Collaboration humain-machine à l'aide de motifs dialogiques pour la réalisation d'une tâche complexe : application à la recherche d'information

Jean-Baptiste Louvet

Normandie Université

THÈSE

Pour obtenir le diplôme de doctorat

Spécialité Informatique

Préparée au sein de l'Institut National des Sciences Appliquées Rouen Normandie

Collaboration humain-machine à l'aide de motifs dialogiques pour la réalisation d'une tâche complexe
Application à la recherche d'information

Présentée et soutenue par

Jean-Baptiste Louvet

Thèse soutenue publiquement le 10 juillet 2019 devant le jury composé de

Frédéric Landragin	Directeur de recherche CNRS, LATTICE, ENS Paris - Université Sorbonne Nouvelle – Paris 3	Rapporteur
Domitile Lourdeaux	Maître de conférences HDR, Heudiasyc, UTC	Rapportrice
Gaëlle Calvary	Professeure, LIG, Grenoble INP	Examinatrice
Maxime Morge	Maître de conférences, CRIStAL, Université de Lille	Examinateur
Laurent Vercouter	Professeur, LITIS, INSA Rouen Normandie	Directeur de thèse
Nathalie Chaignaud	Maître de conférences, LITIS, INSA Rouen Normandie	Co-encadrante
Guillaume Dubuisson Duplessis	Docteur, EDF	Invité
Jean-Philippe Kotowicz	Maître de conférences, LITIS, INSA Rouen Normandie	Invité

Thèse dirigée par Laurent Vercouter et Nathalie Chaignaud

INSA ROUEN NORMANDIE

MIIS

litis

Compilation du 26 août 2019 à 09:04
Version 1.4.3 – HEAD Commit Hash: 41f2ead90ec56e0fa2994e03b3c0f3f765a3726e
Option: publication

Résumé

La diffusion grandissante de l'intelligence artificielle et son extension à de nombreux domaines de nos vies modifient la relation que nous avons avec nos machines. Leur omniprésence dans notre quotidien et leur utilisation pour la résolution d'un nombre toujours plus important de problèmes amènent de nouveaux usages comme le partage entre un humain et une machine de la résolution d'un problème complexe. Cela introduit l'idée d'une collaboration humain-machine pour la résolution de problème, chacun réalisant les actions pour lesquelles il est le plus performant afin de construire une solution dépassant les capacités de l'humain ou de la machine pris indépendamment.

Cette vision d'un échange élaboré entre l'humain et la machine nous amène à nous intéresser au domaine de recherche de l'interaction humain-machine. Il est tout d'abord nécessaire de déterminer s'il existe des modalités d'interaction permettant de représenter les échanges humain-machine respectant les usages humains tout en permettant de supporter la réalisation collaborative d'une tâche. Pour résoudre ce problème, nous représentons l'interaction humain-machine à l'aide d'un modèle de comportement communicatif conventionnel basé sur l'observation d'un corpus d'interactions humain-humain.

En plus d'un modèle d'interaction humain-machine adapté, il est nécessaire de définir une représentation de la tâche permettant à la machine de contribuer utilement à sa réalisation. Cela pose deux problèmes : premièrement, *l'interprétation* d'une action de l'utilisateur, c'est-à-dire lier ce que fait l'utilisateur au contexte courant afin de suivre le déroulement de la tâche au travers de ses actions. Deuxièmement, *la prise d'initiative* par la machine, qui doit être capable de proposer des actions à l'utilisateur qui soient pertinentes avec l'interaction et constructives pour la tâche. Cela nécessite d'avoir une connaissance des interactions permises dans le contexte interactif actuel et des actions réalisables pour faire avancer la tâche.

Dans cette perspective, cette thèse propose un modèle conventionnel de la structuration d'une tâche collaborative humain-machine pour concevoir un système interactif assistant un humain sur la réalisation de tâches complexes. Plus spécifiquement, nous nous focalisons sur des tâches dont la résolution est hautement opportuniste et ne peut pas être planifiée. Nous introduisons un modèle de représentation de la tâche s'appuyant sur les *jeux de dialogue*, des motifs d'interaction dialogique permettant de décrire la structure de l'interaction à l'aide d'enchaînements d'actes de dialogue conventionnellement acceptables. Nous organisons ces motifs dialogiques en *états*, des structures décrivant les comportements attendus de la part de chaque interlocuteur au cours des différentes sous-tâches de la tâche collaborative. Ces états regroupent les motifs dialogiques en un ensemble cohérent vis-à-vis de la sous-tâche à laquelle ils sont associés et les enrichissent avec des règles localement cohérentes. Celles-ci permettent de décrire les effets de la réalisation d'un motif dialogique par les interlocuteurs dans un contexte particulier de la tâche. Les états permettent au système qui les utilise de lier un énoncé de l'humain à l'état actuel du dialogue et de la tâche et d'initier des comportements cohérents avec l'interaction et constructifs pour la tâche. Les décisions prises par le système sont basées sur la notion de *maturité*, une

valeur associée à chaque état représentant la capacité du système à prendre des initiatives dans cet état. Le modèle décisionnel du système est conçu pour être résilient et laisser un maximum de liberté à l'utilisateur.

Ce modèle est implémenté dans CoCoA, un système qui collabore avec un humain pour l'assister lors de la réalisation d'une tâche complexe. À partir de l'étude d'un corpus de dialogues humain-humain sur une tâche de recherche d'information collaborative médicale, nous décrivons la tâche à l'aide de notre modèle. Ce cas d'utilisation est implémenté à l'aide de CoCoA et une évaluation est réalisée en simulant le comportement d'un utilisateur ayant un besoin d'information et en faisant varier ses actions à partir de comportements identifiés dans le corpus.

Abstract

The increasing spread of artificial intelligence and its extension to new areas of our lives alter the relationship we have with our machines. Their omnipresence in our daily life and their use for an ever-increasing number of problems lead to new usages like the sharing of a complex problem solving process between a human and a machine. This introduces the idea of a human-machine collaboration for problem solving, each one performing the actions it is the best at to build a solution that goes beyond the capacity of the human or the machine taken independently.

This vision of an elaborated exchange between the human and the machine leads us to focus on the human-machine interaction domain. It is first necessary to determine if there are interaction modalities allowing representing human-machines interactions while respecting human usages and supporting the collaborative completion of the task. To solve the problem, we represent the human-machine interaction with a conventional communicative behavior model based on the observation of a human-human interaction corpus.

In addition to a suitable human-machine interaction model, it is necessary to represent the task in a way that makes the machine able to usefully contribute to it. This raises two issues: first, the *interpretation* of an action of the user, linking it to the current context in order to follow the task unfolding through his actions. Second, the *initiative taking* by the machine, which must be able to put forward to the user actions that are relevant with the interaction and helpful for the task. This requires knowing the allowed interactions in the current interactive context and the performable actions to move the task forward.

From this standpoint, this thesis offers a conventional model of the structure of a human-machine collaborative task to design an interactive system helping a human performing complex tasks. More specifically, we focus on tasks whose resolution is highly opportunistic and can not be planned. We introduce a task representation model based on *dialogue games*, dialogue patterns making it possible to describe the interaction structure with sequences of conventionally acceptable dialogue acts. We organize these dialogue patterns in *states*, structures describing the expected behaviors from each interlocutor during the different sub-tasks of the collaborative task. These states group dialogue patterns together in a coherent set in regard to the sub-task they are associated to and enrich them with locally relevant rules. They make it possible to describe the effects of the execution of a dialogue pattern by the interlocutors in a given context of the task. The states are used by the system to link an utterance of the human to the current state of the dialogue and of the task and to take the initiative of behaviors relevant with the interaction and helpful for the task. The decisions taken by the system are based on the concept of *maturity*, a value associated to each state representing the system's ability to take initiatives in this state. The decision model of the system is designed to be resilient and give as much freedom as possible to the user.

