simulation-Based Training*”, in proceedings of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems, Melbourne, Australia, pp. 441-448, July, 2003.

[Traum, 2004a] David R. Traum, Susan Robinson, Jens Stephan, “*Evaluation of multi-party virtual reality dialogue interaction*”, in LREC, 2004.

[Traum, 2004b] David Traum, "*Issues in multi-party dialogues*", in Advances in Agent Communication Ed. F. Dignum, Springer-Verlag LNAI 2922 pp. 201-211, 2004.

[Turunen, 2004] Markku Turunen, "Jasis - A Spoken Dialogue Architecture and its Applications", thèse informatique de l'Université de Tampere, 2004.

[Vanderveken, 1988] Vanderveken, D., « Les actes de discours », Perre Margada éd., Bruxelles, 1988.

[Vanderveken, 1990] D. Vanderveken, *La logique illocutoire*. Mandarga éd. Bruxelles, 1990.

[Vanderveken, 1997] D. Vanderveken, « La logique illocutoire et l'analyse de discours », in D. Luzzati et al (eds), *Le dialogique*, Peter Lang, 1997.

[Vanderveken, 1999] Vanderveken, D. « La structure logique des dialogues intelligents ». p. 61-100 paru dans Moulin, B., Delisle, S., & Chaib-draa, B. (eds), *Analyse et simulation de conversations : de la théorie des actes de discours aux systèmes multi-agents*. L'interdisciplinaire informatique(s). 1999.

[Vernant, 2003] Denis Vernant, « Communication interpersonnelle et communication personne-système », dans B. Miège, *Communication personnes systèmes informationnels*, Ed. Lavoisier – Hermes, 2003.

[Vonneumann et Morgenstern, 1967] J. Von NEUMANN , O. MORGENSTERN, *Theory of games and economic behavior*, Princeton University Press, 1967.

[Walker et al., 2000] Walker, M., Wright, J., Langkilde, I., 2000 – *"Using Natural Language Processing and Discourse Features to Identify Understanding Errors in a Spoken Dialogue System"*, in Proceeding of the 17'th International Conference of Machine Learning, pp 1111-1118, 2000.

[Wang et al., 2000] Y. Wang, M. Mahajan, X. Huang, *A Unified Context-Free Grammar and N-Gram Model for Spoken Language Processing*. In proceeding s of the International Conference on Acoustics, Speech and Signal Processing – ICASSP 2000.pp 1639-1642.

[Ward, 1994] Ward W., "*Extracting Information From Spontaneous Speech"*. In proceesdings of *International Conference on Spoken Language Processing*, September 1994

[Ward, 1991] Ward W., Understanding Spontaneous Speech: The Phoenix System. In Proceedings of *IEEE International Conference on Acousitcs, Speech and Signal Processing* (ICASSP'91), 1991, pp 365-367.

[Wilensky, 1983] Robert Wilensky, *Planning and understanding : a computational approach to human reasoning*, Reading, Addison-Wesley, 1983.

[Wittgenstein, 1957] Wittgenstein, L. *Philosophical investigations*. Oxford : Blackwell, 1957.

[Woods, 1970] W. A. Woods, "Transition Network Grammars for Natural Language Analysis", *Communications of the ACM,* vol. 13, n°10, pp. 591-606, 10-1970.

[Xu et Rudnicky, 2000] Xu, W. and Rudnicky, A. "Task-based dialog management using an agenda". *ANLP/NAACL 2000 Workshop on Conversational Systems*, pp. 42-47,2000.

[Xuereb et Caelen, 2004] Xuereb A., Caelen J., « Un modèle d'interprétation pragmatique en dialogue homme-machine basé sur la SDRT ». *Workshop SDRT, TALN-04*, Fès, 22 Avril 2004.

[Yankelovich et al., 1995] Yankelovich, N., Levow, G.A., Marx, M., 1995 – *"Designing SpeechActs: Issues in Speech User Interfaces"*, in Proceedings of CHI'95, pp. 369-376, 1995.

[Zue 97] Zue, V., "Conversational Interfaces: Advances and Challenges", KN speech, Eurospeech, Rhodes Greece, 1997.

ANNEXES

Annexe A : Service de réservation de salle en VoiceXML

A.1 Code source élaboré dans IBM VoiceToolkit

```
< ?xml version= « 1.0 » encoding= « iso-8859-1 » ?>
< !DOCTYPE vxml PUBLIC « vxml » «  »>
<vxml version= « 1.0 »>
     < !—paramètres -->
     <var name=”date” />
     <var name=”heure”/>
     <var name=”taille”/>

     <form id= »frmOuverture »>
       <block>
       Bonjour, Vous êtes en connexion avec le service de réservation de
la salle de réunion.
       </block>
       <field name=”tache_racine”>
             <grammar src= «reservation.jsgf» type= «application/x-jsgf»/>
             <prompt>Je vous écoute !</prompt>
             <prompt count= «2»>Bonjour, bienvenue au service de
réservation de la salle de réunion. Vous pouvez parler pour réserver la
salle à votre réunion</prompt>

             <catch event=”politesse help”>
                 <reprompt/>
             </catch>
             <catch event=”imcomprehension”>
                 Je m’occupe du service de réservation de la salle de
réunion. Que puis-je faire pour vous ?
             </catch>

             <noinput>
                   Je n’entends rien.
                   <throw event=”politesse”/>
             </noinput>

             <nomatch>
                   Désolé, je ne comprends pas ce que vous dites.
                   <throw event=”imcomprehension”/>
             </nomatch>

             <filled>
                   <if cond=”tache racine == ‘poli’” >
                         <clear namelist=”tache_racine”/>
```

```
                        <throw event="politesse"/>
                  <else/>
                        <goto next="#frmEchange01"/>
                  </if>
            </filled>

      </field>

    </form>

    <form id="frmEchange01">

      <field name="fdate" type="date">

            <prompt> »Pour quel jour, s'il vous plait »</prompt>
            <nomatch>
                  Excusez-moi, ce que vous dites est invalide. S'il vous
plait de parler un jour correct.
            </nomatch>

            <noinput count="1">
                  J'ai besoin la date de votre réunion. Présicez votre
date s'il vous plait ?
            </noinput>

            <noinput count="2">
                  Il faut préciser la date de réunion. Présicez votre
date s'il vous plait ?
            </noinput>

            <noinput count="3">
                  Bon, très bien, je considère que vous voulez réserver
la salle de réunion aujourd'hui.
                  <throw event= »non_date »/>
            </noinput>

            <catch event= « non date »>
                  <assign name= « fdate » expr= « aujourd'hui »/>
            </catch>

            <catch event="date passe">
                  Le <value expr= « date »/> est passé. Précisez une
autre date s'il vous plait
            </catch>

            <filled>
                  <value expr="fdate"/>
                  <assign name="date" expr="fdate"/>
            </filled>
      </field>

      <field name="fheure" type="time">

            <prompt> »A quelle heure, s'il vous plait »</prompt>

            <nomatch>
                  Excusez-moi, ce que vous dites est invalide. S'il vous
plait de parler un temps correct.
            </nomatch>

            <noinput count="1">
                  Il me faut le temps de votre réunion. Présicez l'heure
de votre réunion s'il vous plait ?
            </noinput>

            <noinput count="2">
```

```
                    Il faut préciser l'heure de votre réunion. Donnez mois
votre temps de réservation, s'il vous plait ?
              </noinput>

              <noinput count="3">
                    Bon, très bien, je vous considère que vous voulez
réserver maintenant.
                    <throw event= « non_heure »/>
              </noinput>

              <catch event= »non heure »>
                    <assign  name= « fheure » expr= « maintenant »/>
              </catch>

              <filled>
                    <assign name="heure" expr="fheure"/>
              </filled>

      </field>

      <field name="ftaille" type="number">

              <prompt> Pour combien de personnes, s'il vous plait </prompt>

              <nomatch>
                    Excusez-moi, ce que vous dites est invalide. S'il vous
plait de parler un nombre correct.
              </nomatch>

              <noinput count="1">
                    Il me faut le nombre de participants. Présicez-le s'il
vous plait ?
              </noinput>

              <noinput count="2">
                    Bon, très bien, je vous laisse.
                    <throw event= »non_taille »/>
              </noinput>

              <catch event= « non taille »>
                    <assign name="ftaille" expr="-1"/>
              </catch>

              <filled>
                    <assign name="taille" expr="ftaille"/>
                    <goto next="#frmCloture"/>

              </filled>

      </field>

    </form>

    <form id="frmCloture">
      <block>
              Très bien, je vous réserve la salle de réunion le
              <value expr= « date »/> à
              <value expr= « heure » />
              <if cond="taille != '-1' »>pour
                    <value expr= « taille » />personnes.
              </if>
              Merci et au revoir
      </block>
    </form>

   </vxml>
```