This model is implemented in CoCoA, a system that collaborates with a human to assist him performing a complex task. From the study of a human-human dialogue

corpus on a medical collaborative information retrieval task, we use our model to describe the task. This use case is implemented with CoCoA and an evaluation is performed by simulating a human user with an information need and varying its actions according to behaviors identified in the corpus.

À mes deux grands-pères.

Remerciements

Je tiens à remercier très chaleureusement mes rapporteurs et évaluateurs, pour le soin apporté à l'évaluation de mon manuscrit et la considération donnée à mes travaux de thèse. Vos compliments lors de ma soutenance m'ont donné confiance en moi et vos questions m'ont offert des perspectives nouvelles et très enrichissantes sur mes travaux.

Un très grand merci à Laurent, mon directeur de thèse, pour la confiance qu'il m'a accordé depuis le premier jour de mon doctorat. Tu as été un directeur exemplaire, disponible et à l'écoute quand j'en avais le plus besoin, capable de me guider dans toutes les étapes de mon doctorat et toujours prêt à me dire ce que je devais entendre, aussi désagréable que cela puisse être.

Merci à Nathalie et Jean-Philippe, qui me suivent depuis ma première année de cycle ingénieur. Merci pour vos enseignements, vos conseils et la liberté que vous m'avez laissé pour que je puisse faire toutes les erreurs nécessaires à mon apprentissage en GM et pendant ma thèse.

Merci au légendaire Guillaume, pour ses travaux sur les jeux de dialogue. Malgré la difficulté, j'ai eu beaucoup de plaisir à relever le défi de les reprendre et de les prolonger. Merci aussi pour tes précieux conseils qui m'ont beaucoup aidés à cheminer dans ce parcours initiatique qu'est la thèse. Tu as toujours su placer les bons mots aux bons moments pour me faire avancer sur problèmes avec lesquels j'étais aux prises, qu'ils soient scientifiques ou non. Tu constitues un exemple pour moi à bien des égards.

Merci à tous les membres du LITIS et de l'INSA que j'ai eu la chance de côtoyer pendant ces trois[1] années. Merci à tous les titulaires, administratifs, ATER, post-doctorants, doctorants, stagiaires avec qui j'ai partagé cette tranche de vie. J'en oublie forcément mais merci à Zaher, Mukesh, Guillaume, Fadila, Zina, Fabien, Benjamin, Rachel, Franco, Qiu, Sahba, Cyprien, Timothée, Ismaila, Soufiane, Antoine, Filipo, Brigitte, Sandra, Carole, Jean-François, Samia, Cécilia, Julien, Nicolas, Nicolas, Benoît, Daniel, Gilles, Géraldine, Alexandre, Habib, Stéphane, Stéphane, Laura, Mhamed, Jérôme. Merci à ceux qui ont prêté deux oreilles attentives à mes problèmes, merci aux trentenaires planificateurs, merci aux cramés qui venaient avec moi à l'entrainement, merci à ceux qui ont contribué aux barres de rire pendant les pauses et aux avancées sur l'incubaclodobus.

Un merci tout particulier à Mathieu, pour les encouragements, les blagues, les débats inutiles sans fin, les débats utiles sans fin, l'indexation des internets, le « c'est à cause de la faute de l'école », les arguments pour la légitimité d'abandonner un étudiant, la double contradiction pour simplement ne pas être d'accord avec son interlocuteur, les crêpes et les cookies. Je salue au passage le talentueux Vladimir Shuriken.

Merci à mes enseignants de l'INSA et d'ailleurs, qui m'ont éduqué et instruit années après années. Je sais que ce que vous m'avez transmis dépasse de très loin le cadre scolaire et que vous m'avez donné des enseignements qui me suivront tout au long de ma vie.

Merci à celles et ceux qui ont croisé ma route lors de mes études à l'INSA. On était côte à côte pour passer les heures froides et ennuyeuses dans les amphis de l'INSA, se serrer les coudes, profiter de la vie, avancer et grandir ensemble. Malgré la distance, malgré le

1. J'applique une approximation de physicien sur la durée exacte de mon doctorat.

temps qui passe, malgré nos chemins respectifs, on arrive à rester proches. Difficile de ne pas oublier du monde mais merci Fro, Langenais, Bioche, Aratz, Spider-Mario, Valentin, Amandine, Benoît, Mélanie, Thomo, Tanguy, Antoine, Massyl, Grabodon, Anca, Mathias, Zazou, Jeanne, Antoine, Marie, Doche, Hélène, Joss. C'est toujours un plaisir de vous retrouver et de constater que vous avancez, chacun à votre manière, sur votre chemin.

Merci à mes amis rencontrés à Rouen. J'ai pu m'épanouir grâce à des associations peuplées de belles personnes comme le café couture, alternatiba, guidoline, les jeunes écologistes, l'agnel, zéro déchet Rouen. Ici aussi j'en oublie forcément mais merci Sergine, Chloé, Lénou, Énora, Sabrina, Jon, Julie, Julie, Julie, Timothée, Émeline, Élodie, Morgane, Ludo, Théa, Pierre, Aude, Ruby, Sophie. Un merci tout particulier à Clotilde pour ton écoute, ta joie de vivre, le talent que tu as pour faire s'exprimer les autres de plein de manières et pour nous faire explorer toutes les facettes de notre personnalité. J'ai beaucoup de chance d'être entouré de personnes aussi exceptionnelles et précieuses.

Merci mon gros chat de m'avoir fait confiance. Merci pour ta présence, ton accueil, ton écoute, ta douceur, ta tendresse, ton rire, ton courage, ta force. Merci pour le bonheur et la joie que l'on partage. Merci de m'aider à avancer sur mon propre chemin. Merci de me demander de n'être rien d'autre que la meilleure version de moi-même.

Merci à ma famille, c'est un soutien indéfectible que de faire partie d'une tribu aussi soudée et solidaire. Merci pour votre appui et votre protection et merci d'avoir toujours été de mon côté. Merci à mes sœurs pour leur écoute, leurs conseils et pour m'avoir guidé dans la vie, chacune à sa manière. Merci à mes parents, qui m'ont soutenu dans toutes mes entreprises, même les plus délirantes [2]. J'ai eu tellement de chance de naître et de grandir dans votre foyer. Merci maman, pour l'exemple de courage, d'acceptation, de résilience et de calme que tu es pour moi face aux épreuves de la vie.

Merci à mon grand-père, le premier scientifique de ma vie, qui m'a transmis son insatiable curiosité, son envie d'apprendre, de savoir et de partager sa connaissance avec les autres.

Enfin, merci à On. J'ai traversé bien des épreuves pendant mon doctorat et tu as tout le temps été là pour moi. Ce n'est pas toujours facile et confortable de devenir une grande personne. Malgré les difficultés, les revers douloureux et le chaos de la vie, tu as invariablement fait en sorte d'avancer. Merci de ne jamais avoir renoncé, d'avoir compris qu'il n'est pas possible de fuir éternellement ses problèmes et que personne ne viendra les résoudre à ta place. Merci de faire face à tes démons à chaque fois que nécessaire et d'avoir toujours trouvé les ressources pour franchir les obstacles qui parsèment ton chemin. Que c'est bon d'être toi.

2. Faire un doctorat par exemple.

Table des matières

Introduction

Contexte

À l'origine réservés à un public restreint et à la résolution d'un petit nombre de problèmes, l'accès et l'utilisation de l'intelligence artificielle se banalisent chaque jour un peu plus. De nombreux domaines sont investis par des machines intelligentes qui transforment nos modes de vie et de pensée, que ce soit au travail, à la maison ou dans nos loisirs. La résolution d'un problème complexe peut être partagée entre les humains et les machines, chacun réalisant les actions pour lesquelles il est le plus performant, menant ainsi à la co-construction d'une solution au problème en dépassant les capacités de résolution de l'humain ou de la machine pris indépendamment.

Dans le même temps, la montée en puissance des technologies de l'information et de la communication et la complexification des systèmes que nous utilisons a fait croître la quantité d'information que nous avons à traiter. Ces avancées poursuivent le bouleversement du lien existant entre un être humain et les machines avec lesquelles il interagit, amorcé dans les années 60 avec l'émergence de l'interaction humain-machine comme domaine de recherche [Eng62, EE68]. Ce changement donne l'opportunité de continuer de faire évoluer la relation que les humains entretiennent avec les machines en offrant à celles-ci des capacités d'interaction et de prise d'initiative plus naturelles.

Dans cet objectif, les machines doivent être dotées de capacités collaboratives afin d'exposer un comportement proactif à leurs interlocuteurs humains. Cela implique de s'éloigner d'une vision de la machine comme une « boîte noire » résolvant un problème à la place de l'utilisateur et de situer l'interaction avec celui-ci au cœur du processus de résolution. Il ne s'agit pas simplement pour la machine de faire parvenir une information à l'utilisateur mais de la proposer au *bon moment* et de la *bonne manière* [Fis01].

Or actuellement, la collaboration entre humains et machines est entravée par les capacités de ces dernières à comprendre et à être comprises par les humains. Il est donc nécessaire d'enrichir les capacités d'interaction des machines en développant leurs aptitudes à reproduire et interpréter le comportement humain.

Ce problème est à l'interface de nombreux domaines et nécessite de relever des défis dans différentes disciplines, comme la linguistique avec le traitement automatique des langues, la robotique avec la création de machines humanoïdes, la simulation multi-agent avec la modélisation des organisations sociales ou l'interaction humain-machine avec l'établissement de méthodes d'interaction plus naturelles et reproduisant les interactions

humain-humain.

Prenons l'exemple de l'assistance sur une tâche de recherche d'information. Certaines ressources documentaires disponibles en ligne sont spécifiques à un domaine et les moyens permettant de les exploiter peuvent être difficiles à utiliser par un utilisateur quelconque. La technicité de l'outil pose problème à deux niveaux : premièrement, les outils de recherche d'information classiques ne sont pas adaptés à de tels corpus documentaires et il est nécessaire d'utiliser des outils spécifiques. Deuxièmement, une expertise du domaine est requise pour comprendre et exploiter facilement les résultats renvoyés.

À l'heure actuelle, un utilisateur ayant un besoin d'information mais n'étant pas capable d'exploiter ce type de ressource est contraint de faire appel à un expert humain afin de résoudre son problème. L'expert interagit avec l'utilisateur afin de comprendre son besoin d'information et s'engage avec celui-ci dans une tâche de recherche d'information collaborative pour répondre au besoin d'information.

Le passage de la réalisation collaborative d'une tâche humain-humain à une tache humain-machine pose problème à la fois sur le plan interactif, une machine n'ayant pas les capacités communicatives d'un être humain, et sur le plan de la réalisation de la tâche, une machine ayant des capacités de raisonnement et de résolution de problème très différentes de celles d'un expert humain.

Une approche possible pour résoudre ce problème consiste à s'inspirer de l'interaction de deux êtres humains lors de la résolution collaborative d'un problème difficile. Deux aspects peuvent être pris en compte. Le premier est le déroulement du processus de résolution du problème par les interlocuteurs. Une représentation de haut niveau de ce déroulement peut le rendre formellement représentable et ainsi permettre son implémentation au sein d'un système automatique.

Le second consiste en une analyse de l'interaction servant de support à la collaboration entre les interlocuteurs. En effet, celle-ci permet une représentation des régularités des échanges entre les interlocuteurs et rend possible une formalisation des motifs interactifs utilisés pour communiquer au cours de la réalisation de la tâche.

Ces deux aspects sont les deux faces d'une même pièce et il est nécessaire d'étudier les possibilités de lier la représentation de la résolution collaborative d'un problème avec l'interaction qui la supporte.

Problématiques scientifiques

La volonté de donner aux machines des capacités de collaboration modelées à partir de celles des humains soulève les problématiques suivantes, qui seront abordées dans ce manuscrit :

Problématique 1 : comment s'inspirer des interactions humain-humain pour modéliser l'interaction humain-machine ?

Il est utile de se pencher sur la possibilité de représenter formellement les échanges humain-humain afin d'améliorer les capacités interactives des machines. Même si l'interaction humain-humain est différente de l'interaction humain-machine, il est

intéressant de se questionner sur ce qu'il est possible d'en conserver pour l'intégrer à une machine.

Problématique 2 : comment représenter la structure d'une interaction humain-machine véhiculant la réalisation collaborative d'une tâche ?

L'utilisation d'un modèle formel d'interaction humain-machine permet l'amélioration des capacités interactives de la machine. Cependant, afin d'être réellement utile à l'humain, celle-ci doit aussi être capable de contribuer à la résolution de sa tâche. Cela nécessite l'introduction d'une représentation de haut niveau structurant l'interaction autour de la tâche qu'elle permet de résoudre.

Problématique 3 : comment la machine peut-elle tirer parti de la représentation de la tâche et de l'interaction qui la supporte pour interpréter les actions de l'utilisateur et décider de ses propres actions ?

Afin de compléter la représentation formelle de la structure de l'interaction permettant la réalisation d'une tâche humain-machine, il est nécessaire de décrire les processus décisionnels mis en œuvre par la machine pour tirer parti de la représentation de la tâche, interpréter de manière cohérente ce que fait l'utilisateur et y répondre de manière pertinente.

Problématique 4 : étant donné un modèle d'interaction humain-machine pour la réalisation collaborative d'une tâche, comment est-il possible d'évaluer la capacité de ce modèle à suivre correctement le déroulement d'une tâche ?

Le développement d'une application dans le domaine de la recherche d'information collaborative est un exemple d'application du modèle. Il permet de représenter la réalisation d'une tâche collaborative humain-machine et d'évaluer son déroulement selon divers scénarios que l'utilisateur peut suivre.

Notre contribution

Nous défendons dans ce manuscrit l'intérêt d'une description formelle du comportement dialogique conventionnel humain [Dub14]. Plus précisément, nous utilisons des *motifs dialogiques*, des normes sociales restreignant les actions réalisables par les interlocuteurs à des suites d'actions stéréotypées [Mau01]. Ces motifs sont utilisés comme modèles de l'interaction humain-machine tout en se détachant du traitement de la langue naturelle.

Cependant, ils se contentent de normer l'interaction et ne décrivent pas la manière dont elle véhicule l'avancement de la tâche sous-jacente. Leur utilisation n'est donc pas suffisante pour rendre une machine collaborative et il est nécessaire de les coupler à une capacité de prise de décision.

Cette articulation nécessite de trouver un équilibre entre l'aspect conventionnel de l'interaction, capturé par ses motifs d'interaction, et sa structuration intentionnelle, représentant les actions réalisées par les interlocuteurs pour résoudre la tâche. Nous proposons donc une formalisation de tâches collaboratives humain-machine à l'aide de motifs dialogiques en regroupant ceux-ci en structures cohérentes. Pour ce faire, le concept *d'état de la tâche* organise et enrichit les motifs dialogiques conventionnellement attendus de la part

des interlocuteurs pour la réalisation d'une sous-tâche donnée. Ces états sont utilisés lors des raisonnements de notre système, à la fois pour interpréter les actions de l'utilisateur et pour prendre des initiatives pertinentes au niveau de l'interaction et constructives au niveau de la tâche. Cela permet de distinguer clairement les traitements liés à la gestion de l'interaction avec l'utilisateur et ceux liés à la réalisation de la tâche.

En plus d'un modèle théorique nous donnons une représentation technique détaillée du système et de son implémentation ainsi que sa mise en œuvre sur une tâche de recherche d'information et terminons en exposant une évaluation de ce système.

Plan du manuscrit

Ce manuscrit se divise en trois parties, chacune composée de deux chapitres. La première partie est un état de l'art pour contextualiser nos travaux dans le paysage scientifique. La deuxième partie présente la contribution défendue par cette thèse et donne une perspective technique sur sa mise en œuvre. La troisième partie donne un cas d'utilisation de notre modèle de collaboration sur une tâche de recherche d'information collaborative médicale et son évaluation expérimentale. Le contenu de chaque chapitre est brièvement décrit dans les paragraphes suivants.