A.2 Code source élaboré dans V_Builder

```
< ?xml version= « 1.0 » encoding= « Cp1252 » ?>
< !DOCTYPE vxml PUBLIC '-//Nuance/DTD VoiceXML 1.0//EN'
'http ://voicexml.nuance.com/dtd/nuancevoicexml-1-2.dtd'>
<vxml version= « 1.0 »>

< !—paramètres→
  <var name="date" />
  <var name="heure" />
  <var name="taille" />

< !—Ouverture →
  <form id= « frmOuverture »>
    <block>Bonjour, Vous êtes en connexion avec le service de réservation
de la salle de réunion.</block>
    <field name="tache racine">
      <grammar src= « ../grammars/autogen/Main.gsc »
type= »application/x-nuance-gsc » />
      <prompt bargein= « true »>Je vous écoute !</prompt>
      <prompt count= « 2 »>Bonjour, bienvenue au service de réservation
de la salle de réunion. Vous pouvez parler pour réserver une salle à
votre réunion</prompt>
      <catch event="help">
        <reprompt />
      </catch>
      <catch event="imcomprehension">
        Je m'occupe le service de réservation de la salle de réunion. Que
puis-je faire pour vous ?
      </catch>
      <catch event="politesse">
        Bonjour, bienvenue au service de réservation de la salle de
réunion. Vous pouvez parler !
      </catch>
      <noinput>J'entends rien.
        <throw event= « politesse » />
      </noinput>
      <nomatch>Désolé, je ne comprends pas ce que vous dites.
        <throw event="imcomprehension" />
      </nomatch>
      <filled>
        <if cond="tache racine == 'bonjour' || tache_racine == 'salut' ||
tache racine == 'bonsoir'" >
            <clear namelist="tache racine"/>
            <throw event="politesse"/>
         <else/>
            <goto next="#frmEchange01"/>
         </if>
      </filled>
    </field>
  </form>

<!—Echanges→

  <form id="frmEchange01">
    <field name="fdate">
      <grammar src= »../grammars/autogen/LIB_DATE.gsc »
type= »application/x-nuance-gsc » />
      <prompt bargein= «true »>Pour quel jour, s'il vous plait</prompt>
```

```
      <nomatch>Excusez-moi, ce que vous dites est invalide. S'il vous
plait de parler un jour correct.</nomatch>
      <noinput count= « 1 »>Date est l'élément important pour réserver.
Présicez une date précise s'il vous plait ?</noinput>
      <noinput count= « 2 »>Il faut préciser la date de réunion. Présicez
votre date s'il vous plait ?</noinput>
      <noinput count= « 3 »>Bon, très bien, je vous considère que vous
voudriez réserver aujourd'hui.
        <throw event= »non_date » />
      </noinput>
      <catch event= « non date »>
        <assign name= « fdate » expr= « aujourd'hui » />
      </catch>
      <catch event= « date passe »>Le
        <value expr= « date » />est passé. Précisez un jour s'il vous
plait
      </catch>
      <filled>
        <assign name="date" expr="fdate" />
      </filled>
    </field>
    <field name="fheure" type="time">
      <grammar src= « ../grammars/autogen/LIB_TIME.gsc »
type= »application/x-nuance-gsc » />
      <prompt bargein= »true »>A quelle heure, s'il vous plait</prompt>
      <nomatch>Excusez-moi, ce que vous dites est invalide. S'il vous
plait de parler un temps correct.</nomatch>
      <noinput count= « 1 »> Il faut préciser l'heure de réservation.
Présicez l'heure de votre réunion s'il vous plait ?</noinput>
      <noinput count= « 2 »>Je vous demande l'heure de votre réunion.
Présicez votre temps de réservation, s'il vous plait ?</noinput>
      <noinput count= « 3 »>Bon, très bien, je vous considère que vous
voudriez réserver maintenant.
        <throw event= « non_heure » />
      </noinput>
      <catch event= « non heure »>
        <assign name= « fheure » expr= « maintenant » />
      </catch>
      <filled>
        <assign name="heure" expr="fheure" />
      </filled>
    </field>
    <field name="ftaille">
      <grammar src= »../grammars/autogen/LIB_NUMBER.gsc »
type= »application/x-nuance-gsc » />
      <prompt bargein= « true »> Pour combien de personnes, s'il vous
plait </prompt>
      <nomatch>Excusez-moi, ce que vous dites est invalide. S'il vous
plait de parler un nombre correct.</nomatch>
      <noinput count= « 1 »>Il faut préciser le nombre de participants.
Présicez-le s'il vous plait ?</noinput>
      <noinput count= « 2 »>Bon, très bien, je vous laisse.
        <throw event= « non_taille » />
      </noinput>
      <catch event= « non taille »>
        <assign name="ftaille" expr="-1" />
      </catch>
      <filled>
        <assign name="taille" expr="ftaille" />
        <goto next="#frmCloture" />
      </filled>
    </field>
  </form>

<!—Clôture →
```

```
  <form id="frmCloture">
    <block name= « block2 »>Très bien, je vous réserve la salle de
réunion le <value expr= « date »/>à
      <value expr= « heure » />
      <if cond="taille != '-1' »>pour
        <value expr= »taille » />personnes.
      </if>Merci et au revoir
    </block>
  </form>
</vxml>
```

Annexe B : Thèmes et base des ontologies du service d'organisation de réunion

B.1. Thèmes du service d'organisation de réunion

```
#Thème d'ouverture
THEME: Ouverture
       [FP_Ouverture]
;

#Thème de clôture
THEME: Cloture
       [FS_Cloture]
;

#Thème d'identification de l'utilisateur
THEME: Usager
       [FS_Prenom]
       [FS_Nom]
       [FS_NomComplet]
       [FS_Sexe]
       [FS_Role]
;

#Thème de réservation
THEME: Reservation
       [FF_Reserver]
       [FFI_Reservation]
       [FS_Reservation]

       [FFS_NomSalle]
       [FFS_Taille]
       [FFS_Materiel]
       [FFS_Date]
       [FFS_Heure]
       [FFS_Duration]

       [FFS_Disponibilite]

       [FS_NomSalle]
       [FS_Taille]
       [FS_Materiel]
       [FS_Date]
       [FS_Temps]
       [FS_Heure]
       [FS_Duration]

       [FF_Negocier]
       [FF_Reporter]
       [FF_Annuler]

;

#Thème de convocation des participants
THEME: Convocation
       [FFI_Convocation]
       [FF_Convocation]
```

```
       [FFS_Mode]
       [FFS_Membre]

       [FS_Membre]
       [FS_Mode]
;

#Thème d'annonce d'une réunion
THEME: Annonce
       [FS_Identification_Confirmation]
       [FS_Reponse_Occupee]
       [FS_Reponse_Present]
       [FS_Reponse_Absent]
       [FS_Reponse_Message]

       [FFS_Materiel]
       [FS_Materiel]
;

#Thème de négociation
THEME: Negociation
       [FS_Negociation]
       [FFS_ChangerNomSalle]
       [FFS_ChangerDate]
       [FFS_ChangerHeure]
;
```

B.2. Base des ontologies

```
#Concepts pour l'ouverture
[FP_Ouverture]
       (hello)
       (salut)
       (bonjour)
       (bonsoir)
       (*oui allô)
;
#concepts pour le clôture d'un dialogue
[FS_Cloture]
       (stop)
       (au revoir)
       (bye)
       (ciao)
       (à bientôt)
;

[_oui]
       (oui)
       (ouais)
       (d'ac)
       (d'accord)
       (ok)
       (parfait)
       (ça me convient)
       (ça serait bien)
       (génial)
       (cool)
       (très bien)
       (incroyable)
       (excellent)
       (bien sûr)
       (biensûr)
       (sans problème)
       (oui c'est moi)
;
```

```
[_non]
	(non *mais)
	(je ne sais pas)
	(pas possible)
	(aucun)
	(aucune)
	(pas nécessaire)
	(n'en ai pas)
	(n'en aurai pas)
	(n'en aurais pas)
;
[FS_Identification_Confirmation]
	(c'est bien moi)
	(exactement)
	(*oui c'est ça)
;
[FS_Reponse_Occupee]
	(je serais occupé)
	(j'aurai d'autres choses à faire)
	(je ne serai pas disponible)
;
[FS_Reponse_Present]
	(je viendrai)
	(je vais venir)
	(je vais y aller)
	(je serai disponible)
	(je confirme)
	(je suis d'accord)
	(j'y serai)
	(je serai présent)
	(ok j'arrive)
	(ok j'arriverai)
;
[FS_Reponse_Absent]
	(je ne pourrai pas venir)
	(je serai absent)
	(je serai en mission)
;
[FS_Reponse_Message]
	(dis lui)
	(dites lui)
	(tiens lui au courant *que)
	(tenez lui au courant *que)
	(laisse lui un message)
;
[Nom]
	(nguyen)
	(caelen)
	(causse)
	(hollard)
	(millien)
	(fouquet)
	(dupont)
	(serignat)
	(besacier)
	(bigi)
	(lecomte)
;
[Prenom]
	(hoa)
	(jean)
	(jean françois)
	(olivier)
	(solange)
	(evelyne)
	(yannick)
	(anne claire)
```

```
      (paul)
      (laurent)
      (brigitte)
      (alain)
;
[Role]
      (directeur)
      (directrice)
      (étudiant)
      (étudiante)
      (chercheur)
      (stagiaire)
;
[NomSalle]
      (lafayette)
      (aquarium)
      (l'aquarium)
      (c cent quinze)
;
[Materiel]
      (vidéo projecteur)
      (visio conférence)
      (projecteur)
      (rétro projecteur)
      (portable)
      (tableau tactile)
      (feutres)
;
[Mode]
      (email)
      (fax)
      (téléphone)
;
[Membre]
      (pve)
      (geod)
      (clips)
      (multicom)
;

#déterminer la stratégie du contrôleur du dialogue
[FFS_Strategie]
      (quelle stratégie)
      (quelle est la stratégie)
      (quel type de stratégie)
      (quel type de stratégies)
      (c'est ART stratégie *de [Strategie])
      (ART stratégie est [Strategie])
      (est ce que ART stratégie est [Strategie])
ART
      (une)
      (la)
      (cette)
;
[Strategie]
      (directive)
      (réactive)
      (négocié)
      (négociation)
      (constructive)
      (construction)
      (coopérative)
      (coopération)
;
#actes concernant une négociation
[FFS_ChangerNomSalle]
      (change la salle)
```

```
      (changez la salle)
;
[FFS_ChangerDate]
      (trouvez moi une autre date)
;
[FFS_ChangerHeure]
      (trouvez moi une autre heure)
;

#identification de l'utilisateur
[FS_NomComplet]
      ([Prenom] [Nom])
      (je suis [Prenom] [Nom])
      (je m'appelle [Prenom] [Nom])
      (c'est [Prenom] [Nom])
      (ici [Prenom] [Nom])
;
[FS_Prenom]
      (mon prénom *est [Prenom])
      (je suis [Prenom])
      (je m'appelle [Prenom])
      (*ici [Prenom])
      (c'est [Prenom])
;
[FS_Nom]
      (*mon nom *est [Nom])
      (je suis [Nom])
      (je m'appelle [Nom])
      (*[Sexe] [Nom])
;
[FS_Sexe]
      ([Sexe])
;
[FS_Role]
      ([Role])
;

#Réservation
[FF_Reserver]
      ([_reserver])
;
#(<s> il me faut une salle)
[_reserver]
      (pour la réunion)
      (pour une réunion)
      (pour réserver)
      (je voudrais réserver)
      (réserver)
      (je veux réserver)
      (j'aimerais réserver)
      (je souhaite réserver)
      (veux une salle)
      (pour la réservation)
      (organiser une réunion)
      (réservez moi)
      (réserve moi)
;
[FFI_Reservation]
      (réserve une salle)
      (réservez une salle)
;
[FS_Reservation]
      (réserve la)
      (réservez la)
;

[FF_Negocier]
```

```
      ([_negocier])
;
[_negocier]
      (dis lui que j'en ai besoin)
      (dites lui que j'en ai besoin)
      (dis leur que j'en ai besoin)
      (dites leur que j'en ai besoin)
      (laisser cette salle à moi)
      (laissez cette salle à moi)
      (laissez moi cette salle)
      (me laisser cette salle)
      (me laisse cette salle)
      (me laissent cette salle)
      (me la laisser)
;
[FF_Reporter]
      ([_reporter])
;
[_reporter]
      (reporter ma réunion)
      (reportez ma réunion)
      (reporte ma réunion)
      (reporter la mienne)
      (reportez la mienne)
      (reporte la mienne)
;
[FF_Annuler]
      ([_annuler])
;
[_annuler]
      (l'annulation de)
      (VERBE ma réservation)
      (VERBE ma réunion)
VERBE
      (annulez)
      (annule)
      (annuler)
      (supprimer)
      (supprime)
      (supprimez)
      (j'annule)
;

[FFS_NomSalle]
      (quelle salle)
      (laquelle)
      (que peux tu me proposer)
;
[FFS_Taille]
      (quelle taille)
      (quelle est sa capacité)
      (combien de places)
;
[FFS_Materiel]
      (quels sont *les matériels)
      (est ce qu'il y a des matériels)
      (est-ce qu'il y a des matériels)
;
[FFS_Date]
      (quelles sont les possibilités)
      (quel jour)
      (quels jours)
      (quelle date)
;
[FFS_Heure]
      (*à quelle heure)
;
```

```
[FFS_Duration]
      (*pour combien de temps)
;
[FFS_Disponibilite]
      (est ce que la salle *[NomSalle] est disponible)
      (est-ce que la salle *[NomSalle] est disponible)
;

[FS_Negociation]
      ([_Negociation_OK])
      ([_Negociation_NON])
;
[_Negociation_OK]
      (j'accepte)
      (ok je la lui laisse)
      (sans problème)
      (d'accord)
;
[_Negociation_NON]
      (*je *ne veux pas changer)
      (*je *ne peux pas *la laisser)
;
[FS_NomSalle]
      (*une salle *[NomSalle])
      (*la *salle [NomSalle])
      ([Salle_Hasard])
;
[Salle_Hasard]
      (n'importe laquelle)
      (sans important)
      (peu importe)
      (je n'en sais pas)
;
[FS_Taille]
      ([Plus_Taille])
      ([Moins_Taille])
;
[Moins_Taille]
      (maximum [Taille])
;
[Plus_Taille]
      (salle de [Taille])
      (pour *une *réunion *de [Taille])
      (environ [Taille])
      (au moins [Taille])
;
[Taille]
      ([Nombre] UN_NOM)
UN_NOM
      (personne)
      (personnes)
      (place)
      (places)
;

[FS_Materiel]
      (*avec *ACTICLE [Materiel])
      (*ACTICLE [Materiel] et *ACTICLE [Materiel])
ACTICLE
      (un)
      (une)
      (le)
      (la)
      (les)
;
[FS_Date]
      (*le [Jour] [Date] *[Mois] *[Annee])
```

```
      (*le [Jour] [Relative])
      (ce [Jour])
      ([Date] [Mois] *[Annee])
      (le [Date] *[Mois] *[Annee])
      (demain)
      (après demain)
      (maintenant)
      (aujourd'hui)
      (tout de suite)
;

[FS_Temps]
      (de [Heure] *[N_Minute] à [Heure] *[N_Minute])
;
[FS_Heure]
      (*à [Heure] *[N_Minute])
      (vers [Heure] *[N_Minute])
      (avant [Heure] *[N_Minute])
      (après [Heure] *[N_Minute])
      (*Plage *vers [Heure] *[N_Minute])
      (*Plage *avant [Heure] *[N_Minute])
Plage
      (l'après midi)
      (après midi)
      (martin)
;
[Plus_Heure]
      (avant [Heure] *[N_Minute])
;
[Moins_Heure]
      (après [Heure] *[N_Minute])
;

[FS_Duration]
      (ça dure [Heure] *[N_Minute])
      (environ [Heure] *[N_Minute])
      (toute la martinée)
      (toute la journée)
      (un jour entier)
      (jusqu'à [Heure])
;

[FFI_Convocation]
      (prévenez)
      (préviens)
      (envoyez un message)
      (informez)
;
[FF_Convocation]
      (pouvez vous V_Informer)
      (peux tu V_Informer)
      (est ce que tu peux V_Informer)
      (est ce que vous pouvez V_Informer)
V_Informer
      (prévenir)
      (informer)
      (envoyer)
;

[FFS_Mode]
      (quel mode)
;
[FFS_Membre]
      (qui sont les membres)
;

[FS_Membre]
```

```
      (Pre_Mem [Membre])
Pre_Mem
      (membres de l'équipe)
      (membres du projet)
      (membre du projet)
      (membre de l'équipe)
      (membre de)
      (membres de)
      (membre du)
      (membres du)
      (gens du)
      (gens de)
      (personnes du)
;
[FS_Mode]
      (par [Mode])
      (en [Mode])
;

[Relative]
      (prochain)
      (prochaine)
      (prochains)
      (dernier)
      (derniers)
      (dernière)
      (dernières)
;
[Date]
      (premier)
      (deux)
      (trois)
      (quatre)
      (cinq)
      (six)
      (sept)
      (huit)
      (neuf)
      (dix)
      (onze)
      (douze)
      (treize)
      (quatorze)
      (quinze)
      (seize)
      (dix sept)
      (dix huit)
      (dix neuf)
      (vingt)
      (vingt et un)
      (vingt deux)
      (vingt trois)
      (vingt quatre)
      (vingt cinq)
      (vingt six)
      (vingt sept)
      (vingt huit)
      (vingt neuf)
      (trente)
      (trente et un)
;
[Annee]
      (mille huit cent *NBR1_99)
      (mille neuf cent *NBR1_99)
      (deux mille *NBR1_99)
NBR1_99
      (Nbr_1_9)
```

```
       (Nbr_10_19)
       (vingt *Nbr_AND_1_9)
       (trente *Nbr_AND_1_9)
       (quarante *Nbr_AND_1_9)
       (cinquante Nbr_AND_1_9)
       (soixante Nbr_10_19)
       (soixante *Nbr_AND_1_9)
       (quatre vingt Nbr_10_19)
       (quatre vingt *Nbr_AND_1_9)
Nbr_10_19
       (dix)
       (onze)
       (douze)
       (treize)
       (quatorze)
       (quinze)
       (seize)
       (dix sept)
       (dix huit)
       (dix neuf)
Nbr_AND_1_9
       (et un)
       (Nbr_2_9)
Nbr_1_9
       (un)
       (Nbr_2_9)
Nbr_2_9
       (deux)
       (trois)
       (quatre)
       (cinq)
       (six)
       (sept)
       (huit)
       (neuf)
;
[Jour]
       (lundi)
       (mardi)
       (mercredi)
       (jeudi)
       (vendredi)
       (samedi)
       (dimanche)
       (weekend)
;
[Mois]
       (janvier)
       (février)
       (mars)
       (avril)
       (mai)
       (juin)
       (juillet)
       (août)
       (septembre)
       (octobre)
       (novembre)
       (décembre)
;

[Heure]
       (une heure)
       (deux heures)
       (trois heures)
       (quatre heures)
       (cinq heures)
```

```
      (six heures)
      (sept heures)
      (huit heures)
      (neuf heures)
      (dix heures)
      (onze heures)
      (douze heures)
      (midi)
      (treize heures)
      (quatorze heures)
      (quinze heures)
      (seize heures)
      (dix sept heures)
      (dix huit heures)
      (dix neuf heures)
      (vingt heures)
      (vingt et une heures)
      (vingt deux heures)
      (vingt trois heures)
      (vingt quatre heures)
      (minuit)
      (zéro)
;
[N_Minute]
      ([Moins_Min])
      ([Plus_Min])
;
[Moins_Min]
      (moins quart)
      (moins [Nombre])
;
[Plus_Min]
      (et demi)
      (et quart)
      (plus [Nombre] *minutes)
      ([Nombre] *minutes)
;

[_informer]
      (prévenez *les *membres)
;

#Nombre
[Nombre]
      (Nbr_1000000_9999999)
      (Nbr_100000_999999)
      (Nbr_10000_99999)
      (Nbr_1000_9999)
      (Nbr_100_999)
      (Nbr_10_99)
      (Nbr_0_9)
Nbr_1000000_9999999
      (*Nbr_1_9 million *Nbr_1_9)
      (*Nbr_1_9 million Nbr_10_99)
      (*Nbr_1_9 million Nbr_100_999)
      (*Nbr_1_9 million Nbr_1000_9999)
      (*Nbr_1_9 million Nbr_10000_99999)
      (*Nbr_1_9 million Nbr_100000_999999)
Nbr_100000_999999
      (Nbr_100_999 mille *Nbr_1_9)
      (Nbr_100_999 mille Nbr_10_99)
      (Nbr_100_999 mille Nbr_100_999)
Nbr_10000_99999
      (Nbr_10_99 mille *Nbr_1_9)
      (Nbr_10_99 mille Nbr_10_99)
      (Nbr_10_99 mille Nbr_100_999)
Nbr_1000_9999
```

```
      (*Nbr_2_9 mille *Nbr_1_9)
      (*Nbr_2_9 mille Nbr_10_99)
      (*Nbr_2_9 mille Nbr_100_999)
Nbr_100_999
      (*Nbr_2_9 cent *Nbr_1_9)
      (*Nbr_2_9 cent Nbr_10_99)
Nbr_10_99
      (Nbr_10_19)
      (vingt *Nbr_AND_1_9)
      (trente *Nbr_AND_1_9)
      (quarante *Nbr_AND_1_9)
      (cinquante Nbr_AND_1_9)
      (soixante Nbr_10_19)
      (soixante *Nbr_AND_1_9)
      (quatre vingt Nbr_10_19)
      (quatre vingt *Nbr_AND_1_9)
Nbr_10_19
      (dix)
      (onze)
      (douze)
      (treize)
      (quatorze)
      (quinze)
      (seize)
      (dix sept)
      (dix huit)
      (dix neuf)
Nbr_AND_1_9
      (et un)
      (Nbr_2_9)
Nbr_0_9
      (zéro)
      (Nbr_1_9)
Nbr_1_9
      (un)
      (Nbr_2_9)
Nbr_2_9
      (deux)
      (trois)
      (quatre)
      (cinq)
      (six)
      (sept)
      (huit)
      (neuf)
;
```