Chapitre 1 – Réalisation collaborative de tâche Ce chapitre présente un état de l'art sur la collaboration et plus spécifiquement la collaboration humain-machine pour la réalisation d'une tâche. Nous y introduisons le concept *d'agent* et spécifions les caractéristiques d'un *agent collaboratif*. Nous présentons ensuite deux stratégies que peut mettre en œuvre un agent collaboratif pour interagir avec un être humain : les approches par plan et les approches cognitives. Nous terminons ce chapitre sur la recherche d'information collaborative, qui constitue l'exemple applicatif de cette thèse.

Chapitre 2 – Modèles formels des interactions dialogiques Ce chapitre présente un état de l'art sur la modélisation formelle des interactions dialogiques. Nous y donnons dans un premier temps des généralités sur le dialogue. Ensuite nous décrivons les approches modélisant le dialogue comme une structure résultant des *intentions* des participants au dialogue. Puis nous présentons les approches *conventionnelles*, qui considèrent que le dialogue est organisé autour de normes sociales contraignant sa structure à des motifs stéréotypés. Nous introduisons la structure des *jeux de dialogue* et montrons comment ceux-ci peuvent être utilisés dans une approche mixte entre intentionnel et conventionnel.

Chapitre 3 – Modèle formel de la collaboration humain-machine Ce chapitre présente notre contribution principale. Il introduit notamment les états de la tâche, des structures basées sur les jeux de dialogue permettant d'articuler motifs dialogiques et réalisation d'une tâche sous-jacente au dialogue. Ces tables représentent un contexte local particulier définissant les comportements conventionnellement possibles entre un agent et un humain. Le comportement intentionnel de l'agent est défini par la notion de *maturité* d'un état. Celle-ci permet à l'agent d'interpréter un acte de dialogue de l'utilisateur

et de décider de quel comportement adopter et de l'enchaînement des différents états représentant une tâche.

Chapitre 4 – Implémentation dans l'architecture CoCoA Ce chapitre présente CoCoA, le système implémentant le modèle formel décrit dans le chapitre précédent. Ce système s'organise autour d'un *état d'information* dont la partie publique correspond à la représentation du dialogue avec l'utilisateur et dont la partie privée contient les informations nécessaires à la représentation de la tâche et aux prises de décision par le système. Différentes perspectives techniques sont données pour illustrer la mise en œuvre de notre modèle dans CoCoA.

Chapitre 5 – Application à une recherche d'information assistée Ce chapitre présente l'application du modèle sur une tâche de recherche d'information. Il décrit la structure de la tâche de recherche d'information collaborative médicale issue de l'étude d'un corpus d'interactions humain-humain. Nous introduisons ensuite la représentation de cette tâche à l'aide de notre formalisme et discutons des propriétés de la tâche conservées malgré le passage d'une interaction humain-humain à une interaction humain-machine.

Chapitre 6 – Mise en œuvre expérimentale Ce chapitre présente une évaluation expérimentale de notre modèle basée sur l'exemple de recherche d'information collaborative médicale précédemment donné. Nous utilisons la structure de la tâche de recherche d'information collaborative médicale humain-humain précédemment décrite pour définir le comportement conventionnel d'un utilisateur humain interagissant avec notre système. Nous introduisons différentes altérations apportées à ce comportement conventionnel correspondant à des situations observées dans le corpus d'interaction humain-humain et évaluons les réactions de notre système face à ces altérations.

Nous terminons ce manuscrit en reprenant les problématiques posées et en donnant des perspectives sur les évolutions pouvant être apportées à nos travaux.

Première partie

État de l'art

Chapitre 1

Réalisation collaborative de tâche

« La pierre n'a point d'espoir d'être autre chose que pierre.
Mais, de collaborer, elle s'assemble et devient temple. »

Antoine de Saint-Exupéry, Citadelle – LXXXVII

Sommaire

Dans ce chapitre, nous présentons sous quelles perspectives il est possible d'utiliser un ordinateur comme outil pour la réalisation de tâches collaboratives. En effet, nous pouvons espérer voir les machines nous assister de manière constructive et pertinente dans la réalisation de tâches qui nous sont difficiles. Celles-ci dépasseraient alors la métaphore de l'outil servant de support ou d'intermédiaire à l'interaction humain-humain pour devenir des interlocuteurs à part entière participant à une tâche partagée avec un ou plusieurs utilisateurs.

L'orientation suivie par les approches collaboratives humain-machine partent du principe que les humains et les machines ont des compétences complémentaires. Plutôt que de les mettre en concurrence, cette complémentarité permet de développer des situations où les humains et les machines vont s'apporter un soutien mutuel. En d'autres termes, les machines collaboratives n'ont pas vocation à remplacer des êtres humains mais à réaliser une tâche avec eux.

Dès le début des années 70, Negroponte [Neg70] a donné une vision de la manière dont des humains et des machines « intelligentes » peuvent s'associer pour concevoir des projets architecturaux. Il imagine des machines capables d'apprendre et de s'adapter en dépassant le paradigme de l'exécution répétitive de tâches automatiques. La relation entre un humain et une machine ne serait donc plus de maître à esclave, mais entre deux associés permettant un développement et un apprentissage mutuel. Pour lui, l'intérêt de coopérer avec une machine est la complémentarité des points de vue et des traitements que chacun peut apporter au problème à résoudre, l'humain et la machine ayant des représentations et des modes opératoires radicalement différents.

Kay [Kay89] est l'un des premiers a avoir eu l'intuition d'une communication avec des machines qui agiraient comme des guides, des entraîneurs ou des assistants dans ce qu'il a appelé une *grande collaboration*. La vision de Kay est de tenir compte des processus cognitifs de l'utilisateur lors de la conception de l'interface d'un programme pour améliorer la communication entre le programme et l'utilisateur.

Ce point de vue a été prolongé par Epstein [Eps15], pour qui il est nécessaire de tirer des ponts entre l'intelligence artificielle et les sciences cognitives pour permettre aux machines de collaborer efficacement avec une personne. Cela dans un but de permettre aux machines de tenir compte de la manière dont les humains fonctionnent pour s'adapter et à communiquer au mieux.

La collaboration humain-machine dépasse donc le paradigme du CSCW [1], qui se borne à la conceptions d'outils collaboratifs pour le travail intermédié par ordinateur.

Dans la section 1.1, nous présentons différentes définitions de la collaboration et adoptons la nôtre. Dans la section 1.2, nous montrons de quelle manière il est possible d'envisager une collaboration entre un humain et un système informatique. Dans la section 1.3, nous présentons deux approches utilisées pour rendre un système collaboratif : les approches par plans et les approches cognitives. Dans la section 1.4, nous illustrons concrètement les possibilités de collaboration sur une tâche de recherche d'information.

1. Pour « Computer Supported Collaborative Work », travail collaboratif assisté par ordinateur.

1.1 Définition de la collaboration

La caractérisation de la collaboration et l'identification des ses différentes formes et propriétés a fait l'objet de nombreuses études dans des domaines variés comme les sciences sociales, la psychologie et l'intelligence artificielle. Préalablement à la présentation de notre vision de la collaboration, nous définissons le concept d'interaction :

Définition 1.1 – Interaction : « Les interactions sont des événements réciproques nécessitant au moins deux objets et deux actions. Elles ont lieu quand ces objets et événements s'influencent mutuellement. » ([Wag94], p.8)

Tout échange qui n'est pas à sens unique entre au moins deux entités et ayant une influence sur ces entités est donc perçu comme une interaction.