Annexe C : Dialogues pertinents enregistrés par Mélina

Nous présentons ici un corpus de dialogue homme-machine enregistré au cours de la phase d'expérimentation du système Mélina. Ce corpus comporte 14 dialogues, qui couvrent tous les cas pertinents concernant le service de réservation de salle de réunion.

Dialogue 1 :

```
[Tue Oct 05 10:11:18 CEST 2004]
     Stratégie : directive
     But : Ouverture ; Status : Atteint ; Valeur : commencer
     Actes du système : FS[Ouverture(commencer)]
[Mélina] : ici le service de réservation du laboratoire clips, je suis
           Mélina et vous ?
[Usager] : salut c'est hoa
     Actes de l'utilisateur : FP[Ouverture(x)] & FS[Prenom(hoa)]
     Stratégie : coopérative
     But : Usager ; Status : Atteint ; Valeur : 2
     Actes du système : FS[Sexe(0)] & FS[Nom(nguyen)] &
            FS[Role(thésard)] & FS[Groupe(PVE)]
[Mélina] : bonjour monsieur nguyen, je vous écoute...
[Usager] : réserve moi la salle lafayette jeudi prochain de neuf heures à
           onze heures avec un vidéo projecteur s'il te plaît
     Actes de l'utilisateur : FF[Action(reserver)] &
           FS[NomSalle(lafayette)] & FS[Date(20041007)] & FS[Heure(540)]
           & FS[Duration(120)] & FS[Materiel(vidéo projecteur)]
     Stratégie : coopérative
     But : Reservation ; Status : Atteint ; Valeur : 2
     Actes du système : FS[Taille(45)]
[Mélina] : sa capacité est de 45 places,
[Usager] : je sais merci beaucoup et au revoir
     Actes de l'utilisateur : FS[Cloture(x)]
     Stratégie : réactive
     But : Cloture ; Status : En cours de traitement ; Valeur : x
     Actes du système : FS[Cloture(OK)]
[Mélina] : au revoir monsieur nguyen à bientôt.
```

Dialogue 2 :

```
[Tue Oct 05 10:12:27 CEST 2004]
     Stratégie : directive
     But : Ouverture ; Status : Atteint ; Valeur : commencer
     Actes du système : FS[Ouverture(commencer)]
[Mélina] : ici le service de réservation du laboratoire clips, je suis
           Mélina et vous ?
[Usager] : bonjour ici jean françois
     Actes de l'utilisateur : FP[Ouverture(x)] & FS[Prenom(jean
           françois)]
     Stratégie : coopérative
     But : Usager ; Status : Atteint ; Valeur : 6
     Actes du système : FS[Sexe(0)] & FS[Nom(serignat)] & FS[Role(chef
           d'équipe)] & FS[Groupe(geod)]
[Mélina] : bonjour monsieur serignat, je vous écoute...
[Usager] : je veux réserver la salle lafayette pour la réunion de
           l'équipe geod
     Actes de l'utilisateur : FF[Action(reserver)] &
           FS[NomSalle(lafayette)]
```

```
     Stratégie : coopérative
     But : Temps ; Status : Nouveau but ; Valeur : null
     Actes du système : FS[Taille(45)] & FFS[Temps(x)]
[Mélina] : sa capacité est de 45 places, c'est pour quel jour, s'il vous
           plaît ?
[Usager] : pour jeudi prochain de onze heures à treize heures
     Actes de l'utilisateur : FS[Date(20041007)] & FS[Heure(660)] &
           FS[Duration(120)]
     Stratégie : coopérative
     But : Reservation ; Status : Atteint ; Valeur : 2
     Actes du système : FS[Reservation(2)]
[Mélina] : je vous réserve la salle lafayette à 11 heures le jeudi 7
           octobre 2004 pendant 2 heures
[Usager] : très bien et prévenez les membres de geod aussi
     Actes de l'utilisateur : FS[Reservation(OK)] & FF[Convocation(x)] &
           FS[Membre(geod)]
     Stratégie : réactive
     But : Convocation ; Status : En cours de traitement ; Valeur : OK
     Actes du système : FFS[Convocation(OK)]
[Mélina] : je vais les contacter par téléphone ?
[Usager] : par téléphone
     Actes de l'utilisateur : FS[Mode(téléphone)]
     Stratégie : réactive
     But : Convocation ; Status : Atteint ; Valeur : OK
     Actes du système : FS[Convocation(OK)]
[Mélina] : très bien, je vais prévenir les participants
[Usager] : merci et au revoir
     Actes de l'utilisateur : FS[Cloture(x)]
     Stratégie : réactive
     But : Cloture ; Status : En cours de traitement ; Valeur : x
     Actes du système : FS[Cloture(OK)]
[Mélina] : au revoir monsieur serignat à bientôt.
```

Dialogue 3 :

```
[Tue Oct 05 10:13:51 CEST 2004]
     Stratégie : directive
     But : Ouverture ; Status : Atteint ; Valeur : commencer
     Actes du système : FS[Ouverture(commencer)]
[Mélina] : ici le service de réservation du laboratoire clips, je suis
           Mélina et vous ?
[Usager] : c'est jean caelen à l'appareil
     Actes de l'utilisateur : FS[Prenom(jean)] & FS[Nom(caelen)]
     Stratégie : coopérative
     But : Usager ; Status : Atteint ; Valeur : 1
     Actes du système : FS[Sexe(0)] & FS[Role(directeur)] &
           FS[Groupe(PVE)]
[Mélina] : bonjour monsieur le directeur ! je vous écoute...
[Usager] : est ce que tu peux me réserver la salle lafayette jeudi
           prochain de dix heures à midi
     Actes de l'utilisateur : FF[Action(reserver)] &
           FS[NomSalle(lafayette)] & FS[Date(20041007)] & FS[Heure(600)]
           & FS[Duration(120)]
     Stratégie : coopérative
     But : Conflit ; Status : En cours de traitement ; Valeur : 2
     Actes du système : FS[Erreur(-12)] & FS[Conflit(2)]
[Mélina] : désolé, cette salle est déjà réservée !  Plus précisément, la
           salle lafayette est déjà retenue par monsieur hoa nguyen de 9
           heures  à 11 heures et par monsieur jean françois serignat de
           11 heures  à 13 heures le jeudi 7 octobre 2004
[Usager] : dis leur que j'en ai besoin et qu'ils doivent me la laisser
     Actes de l'utilisateur : FF[Action(negocier)]
     Stratégie : réactive
     But : Cloture ; Status : Nouveau but ; Valeur : null
```

```
     Actes du système : FS[Action(suspendre à négocier)]
[Mélina] : d'accord je vais l'appeler et je vous tiens au courrant...
```

Dialogue 4 :

```
[Tue Oct 05 10:15:03 CEST 2004]
     Stratégie : réactive
     But : Ouverture ; Status : Atteint ; Valeur : commencer
     Actes du système : FS[Ouverture(commencer)] & FFS[Patient(1)]
[Mélina] : bonjour ici Mélina, le service de réservation du laboratoire
           clips ! êtes vous bien monsieur hoa nguyen ?
[Usager] : oui
     Actes de l'utilisateur : FS[Patient(OK)]
     Stratégie : réactive
     But : Negociation ; Status : En cours de traitement ; Valeur : null
     Actes du système : FFS[Negociation(x)]
[Mélina] : monsieur jean caelen a une réunion importante à 10 heures
           pendant 2 heures le jeudi 7 octobre 2004 . Il vous demande si
           vous pourriez lui laisser la salle lafayette à ce moment là ?
[Usager] : c'est vrai ... ok j'accepte
     Actes de l'utilisateur : FS[Negociation(OK)]
     Stratégie : coopérative
     But : Temps ; Status : En cours de traitement ; Valeur : +1
     Actes du système : FS[Action(demander_reporter)] & FFS[Temps(+1)]
[Mélina] : voulez vous reporter votre réunion à un autre jour ? par
           exemple, le vendredi 8 octobre 2004
[Usager] : non réserve moi donc la salle aquarium
     Actes de l'utilisateur : FS[Temps(NON)] & FF[Action(reserver)] &
           FS[NomSalle(aquarium)]
     Stratégie : réactive
     But : Reservation ; Status : Atteint ; Valeur : 0
     Actes du système : FS[Taille(20)] & FS[Date(20041007)] &
           FS[Heure(540)] & FS[Duration(120)]
[Mélina] : sa capacité est de 20 places, je vous la réserve donc le jeudi
           7 octobre 2004 à 9 heures pendant 2 heures
[Usager] : très bien
     Actes de l'utilisateur : FS[Reservation(OK)]
     Stratégie : coopérative
     But : Convocation ; Status : Nouveau but ; Valeur : null
     Actes du système : FFS[Convocation(x)]
[Mélina] : voulez vous que je prévienne les participants ?
[Usager] : non merci au revoir
     Actes de l'utilisateur : FS[Convocation(NON)] & FS[Cloture(x)]
     Stratégie : réactive
     But : Cloture ; Status : En cours de traitement ; Valeur : x
     Actes du système : FS[Cloture(OK)]
[Mélina] : au revoir monsieur nguyen à bientôt.
```