Nous cherchons maintenant à décrire ce qui fait la particularité de la collaboration en tant qu'interaction. Ses définitions et les concepts qui s'y rattachent ne font pas forcément consensus au sein des différentes communautés de la recherche [WG91] et sont parfois contradictoires. Nous essayons dans un premier temps, en section 1.1.1, de donner une définition générale de la collaboration et des concepts qui y sont liés, en la différenciant des notions de communication, contribution, coopération et de coordination, dans un contexte humain-humain. En section 1.1.2, nous nous intéressons aux modalités de mise en œuvre de la collaboration, ainsi qu'à ses limites et à ce qui peut l'entraver.

1.1.1 Concepts liés à la collaboration

Le bénéfice apporté par la collaboration peut se rattacher au concept d'intelligence collective, qui suppose que la capacité à prendre de bonnes décisions émerge d'un groupe plutôt que de quelques personnes éclairées [Sur05]. Autrement dit, la mise en commun de différents points de vue et expertises individuelles apporte un profit mutuel [FBP+00]. Les problèmes difficiles, interconnectés et évolutifs nécessitent la création d'une relation élaborée et dynamique entre ceux qui essaient de les résoudre, amenant à la formation d'un groupe dont le résultat dépasse le cumul des contributions individuelles [TPRG98]. C'est ce qui est décrit par Gray [Gra89] comme établissant « un compromis qui permet aux parties prenantes de concevoir des solutions qui n'auraient pas été accessibles si elles avaient travaillé indépendamment » (cité dans [Sha12]) et est résumé par Shah et González-Ibáñez sous le nom de *synergie* [SGI11].

Roberts [Rob00] classifie les problèmes en trois degrés de difficulté :

Un problème simple – sa définition et sa solution font l'objet d'un consensus entre toutes les personnes concernées ;

Un problème complexe – sa définition fait l'objet d'un consensus mais les solutions et leurs mises en œuvre sont sources de conflits ;

Un problème « confus » ou « épineux » [2] – sa définition n'est ni définitive ni unanime et est sujette à controverse. La recherche de solution est ouverte et change

2. Traductions de « messy » et « wicked. »

selon la définition du problème et celui qui cherche une solution. Le processus de résolution ainsi que les ressources disponibles sont contraints et changent constamment. En cours de résolution, les parties prenantes peuvent s'associer et se dissocier du problème et de sa résolution, changer d'avis et échouer à communiquer.

London [Lon95] ajoute des caractéristiques aux problèmes de cette dernière catégorie : les parties prenantes sont interdépendantes et ne sont pas clairement identifiées ou organisées à priori. La répartition du pouvoir ou des ressources entre leurs mains est hétérogène. Leur degré d'expertise et niveau d'accès à l'information du problème est variable. Les différentes perspectives sur le problème mènent à des relations conflictuelles. Le problème est souvent caractérisé par une complexité technique et une incertitude scientifique. Les efforts unilatéraux ainsi que les méthodes existantes pour résoudre le problème produisent des résultats qui ne sont pas satisfaisants.

Face à un problème de cette catégorie, le travail collaboratif n'est pas forcément la première option envisagée et est précédé par des méthodes compétitives ou autoritaires [DY08, Rob00]. C'est seulement une fois que ces deux méthodes ont échoué que « les gens tombent dans la collaboration » [3] [Rob00].

Même si certains auteurs utilisent indifféremment « collaboration », « coopération » et « coordination » pour se référer à un même concept [Sha14], d'autres cherchent à nuancer leur définition.

Taylor-Powell *et al.* [TPRG98] donnent le modèle des « 5-C » : *communication*, *contribution*, *coordination*, *coopération* et *collaboration*. Ce modèle reprend la définition de Gray en y ajoutant la *mise en œuvre* des solutions trouvées. Ces cinq notions reflètent des degrés croissants d'implication des différentes parties dans un processus réalisé en commun, comme représenté en figure 1.1.

Les auteurs donnent les définitions suivantes pour les concepts du modèle :

Communication – processus basé sur l'échange d'informations et de sens. C'est par la communication que s'établit l'interaction entre les interlocuteurs.

Contribution – relation informelle entre les parties au cours de laquelle celles-ci s'apportent des ressources et du soutien nécessaires pour atteindre leurs objectifs, indépendants de ceux des autres.

Coordination – un processus de communication, de planification, de partage de ressources, de risques et de récompenses dans un but d'efficience et d'efficacité pour atteindre les buts complémentaires des parties concernées.

Coopération – un processus où les parties ayant des intérêts similaires planifient ensemble, négocient des rôles mutuels et partagent des ressources pour atteindre des buts communs tout en conservant des identités distinctes.

Dans une définition largement répandue, Gray [Gra89] définit la collaboration :

Définition 1.2 – Collaboration : « un processus par lequel les parties qui voient différents aspects d'un problème peuvent explorer constructivement leurs différences et

3. Traduction de « people fail into collaboration ».

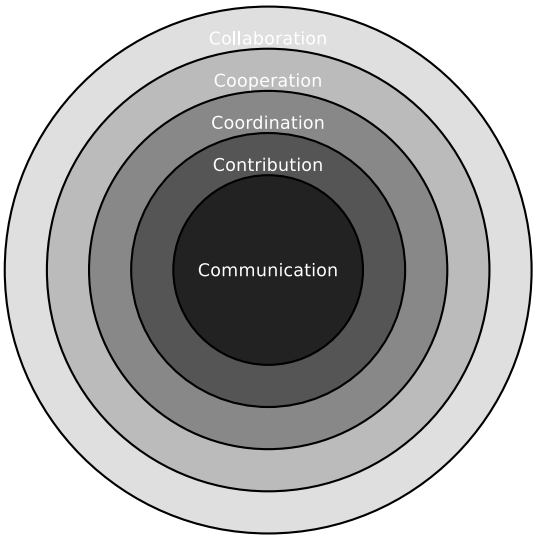

FIGURE 1.1 – Synthèse des niveaux d'engagement dans un processus commun entre plusieurs parties (représentation réalisée par Shah [Sha10], adapté des travaux de Taylor-Powell *et al.* [TPRG98], p.4-5). L'établissement d'un niveau d'engagement correspondant à un cercle est supporté par les niveaux d'engagement représentés par les cercles qui sont plus petits.

chercher des solutions qui dépassent leurs propre vision limitée de ce qui est possible » (cité dans [WG91] et [Lon95]).

La collaboration tirerait donc partie de la variété des points de vue de ceux qui y prennent part, pour construire une solution qui serait hors de portée autrement. Cela correspond à une « gestion constructive des différences » [Gra89].

Ces concepts sont liés par un degré croissant d'organisation entre les parties, la contribution étant ad hoc et informelle, alors que la coordination, la coopération et la collaboration nécessitent des échanges plus formels et protocolaires, incluant de plus la notion de performance dans leur mise en œuvre. La différence s'établit aussi sur le niveau de similarité des buts des différentes parties, la contribution et la coordination impliquant des buts complémentaires alors que la coopération et la collaboration impliquent des buts communs. Cela amène à une gradation de l'interdépendance des parties, celle-ci augmentant avec leur implication dans le processus commun.

Le degré d'implication de chaque partie se renforce avec le temps passé à la réalisation du processus commun. En effet, celui-ci peut commencer de manière simple et informelle et s'organiser à mesure que les parties s'investissent dans sa réalisation. Cela entraîne que l'établissement d'un concept à fort degré d'implication nécessite l'établissement des concepts d'implication plus faible. De cette manière, chaque concept du modèle « 5-C » sert de support à la mise en place du concept de degré supérieur [Sha10, Sha15]. La collaboration est donc un processus à long terme qui dépend des processus de communication, contribution, coordination et coopération qui lui sont associés.

Le tableau 1.1 synthétise les différentes caractéristiques de chaque niveau du modèle en « 5-C ».

Concept	Caractéristiques
Communication	Échange d'information et de sens
Contribution	Communication + mise à disposition de ressources
Coordination	Contribution + efficacité
Coopération	Coordination + buts communs
Collaboration	Coopération + utilisation constructive des différences

TABLEAU 1.1 – Synthèse du modèle en « 5-C » [TPRG98].