Dialogue 5 :

```
[Tue Oct 05 10:18:43 CEST 2004]
     Stratégie : réactive
     But : Ouverture ; Status : Atteint ; Valeur : commencer
     Actes du système : FS[Ouverture(commencer)] & FFS[Patient(2)]
[Mélina] : bonjour ici Mélina, le service de réservation du laboratoire
           clips ! êtes vous bien monsieur jean françois serignat ?
[Usager] : oui c'est bien moi
     Actes de l'utilisateur :
     Stratégie : réactive
     But : Negociation ; Status : En cours de traitement ; Valeur : null
     Actes du système : FFS[Negociation(x)]
```

```
[Mélina] : monsieur jean caelen a une réunion importante à 10 heures
           pendant 2 heures le jeudi 7 octobre 2004 . Il vous demande si
           vous pourriez lui laisser la salle lafayette à ce moment là ?
[Usager] : ah bon ... d'accord mais reportez ma réunion à vendredi
           prochain
     Actes de l'utilisateur : FS[Negociation(OK)] & FF[Action(reporter)]
           & FS[Date(20041008)]
     Stratégie : coopérative
     But : Action ; Status : En cours de traitement ; Valeur : reserver
     Actes du système : FS[NomSalle(lafayette)] & FS[Taille(45)] &
           FS[Heure(660)] & FS[Duration(120)] & FFS[Action(reserver)]
[Mélina] : la salle lafayette serait à votre disposition, sa capacité est
           de 45 places, à 11 heures pendant 2 heures
[Usager] : parfait
     Actes de l'utilisateur : FS[Action(OK)]
     Stratégie : coopérative
     But : Cloture ; Status : Nouveau but ; Valeur : null
     Actes du système : FFS[Cloture(x)]
[Mélina] : j'espère que vous êtes satisfait...
[Usager] : à bientôt
     Actes de l'utilisateur : FS[Cloture(x)]
     Stratégie : réactive
     But : Cloture ; Status : En cours de traitement ; Valeur : x
     Actes du système : FS[Cloture(OK)]
[Mélina] : au revoir monsieur serignat à bientôt.
```

Dialogue 6 :

```
[Tue Oct 05 10:23:29 CEST 2004]
     Stratégie : réactive
     But : Ouverture ; Status : Atteint ; Valeur : commencer
     Actes du système : FS[Ouverture(commencer)] & FFS[Demandeur(1)]
[Mélina] : bonjour ici Mélina, le service de réservation du laboratoire
           clips ! êtes vous bien monsieur jean caelen ?
[Usager] : oui c'est bien moi
     Actes de l'utilisateur :
     Stratégie : réactive
     But : Notification ; Status : En cours de traitement ; Valeur : 5
     Actes du système : FFS[Notification(5)]
[Mélina] : monsieur hoa nguyen et monsieur jean françois serignat ont
           accepté de laisser la salle lafayette à 10 heures le jeudi 7
           octobre 2004  pour vous. Voulez vous accepter ce résultat ?
[Usager] : bien sûr
     Actes de l'utilisateur : FS[Notification(OK)]
     Stratégie : coopérative
     But : Reservation ; Status : Atteint ; Valeur : 3
     Actes du système : FS[Reservation(3)]
[Mélina] : je vous réserve la salle lafayette à 10 heures le jeudi 7
           octobre 2004 pendant 2 heures
[Usager] : et préviens aussi les membres du clips
     Actes de l'utilisateur : FF[Convocation(x)] & FS[Membre(clips)]
     Stratégie : réactive
     But : Convocation ; Status : En cours de traitement ; Valeur : OK
     Actes du système : FFS[Convocation(OK)]
[Mélina] : je vais les contacter par téléphone ?
[Usager] : par téléphone bien sûr
     Actes de l'utilisateur : FS[Mode(téléphone)] & FS[Convocation(OK)]
     Stratégie : coopérative
     But : Convocation ; Status : Satisfait ; Valeur : OK
     Actes du système : FS[Convocation(OK)]
[Mélina] : très bien, je vais prévenir les participants
[Usager] : merci bien au revoir
     Actes de l'utilisateur : FS[Cloture(x)]
     Stratégie : réactive
```

```
     But : Cloture ; Status : En cours de traitement ; Valeur : x
     Actes du système : FS[Cloture(OK)]
[Mélina] : au revoir monsieur caelen à bientôt.
```

Dialogue 7 :

```
[Tue Oct 12 12:10:51 CEST 2004]
     Stratégie : directive
     But : Ouverture ; Status : Atteint ; Valeur : commencer
     Actes du système : FS[Ouverture(commencer)]
[Mélina] : ici le service de réservation du laboratoire clips, je suis
           Mélina et vous ?
[Usager] : je si elle vient
     Actes de l'utilisateur : FS[Incomprehension(MDI-1)]
     Stratégie : directive
     But : Usager ; Status : Nouveau but ; Valeur : null
     Actes du système : FS[Ouverture(répéter)] & FFS[Usager(x)]
[Mélina] : ici Mélina, le service de résevation du clips ! puis je vous
           demander votre nom ?
[Usager] : nguyen
     Actes de l'utilisateur : FS[Nom(nguyen)]
     Stratégie : coopérative
     But : Reservation ; Status : Nouveau but ; Valeur : null
     Actes du système : FS[Groupe(PVE)] & FFS[Reservation(x)]
[Mélina] : quelles sont vos préférences de réservation ?
[Usager] : j' aimerais réserver la salle lafayette
     Actes de l'utilisateur : FF[Action(reserver)] &
           FS[NomSalle(lafayette)]
     Stratégie : coopérative
     But : Temps ; Status : Nouveau but ; Valeur : null
     Actes du système : FS[Taille(45)] & FFS[Temps(x)]
[Mélina] : sa capacité est de 45 places, c'est pour quel jour, s'il vous
           plaît ?
[Usager] : est elle disponible le genre jeudi prochain
     Actes de l'utilisateur : FS[Date(20041014)]
     Stratégie : coopérative
     But : Heure ; Status : Nouveau but ; Valeur : null
     Actes du système : FFS[Heure(x)]
[Mélina] : à quelle heure s'il vous plaît ?
[Usager] : non de dix heures à midi
     Actes de l'utilisateur : FS[Heure(600)] & FS[Duration(120)]
     Stratégie : réactive
     But : Reservation ; Status : Atteint ; Valeur : 2
     Actes du système : FS[Reservation(2)]
[Mélina] : je vous réserve la salle lafayette à 10 heures le jeudi 14
           octobre 2004 pendant 2 heures
[Usager] : j' aimerais aussi sinon vidéo projecteur
     Actes de l'utilisateur : FS[Materiel(vidéo projecteur)]
     Stratégie : réactive
     But : Convocation ; Status : Nouveau but ; Valeur : null
     Actes du système : FFS[Convocation(x)]
[Mélina] : voulez vous que je prévienne les participants ?
[Usager] : oui tous les membres du projet pve
     Actes de l'utilisateur : FS[Convocation(OK)] & FS[Membre(pve)]
     Stratégie : coopérative
     But : Mode ; Status : Nouveau but ; Valeur : null
     Actes du système : FFS[Mode(x)]
[Mélina] : par email ou par téléphone ?
[Usager] : pas téléphone
     Actes de l'utilisateur : FS[Incomprehension(MDI-1)]
     Stratégie : directive
     But : Mode ; Status : Nouveau but ; Valeur : null
     Actes du système : FFS[Mode(téléphone)]
[Mélina] : je vais les contacter par quel moyen, téléphone par exemple ?
[Usager] : oui par téléphone
```

```
     Actes de l'utilisateur : FS[Mode(téléphone)]
     Stratégie : directive
     But : Convocation ; Status : Atteint ; Valeur : OK
     Actes du système : FS[Convocation(OK)]
[Mélina] : très bien, je vais prévenir les participants
[Usager] : merci bien et au revoir
     Actes de l'utilisateur : FS[Cloture(x)]
     Stratégie : réactive
     But : Cloture ; Status : En cours de traitement ; Valeur : x
     Actes du système : FS[Cloture(OK)]
[Mélina] : au revoir monsieur nguyen à bientôt.
```

Dialogue 8 :

```
[Tue Oct 19 14:18:47 CEST 2004]
     Stratégie : directive
     But : Ouverture ; Status : Atteint ; Valeur : commencer
     Actes du système : FS[Ouverture(commencer)]
[Mélina] : ici le service de réservation du laboratoire clips, je suis
           Mélina et vous ?
[Usager] : bonjour c'est yannick
     Actes de l'utilisateur : FP[Ouverture(x)] & FS[Prenom(yannick)]
     Stratégie : coopérative
     But : Usager ; Status : Atteint ; Valeur : 8
     Actes du système : FS[Sexe(0)] & FS[Nom(fouquet)] &
           FS[Role(thésard)] & FS[Groupe(geod)]
[Mélina] : bonjour monsieur fouquet, je vous écoute...
[Usager] : je voudrais réserver une salle pour mercredi prochain
     Actes de l'utilisateur : FF[Action(reserver)] & FS[NomSalle(x)] &
           FS[Date(20041020)]
     Stratégie : coopérative
     But : Salle ; Status : Nouveau but ; Valeur : null
     Actes du système : FFS[Salle(x)]
[Mélina] : quelles sont vos préférences de salle ?
[Usager] : ben il me faut une salle de quinze places
     Actes de l'utilisateur : FS[Taille(+15)]
     Stratégie : coopérative
     But : Heure ; Status : Nouveau but ; Valeur : null
     Actes du système : FS[NomSalle(aquarium)] & FS[Taille(20)] &
           FFS[Heure(x)]
[Mélina] : la salle aquarium serait à votre disposition, sa capacité est
           de 20 places, à quelle heure s'il vous plaît ?
[Usager] : de quatorze heures à seize heures
     Actes de l'utilisateur : FS[Heure(840)] & FS[Duration(120)]
     Stratégie : coopérative
     But : Reservation ; Status : Atteint ; Valeur : 0
     Actes du système : FS[Reservation(0)]
[Mélina] : je vous réserve la salle aquarium à 14 heures le mercredi 20
           octobre 2004 pendant 2 heures
[Usager] : merci bien au revoir
     Actes de l'utilisateur : FS[Cloture(x)]
     Stratégie : réactive
     But : Cloture ; Status : En cours de traitement ; Valeur : x
     Actes du système : FS[Cloture(OK)]
[Mélina] : au revoir monsieur fouquet à bientôt.
```

Dialogue 9 :

```
[Tue Oct 19 14:20:32 CEST 2004]
     Stratégie : directive
     But : Ouverture ; Status : Atteint ; Valeur : commencer
     Actes du système : FS[Ouverture(commencer)]
[Mélina] : ici le service de réservation du laboratoire clips, je suis
           Mélina et vous ?
```

```
[Usager] : bonjour c'est solange hollard
     Actes de l'utilisateur : FP[Ouverture(x)] & FS[Prenom(solange)] &
            FS[Nom(hollard)]
     Stratégie : coopérative
     But : Usager ; Status : Atteint ; Valeur : 4
     Actes du système : FS[Sexe(2)] & FS[Role(chercheur)] &
            FS[Groupe(PVE)]
[Mélina] : bonjour madame hollard, je vous écoute...
[Usager] : est ce que tu peux me réserver la salle aquarium mercredi
            prochain de seize heures à dix huit heures s'il te plaît
     Actes de l'utilisateur : FF[Action(reserver)] &
            FS[NomSalle(aquarium)] & FS[Date(20041020)] & FS[Heure(960)] &
            FS[Duration(120)]
     Stratégie : coopérative
     But : Reservation ; Status : Atteint ; Valeur : 0
     Actes du système : FS[Taille(20)]
[Mélina] : sa capacité est de 20 places,
[Usager] : très bien j'ai besoin d'un vidéo projecteur et préviens aussi
            les membres du projet pve
     Actes de l'utilisateur : FS[Reservation(OK)] & FS[Materiel(vidéo
            projecteur)] & FF[Convocation(x)] & FS[Membre(pve)]
     Stratégie : réactive
     But : Mode ; Status : Nouveau but ; Valeur : téléphone
     Actes du système : FFS[Mode(téléphone)]
[Mélina] : je vais les contacter par quel moyen, téléphone par exemple ?
[Usager] : non je préfère que tu les contacte par email
     Actes de l'utilisateur : FS[Mode(email)]
     Stratégie : réactive
     But : Convocation ; Status : Atteint ; Valeur : OK
     Actes du système : FS[Convocation(OK)]
[Mélina] : très bien, je vais prévenir les participants
[Usager] : merci et au revoir
     Actes de l'utilisateur : FS[Cloture(x)]
     Stratégie : réactive
     But : Cloture ; Status : En cours de traitement ; Valeur : x
     Actes du système : FS[Cloture(OK)]
[Mélina] : au revoir madame hollard à bientôt.
```

Dialogue 10 :

```
[Tue Oct 19 14:23:22 CEST 2004]
     Stratégie : directive
     But : Ouverture ; Status : Atteint ; Valeur : commencer
     Actes du système : FS[Ouverture(commencer)]
[Mélina] : ici le service de réservation du laboratoire clips, je suis
            Mélina et vous ?
[Usager] : ici jean françois à l'appareil
     Actes de l'utilisateur : FS[Prenom(jean françois)]
     Stratégie : coopérative
     But : Usager ; Status : Atteint ; Valeur : 6
     Actes du système : FS[Sexe(0)] & FS[Nom(serignat)] & FS[Role(chef
            d'équipe)] & FS[Groupe(geod)]
[Mélina] : bonjour monsieur serignat, je vous écoute...
[Usager] : je veux réserver une salle pour mercredi prochain
     Actes de l'utilisateur : FF[Action(reserver)] & FS[NomSalle(x)] &
            FS[Date(20041020)]
     Stratégie : coopérative
     But : Salle ; Status : Nouveau but ; Valeur : null
     Actes du système : FFS[Salle(x)]
[Mélina] : quelles sont vos préférences de salle ?
[Usager] : comme d'habitudes la salle aquarium avec un vidéo projecteur
     Actes de l'utilisateur : FS[NomSalle(aquarium)] & FS[Materiel(vidéo
            projecteur)]
     Stratégie : coopérative
     But : Heure ; Status : Nouveau but ; Valeur : null
```

```
     Actes du système : FS[Taille(20)] & FFS[Heure(x)]
[Mélina] : sa capacité est de 20 places, à quelle heure s'il vous plaît ?
[Usager] : de quinze heures à dix sept heures
     Actes de l'utilisateur : FS[Heure(900)] & FS[Duration(120)]
     Stratégie : coopérative
     But : Conflit ; Status : En cours de traitement ; Valeur : 2
     Actes du système : FS[Erreur(-12)] & FS[Conflit(2)]
[Mélina] : désolé, cette salle est déjà réservée !  Plus précisément, la
           salle aquarium est déjà retenue par monsieur yannick fouquet
           de 14 heures  à 16 heures et par madame solange hollard de 16
           heures  à 18 heures le mercredi 20 octobre 2004
[Usager] : téléphone leur en disant que je leur demande de me laisser
           cette salle
     Actes de l'utilisateur : FF[Action(negocier)]
     Stratégie : réactive
     But : Cloture ; Status : Nouveau but ; Valeur : null
     Actes du système : FS[Action(suspendre à négocier)]
[Mélina] : d'accord je vais l'appeler et je vous tiens au courrant...
```

Dialogue 11 :

```
[Tue Oct 19 14:43:50 CEST 2004]
     Stratégie : réactive
     But : Ouverture ; Status : Atteint ; Valeur : commencer
     Actes du système : FS[Ouverture(commencer)] & FFS[Patient(9)]
[Mélina] : bonjour ici Mélina, le service de réservation du laboratoire
           clips ! êtes vous bien monsieur yannick fouquet ?
[Usager] : oui
     Actes de l'utilisateur : FS[Patient(OK)]
     Stratégie : réactive
     But : Negociation ; Status : En cours de traitement ; Valeur : null
     Actes du système : FFS[Negociation(x)]
[Mélina] : monsieur jean françois serignat a une réunion importante à 15
           heures  pendant 2 heures le mercredi 20 octobre 2004 . Il vous
           demande si vous pourriez lui laisser la salle aquarium à ce
           moment là ?
[Usager] : d'accord
     Actes de l'utilisateur : FS[Negociation(OK)]
     Stratégie : coopérative
     But : Temps ; Status : En cours de traitement ; Valeur : +1
     Actes du système : FS[Action(demander_reporter)] & FFS[Temps(+1)]
[Mélina] : voulez vous reporter votre réunion à un autre jour ? par
           exemple, le jeudi 21 octobre 2004
[Usager] : non je veux la salle lafayette alors
     Actes de l'utilisateur : FS[Temps(NON)] & FS[NomSalle(lafayette)]
     Stratégie : réactive
     But : Reservation ; Status : Atteint ; Valeur : 2
     Actes du système : FS[Taille(45)] & FS[Date(20041020)] &
           FS[Heure(840)] & FS[Duration(120)]
[Mélina] : sa capacité est de 45 places, je vous la réserve donc le
           mercredi 20 octobre 2004 à 14 heures pendant 2 heures
[Usager] : très bien
     Actes de l'utilisateur : FS[Reservation(OK)]
     Stratégie : coopérative
     But : Convocation ; Status : Nouveau but ; Valeur : null
     Actes du système : FFS[Convocation(x)]
[Mélina] : voulez vous que je prévienne les participants ?
[Usager] : non je vais le faire
     Actes de l'utilisateur : FS[Convocation(NON)]
     Stratégie : réactive
     But : Convocation ; Status : Atteint ; Valeur : NON
     Actes du système : FS[Convocation(NON)]
[Mélina] : bon je vous laisse de prévenir les participants
[Usager] : merci et au revoir
     Actes de l'utilisateur : FS[Cloture(x)]
```

```
     Stratégie : réactive
     But : Cloture ; Status : En cours de traitement ; Valeur : x
     Actes du système : FS[Cloture(OK)]
[Mélina] : au revoir monsieur fouquet à bientôt.
```