Cette vision des différents degrés d'implication des participants dans un processus commun est aussi utilisée par Denning et Yaholkovsky [DY08], qui définissent la collaboration comme « la forme la plus synergique pour travailler ensemble », la coordination et la coopération en étant des formes plus faibles.

1.1.2 Mise en œuvre et limitations de la collaboration

Même si la collaboration est une option intéressante pour résoudre certaines catégories de problèmes, elle n'est pas envisageable de manière systématique et peut parfois s'avérer contre-productive. En effet, sa mise en œuvre nécessite de respecter certaines contraintes qui peuvent s'avérer couteuses et difficiles à réaliser.

Mise en œuvre

Terveen [Ter95] décrit ce qu'il est nécessaire de mettre en œuvre pour réussir une collaboration :

Se mettre d'accord sur les buts communs – les agents doivent déterminer les buts qu'ils poursuivent par discussion explicite ou par inférence sur des énoncés et des actions des autres participants. Généralement les agents n'atteignent pas une définition complète et non ambiguë de leur buts avant de commencer la résolution du problème. La mise en œuvre de la résolution les mène souvent à spécifier ou reformuler leurs buts ;

Planifier, distribuer la responsabilité, se coordonner – les agents doivent décider comment réaliser leurs buts, quelles actions chacun aura à réaliser et comment les coordonner ;

Avoir un référentiel commun – les agent doivent être capables de suivre ce qui a été réalisé et ce qu'il leur reste à faire. Ils doivent évaluer les effets de leurs actions et déterminer si une solution acceptable a été trouvée ;

Communiquer – les agents doivent échanger des informations et s'observer mutuellement pour définir des buts communs et évaluer les progrès et résultats de la collaboration ;

S'adapter – la collaboration peut avoir lieu sur le court terme (au sein d'une seule session de travail) ou le long terme (plusieurs sessions) et nécessite une adaptation mutuelle des interlocuteurs.

Les deux premiers aspects seront développés en section 1.3. Le chapitre 2 de ce manuscrit défend une vision du dialogue comme outil de communication collaboratif.

Les contraintes apportées par la mise en œuvre d'une collaboration peuvent venir entraver son bon déroulement et la faire échouer, empêchant la résolution du problème sous-jacent. Il existe principalement deux contre-indications à la collaboration pour la résolution d'un problème : quand celui-ci est *trop simple* et quand la collaboration ne peut pas avoir lieu dans de *bonnes conditions*.

Nous venons de présenter ce qui permet, dans un contexte général, de caractériser une interaction comme étant une collaboration. Utilisé dans des conditions favorables, c'est un processus qui peut être largement avantageux pour les entités y prenant part. Cependant, pour être efficace elle nécessite un degré important d'organisation entre ses participants et sa mise en œuvre peut échouer quand les conditions ne sont pas propices.

1.2 Collaboration humain-machine

Nous avons précédemment discuté de la collaboration dans un sens large et tournée vers une interaction humain-humain. Cette section présente les aspects que revêt une collaboration humain-machine. Cette perspective est motivée par le fait que le perfectionnement des machines les a fait passer en quelques années du statut d'objet passif inanimé exécutant des instructions ordonnées par un utilisateur au statut d'entités capables de prendre des initiatives et d'apporter une aide active à l'utilisateur [JMN+14]. La collaboration avec des machines présente l'avantage de mettre à la disposition de l'humain des capacités d'accès, de traitement et de stockage de l'information qui lui sont inaccessibles. L'interaction humain-machine ne se limiterait plus à la délégation de tâches rébarbatives ou fastidieuses à la machine et prendrait des formes plus complexes où celle-ci serait capable d'interpréter les actions de l'utilisateur et de prendre des initiatives pour l'aider à atteindre ses buts.

Le domaine de l'intelligence artificielle a fait apparaître dans les années 90 des entités logicielles appelées *agent*. Wooldridge définit un agent de la manière suivante :

Définition 1.3 – Agent : « Un *agent* est un système informatique *situé* dans un *environnement* et capable d'*agir de façon autonome* dans cet environnement de manière à atteindre les buts pour lesquels il a été conçu »([Woo02], p.15).

En appliquant cette définition à une interaction humain-agent, *l'environnement* dans lequel est situé l'agent est l'environnement de travail de l'humain avec lequel il interagit. L'agent est doté d'une capacité de *prise de décision et d'action autonome* sur cet environnement, au même titre que l'humain.

Il existe différentes classes d'agents, selon leurs fonctions, architectures et langages. Nous ne traitons pas en détails ces différentes classes et nous contentons de présenter celles qui permettent d'envisager une collaboration humain-machine. Pour trouver une définition et une typologie complète du concept d'agent, le lecteur peut se tourner vers les travaux de Wooldridge et Jennings [WJ95] ou de Nwana [Nwa96].

La section 1.2.1 présente les *agents autonomes*, la section 1.2.2 présente les *agents d'interface*, la section 1.2.3 présente les *agents autonomes d'interface*, la section 1.2.4

présente les *agents collaboratifs* et la section 1.2.5 discute des problèmes que la collaboration humain-machine soulève.

1.2.1 Les agents autonomes

Un *agent autonome* est un agent capable de prendre des décisions en autonomie, dont le fonctionnement est inaccessible au monde extérieur et dont les actions n'ont pas vocation à être rendues visibles pour les autres membres de l'environnement partagé. L'autonomie d'un agent vient du fait qu'il n'est pas constamment en attente d'une action ou sollicitation de l'utilisateur [Cas95] et exécute des tâches indépendamment de celui-ci. Un agent autonome interagissant avec un humain ne nécessite pas une supervision permanente pour fonctionner. Cette propriété permet à un utilisateur de lui déléguer des activités pour focaliser son attention sur d'autres tâches.

Cependant, le recours à un agent autonome peut impliquer une diminution de la perception de l'environnement par l'utilisateur. En effet, un agent autonome n'a pas vocation à interagir avec un humain ou à rendre publique toutes les actions qu'il réalise. Les actions qu'un agent autonome réalise et le processus qui l'y a conduit restent opaques pour les autres agents évoluant dans le même environnement et pour l'utilisateur. Cela présente le risque pour l'humain de ne pas avoir connaissance des modifications faites par l'agent sur l'environnement partagé. La conscience qu'a l'utilisateur de la situation dans laquelle il se trouve peut être altérée par une manipulation de l'environnement de la part de l'agent.

1.2.2 Les agents d'interface

Un des premiers paradigmes d'interaction avec les machines était la *manipulation directe* de l'interface [Shn83]. Cette méthode nécessite des instructions ou des commandes directes de la part d'un utilisateur pour modifier des artefacts de l'interface du système. Elle permet à l'utilisateur de connaître toutes les actions réalisées sur l'interface et leurs répercussions sur les objets qu'il manipule. Cependant ce paradigme peut nécessiter un degré d'expertise important pour utiliser les fonctionnalités les plus avancées d'une application, et rester difficile d'accès pour des novices. De plus, la manipulation directe nécessite de la part de l'utilisateur de répéter la même action ou la même commande autant de fois qu'il souhaite l'appliquer, entraînant un travail rébarbatif, fastidieux et à faible valeur ajoutée.

Les *agents d'interface* sont une deuxième classe d'agents qui, comme l'utilisateur, réalisent des actions sur les représentations visuelles de l'interface graphique. À la différence d'une manipulation directe de l'interface par l'utilisateur, ces actions ne sont pas réalisées sur instruction directe de celui-ci. Toutes les actions d'un agent d'interface sont *publiques* et celui-ci agit en réaction d'une ou plusieurs actions réalisées par l'utilisateur.

1.2.3 Les agents autonomes d'interface

Lieberman présente un modèle d'agent réconciliant les agents autonomes et les agents d'interface, qu'il nomme *agents autonomes d'interface* [Lie97]. Ce sont des agents qui, tout en réalisant des actions en privé et en fonctionnant de manière asynchrone vis-à-vis de l'utilisateur, sont capables de manipuler directement des éléments de l'interface. Cette vision est à rapprocher de la définition d'agent d'interface donnée par Nwana [Nwa96] et de la métaphore *d'assistant personnel numérique* de Maes [Mae94].