Dialogue 12 :

```
[Tue Oct 19 16:59:38 CEST 2004]
     Stratégie : réactive
     But : Ouverture ; Status : Atteint ; Valeur : commencer
     Actes du système : FS[Ouverture(commencer)] & FFS[Patient(10)]
[Mélina] : bonjour ici Mélina, le service de réservation du laboratoire
           clips ! êtes vous bien madame solange hollard ?
[Usager] : oui c'est bien moi
     Actes de l'utilisateur : FS[Patient(OK)]
     Stratégie : réactive
     But : Negociation ; Status : En cours de traitement ; Valeur : null
     Actes du système : FFS[Negociation(x)]
[Mélina] : monsieur jean françois serignat a une réunion importante à 15
           heures  pendant 2 heures le mercredi 20 octobre 2004 . Il vous
           demande si vous pourriez lui laisser la salle aquarium à ce
           moment là ?
[Usager] : je regrette mais ce n'est pas possible
     Actes de l'utilisateur : FS[Negociation(NON)]
     Stratégie : réactive
     But : Negociation ; Status : En cours de traitement ; Valeur : 4
     Actes du système : FFS[Negociation(4)]
[Mélina] : pourriez vous reporter votre réunion à un autre jour ?
[Usager] : désolé je ne veux pas changer la date de la réunion
     Actes de l'utilisateur : FS[Negociation(NON)]
     Stratégie : réactive
     But : Cloture ; Status : En cours de traitement ; Valeur : null
     Actes du système : FS[Cloture(OK)]
[Mélina] : Ce n'est pas grave, je vais lui proposer une autre solution !
           au revoir madame hollard à bientôt.
```

Dialogue 13 :

```
[Tue Oct 19 17:53:33 CEST 2004]
     Stratégie : réactive
     But : Ouverture ; Status : Atteint ; Valeur : commencer
     Actes du système : FS[Ouverture(commencer)] & FFS[Demandeur(6)]
[Mélina] : bonjour ici Mélina, le service de réservation du laboratoire
           clips ! êtes vous bien monsieur jean françois serignat ?
[Usager] : oui
     Actes de l'utilisateur : FS[Demandeur(OK)]
     Stratégie : réactive
     But : Notification ; Status : En cours de traitement ; Valeur : 3
     Actes du système : FFS[Notification(3)]
[Mélina] : monsieur yannick fouquet et  a accepté de laisser la salle
           aquarium à 15 heures le mercredi 20 octobre 2004  pour vous.
           Mais madame solange hollard ne voudrait pas modifié sa
           réservation ! Veuillez trouver une autre date s'il vous plaît
           ?
[Usager] : ben reporte ma réunion à jeudi prochain de dix heures à midi
           alors
     Actes de l'utilisateur : FF[Action(reporter)] & FS[Date(20041021)] &
           FS[Heure(600)] & FS[Duration(120)]
     Stratégie : coopérative
     But : Reservation ; Status : Atteint ; Valeur : 11
     Actes du système : FS[NomSalle(aquarium)] & FS[Taille(20)]
[Mélina] : la salle aquarium serait à votre disposition, sa capacité est
           de 20 places,
[Usager] : parfait
```

```
     Actes de l'utilisateur : FS[Reservation(OK)]
     Stratégie : coopérative
     But : Convocation ; Status : Nouveau but ; Valeur : null
     Actes du système : FFS[Convocation(x)]
[Mélina] : voulez vous que je prévienne les participants ?
[Usager] : oui tous les membres de l'équipe geod
     Actes de l'utilisateur : FS[Convocation(OK)] & FS[Membre(geod)]
     Stratégie : coopérative
     But : Mode ; Status : Nouveau but ; Valeur : null
     Actes du système : FFS[Mode(x)]
[Mélina] : par email ou par téléphone ?
[Usager] : par téléphone bien sûr
     Actes de l'utilisateur : FS[Mode(téléphone)]
     Stratégie : coopérative
     But : Convocation ; Status : Atteint ; Valeur : OK
     Actes du système : FS[Convocation(OK)]
[Mélina] : très bien, je vais prévenir les participants
[Usager] : merci bien et au revoir
     Actes de l'utilisateur : FS[Cloture(x)]
     Stratégie : réactive
     But : Cloture ; Status : En cours de traitement ; Valeur : x
     Actes du système : FS[Cloture(OK)]
[Mélina] : au revoir monsieur serignat à bientôt.
```

Dialogue 14 :

```
[Mon Nov 08 18:07:59 CET 2004]
     Stratégie : directive
     But : Ouverture ; Status : Atteint ; Valeur : commencer
     Actes du système : FS[Ouverture(commencer)]
[Mélina] : ici le service de réservation du laboratoire clips, je suis
           Mélina et vous ?
[Usager] : salut mélina c'est hoa
     Actes de l'utilisateur : FP[Ouverture(x)] & FS[Prenom(hoa)]
     Stratégie : coopérative
     But : Usager ; Status : Atteint ; Valeur : 2
     Actes du système : FS[Sexe(0)] & FS[Nom(nguyen)] & FS[Role(thésard)]
           & FS[Groupe(PVE)]
[Mélina] : bonjour monsieur nguyen, je vous écoute...
[Usager] : tu peux me réserver une salle demain martin vers dix heures
     Actes de l'utilisateur : FF[Action(reserver)] & FS[NomSalle(x)] &
           FS[Date(20041109)] & FS[Heure(600)]
     Stratégie : coopérative
     But : Salle ; Status : Nouveau but ; Valeur : null
     Actes du système : FFS[Salle(x)]
[Mélina] : quelles sont vos préférences de salle ?
[Usager] : j'en sais rien mail j'ai besoin d'une salle de trentes places
     Actes de l'utilisateur : FS[NomSalle(x)]
     Stratégie : coopérative
     But : Salle ; Status : Nouveau but ; Valeur : null
     Actes du système : FFS[Salle(x)]
[Mélina] : quelles sont vos préférences de salle ?
[Usager] : j'en sais rien mail j'ai besoin d'une salle de trente places
     Actes de l'utilisateur : FS[Taille(+30)]
     Stratégie : coopérative
     But : Reservation ; Status : Atteint ; Valeur : 2
     Actes du système : FS[NomSalle(lafayette)] & FS[Taille(45)]
[Mélina] : la salle lafayette serait à votre disposition, sa capacité est
           de 45 places,
[Usager] : parfait
     Actes de l'utilisateur : FS[Reservation(OK)]
     Stratégie : coopérative
     But : Materiel ; Status : Nouveau but ; Valeur : null
     Actes du système : FFS[Materiel(x)]
[Mélina] : quels matériels voulez vous réserver ?
```

```
[Usager] : ben un vidéo projecteur ça suffit alors
     Actes de l'utilisateur : FS[Materiel(vidéo projecteur)]
     Stratégie : coopérative
     But : Convocation ; Status : Nouveau but ; Valeur : null
     Actes du système : FFS[Convocation(x)]
[Mélina] : voulez vous que je prévienne les participants ?
[Usager] : oui les membres du projet pve et comme d'habitudes par
           téléphone
     Actes de l'utilisateur : FS[Convocation(OK)] & FS[Membre(pve)] &
           FS[Mode(téléphone)]
     Stratégie : coopérative
     But : Convocation ; Status : Satisfait ; Valeur : OK
     Actes du système : FS[Convocation(OK)]
[Mélina] : très bien, je vais prévenir les participants
[Usager] : merci bien mélina et à bientôt
     Actes de l'utilisateur : FS[Cloture(x)]
     Stratégie : réactive
     But : Cloture ; Status : En cours de traitement ; Valeur : x
     Actes du système : FS[Cloture(OK)]
[Mélina] : au revoir monsieur nguyen à bientôt.
```