Les agents autonomes d'interface donnent à l'utilisateur la possibilité de se détacher de la manipulation directe de l'interface. Il est en effet possible de leur déléguer des corvées nécessaires mais ennuyeuses et répétitives. Les agents autonomes d'interface ne s'intercalent pourtant pas entre l'utilisateur et l'application, mais agissent comme des collaborateurs que l'utilisateur peut contourner ou ignorer s'il le souhaite. L'utilisateur garde la main sur l'interface et peut la manipuler de manière directe, mais l'agent observe « par-dessus son épaule » et peut intervenir auprès de lui à n'importe quel moment.

Un exemple d'agent autonome d'interface est Letizia [Lie95, Lie97], qui interagit avec un utilisateur naviguant sur le web. Letizia collecte en temps réel les URL visitées par l'utilisateur pour tenter d'anticiper sur ce qui pourrait l'intéresser. Il explore et traite les pages liées à celles visitées par l'utilisateur pour lui proposer dynamiquement des options de navigation. L'intérêt est qu'il suggère activement des pages à visiter sans requête ou intervention de l'utilisateur, contrairement aux moteurs de recherche qui nécessitent d'être explicitement sollicités par celui-ci. La partie autonome de Letizia explore le web à la recherche de ressources pertinentes et la partie interface met à jour l'interface graphique pour les proposer à l'utilisateur.

Les agents autonomes d'interface sont capables d'apprendre et de s'adapter à l'utilisateur de différentes manières. La première est d'observer celui-ci en essayant de repérer les motifs qui se répètent dans son comportement. La deuxième est de modifier son comportement en fonction des retours de l'utilisateur. La troisième est d'être explicitement entraîné par l'utilisateur, montrant à l'agent des exemples. Une dernière manière consiste à solliciter d'autres agents autonomes d'interface aidant d'autres utilisateurs [Mae94, LMM98]. Cela permet aux nouveaux agents de ne pas avoir à ré-apprendre des comportements qui auraient déjà été appris par d'autres agents.

Pour synthétiser les classes d'agents qui viennent d'être données en les rapprochant du modèle en « 5-C » de Taylor-Powell, un *agent autonome* ne présente *aucune caractéristique* lui permettant de réaliser une collaboration avec l'utilisateur. Plus précisément, un agent autonome n'étant pas doté de capacités de communications avec l'utilisateur, il lui est impossible d'échanger de l'information et du sens avec celui-ci et donc d'établir les processus de contribution, coordination, coopération et de collaboration. Les *agents d'interface* sont quand à eux capables de mettre en place une *communication* avec l'utilisateur et respectent les conditions pour établir le premier niveau du modèle de collaboration de Taylor-Powell. En effet, les actions d'un agent d'interface sont publiques et toujours visibles par l'utilisateur et réciproquement, l'agent d'interface est capable de capter les actions réalisées par l'utilisateur. Les *agents autonomes d'interface* présentent quant à eux les caractéristiques nécessaires pour apporter une *contribution* à la tâche de l'utilisateur. Un

agent autonome est en effet capable de mettre à disposition de l'utilisateur des ressources l'aidant à atteindre ses objectifs. De plus, cette relation peut être étendue à une *coordination* car l'utilisateur peut tirer parti de l'agent pour planifier la réalisation efficace de sa tâche en lui déléguant certaines sous-tâches et en se focalisant sur d'autres sous-tâches.

Les agents autonomes d'interface semblent être un modèle prometteur pour interagir avec un être humain. Cependant, comme nous venons de le montrer, ceux-ci n'ont pas la capacité de réaliser pleinement une collaboration avec un être humain et se limitent au mieux à une coordination avec celui-ci. Il est donc nécessaire dans un premier temps de doter un agent autonome d'interface de capacités de coopération.

1.2.4 Les agents collaboratifs

Pour être pleinement collaboratif, au sens de Taylor-Powell, un agent doit être capable de faire une utilisation constructive des différences entre l'utilisateur et lui. De ce fait, il est nécessaire pour l'agent de disposer d'une représentation de l'utilisateur ou du moins de ses processus décisionnels. Cette représentation permet à l'agent de devancer les besoins de l'utilisateur et de lui proposer des solutions calculées en faisant usage de ses propres ressources. Cette situation peut être particulièrement bénéfique pour l'utilisateur, étant donné que ses capacités d'accès et de traitement de l'information sont radicalement différentes de celles de l'agent. La collaboration humain-machine tire parti de cette complémentarité, en permettant à l'agent de proposer à l'utilisateur des options qui d'ordinaire lui sont hors de portée.

Cette approche *par complémentarité* est habituellement opposée à une approche *par émulation* [Ter95]. Celle-ci vise à améliorer la collaboration entre les humains et les agents en donnant à ces derniers des capacités similaires à celles de humains. Néanmoins, l'approche par émulation n'a eu qu'un succès limité du fait de sa difficulté de mise en œuvre et celle par complémentarité s'est avérée plus tractable et viable [Fis01].

Terveen propose cependant une synthèse de ces deux approches, en montrant qu'elles ne s'opposent pas totalement. Une approche unifiée est proposée, recoupant les deux précédentes [Ter95]. Elle implique les caractéristiques suivantes :

Communication naturelle – comprendre les aspects fondamentaux de la communication humaine et développer des équivalents pour la communication humain-machine utilisant les propriétés spécifique de la machine ;

Équilibrage entre représentation, raisonnement et interaction – trouver un équilibre approprié entre les raisonnements du système et les interactions avec l'utilisateur ;

Adaptation collaborative – rendre collaboratif le processus d'adaptation du système à un utilisateur, chacun jouant un rôle approprié ;

Réification – rendre les connaissances, la résolution du problème et la structure de l'interaction *visibles*, *partagées* et *manipulables* par le système et l'utilisateur.

La réification est d'autant plus importante que l'environnement dans lequel sont situés l'humain et l'agent évolue au cours de la réalisation de la tâche collaborative.

Cet environnement constitue une partie du *référentiel commun* de Terveen présenté en section 1.1.2 comme l'un des points clé de la mise en œuvre d'une collaboration et partagé par les collaborateurs. Le référentiel commun est une référence nécessaire pour leur communication, l'établissement de buts partagés et la mise en œuvre des plans de résolution.

La bonne perception du référentiel commun permet à l'humain d'avoir une *sensibilité à la situation*[4]. À l'échelle individuelle, Endsley [End95b, End95a] définit la sensibilité à la situation comme « une compréhension de l'état de l'environnement » et a montré qu'elle est constituée de trois niveaux :

La perception de l'environnement – c'est la reconnaissance d'éléments de l'environnement ainsi que leur statut, leurs attributs et leurs dynamiques ;

La compréhension de la situation actuelle – c'est une synthèse fondée sur des éléments disjoints du premier niveau pour créer une représentation holistique de l'environnement, en donnant un sens aux objets et aux événements ;

La projection dans des états futurs – c'est une anticipation sur les actions futures des éléments de l'environnement à l'aide des deux premiers niveaux.

Hoc [Hoc01] reproche à cette définition de considérer la situation comme un objet purement extérieur aux participants. Il propose d'y intégrer les participants eux-mêmes, ainsi que leurs plans, leurs ressources et les risques qu'ils encourent. Cette vision de la sensibilité à la situation offre la possibilité de représenter les buts et intentions des participants, ainsi que leurs charge mentale et leur charge de travail et le contrôle qu'ils ont sur la situation.

En synthèse, un agent collaboratif est un agent autonome d'interface observant les actions de l'utilisateur et capable de prendre l'initiative d'une interaction avec celui-ci afin de lui proposer des actions lui permettant de poursuivre ses buts, tout en lui laissant la capacité de prendre des initiatives par lui-même. Cet équilibre entre capacités réactives et directives est appelé *initiative mixte* [Hor99, Hor07]. Horvitz définit l'initiative mixte de la manière suivante :

Définition 1.4 – Initiative mixte : « L'initiative mixte est globalement une méthode permettant un entrelacement efficient et naturel des contributions faites par les utilisateurs contribuant à se rapprocher d'une solution à un problème. » ([Hor99] p.1)

Un agent collaboratif doit donc être capable d'initiative mixte en tenant compte de la situation de l'utilisateur pour intervenir et proposer des actions pertinentes en fonction de la tâche à réaliser, des ressources, états mentaux, processus cognitifs, et risques de l'utilisateur. Un agent collaboratif doit de plus être doté d'une capacité de communication pour capter et interpréter les actions de l'utilisateur, et pour les contextualiser dans la tâche à réaliser.

La figure 1.2 fait le lien entre les différentes classes d'agents et le modèle de la collaboration en « 5-C » de Taylor-Powell.

4. Traduction de « situation awareness ».

Il est important de noter que notre définition des « agents collaboratifs » diffère de celle de Nwana [Nwa96]. En effet, pour Nwana un agent collaboratif doit être capable de coopérer avec d'autres agents pour réaliser ses buts. Cette définition est à relier au concept *d'intelligence artificielle distribuée* et de collaboration agent-agent plutôt qu'à la collaboration humain-machine.

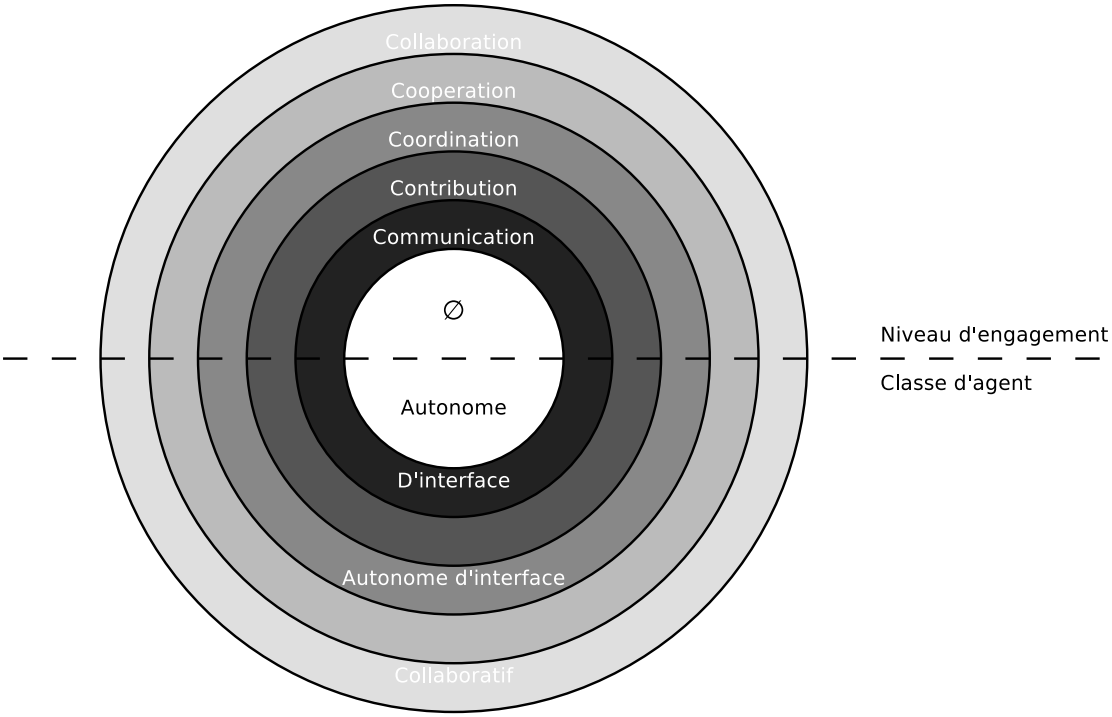

FIGURE 1.2 – Mise en correspondance de chaque classe d'agent avec les différents niveaux d'engagement du modèle « 5-C ».

1.2.5 Problèmes soulevés par la collaboration humain-machine

Si les diverses applications de la collaboration humain-machine semblent prometteuses, il est nécessaire de pointer les interrogations qui s'en suivent et d'émettre des réserves sur certains aspects. En effet, traiter des machines comme des collaborateurs à qui il est possible de déléguer des responsabilités soulève certaines questions dans différents domaines.

Acceptation sociale des machines

Les machines dotées d'intelligence artificielle peuvent être considérées comme des dangers ou des menaces par certaines personnes. En effet, il est possible de les percevoir comme des concurrents mis en compétition avec des être humains plutôt que comme des partenaires [Eps15]. C'est ce que Parasuraman et Riley [PR97] mettent en avant comme

des « applications inappropriées » qui peuvent être faites de l'automatisation de certaines tâches, où elle est appliquée sans tenir compte des conséquences qu'elle peut avoir sur les humains impliqués dans leur réalisation. Dans ces situations, les opérateurs humains sont relégués au second plan face à l'agent, dévaluant de ce fait leur rôle dans la complétion de la tâche.

Gestion des interruptions de l'utilisateur

Un problème délicat est la gestion des interruptions de l'utilisateur par un agent collaboratif. En effet, des interruptions trop fréquentes risquent d'entraver le travail de l'utilisateur, de le ralentir et d'altérer sa sensibilité à la situation. De plus, ces perturbations risquent d'intervenir au moment où il est focalisé sur un point particulier de la tâche et pendant lequel il ne souhaite pas être distrait. Endsley indique que les facteurs limitants de la sensibilité à la situation sont l'attention et la mémoire de travail [End95b]. Si cela peut sembler marginal dans de nombreuses applications, la bonne gestion des interruptions de l'utilisateur est critique dans des environnements où il est nécessaire de superviser en temps réel de nombreuses informations. C'est par exemple le cas d'un poste de pilotage d'avion de ligne, où le pilote doit gérer de nombreux facteurs dans un environnement dynamique, avec des sollicitations issues à la fois de systèmes embarqués, de l'équipage et des communication avec le trafic aérien [ML02]. L'interruption d'un pilote lors de tâches critiques a été identifié comme la cause principale d'accidents aériens [ATP95].

Les agents collaboratifs doivent donc être capables de gérer le moment auquel ils sollicitent un utilisateur, plutôt que de le faire dès que possible. Cela pose un nouveau problème de recherche : la prise de décision par un agent de communiquer avec un utilisateur. Cette décision doit tenir compte du coût de la communication et surtout de l'effet qu'elle pourrait avoir sur la tâche réalisée par l'utilisateur [UYS17].

Perte d'expertise et de contrôle

Maes [Mae94] avance que les agents collaboratifs sont perçus comme *compétents* au sens où ils sont de plus en plus utiles et concentrent de la connaissance. L'utilisateur a donc plus tendance à *lui faire confiance*, à mesure qu'il est capable de se représenter les compétences et limitations de l'agent. Si les humains peuvent se libérer de certaines tâches et de la charge mentale qui y est associée en les déléguant à des agents, cela n'est pas sans conséquence sur l'humain lui-même. En effet, il est possible que cela entraîne une perte d'expertise ou de confiance en soi de la part de l'humain. De plus, cette délégation peut s'accompagner d'un laisser-aller et d'une trop grande confiance en la machine [PR97].

Un risque supplémentaire est la perte d'adaptivité de l'humain dans l'environnement partagé avec les machines. La délégation d'une tâche à un agent s'accompagne en effet d'une diminution de la sensibilité à la situation par l'utilisateur [End17]. Celui-ci est alors « hors de la boucle de décision » et a moins d'informations à sa disposition pour anticiper et s'adapter à une évolution inattendue de l'environnement [Hoc01]. La trop forte confiance accordée au pilote automatique dans des conditions difficiles où le pilote avait peu confiance en ses aptitudes, ainsi qu'une mauvaise compréhension par l'équipage

des informations venant des systèmes automatiques embarqués sont aussi à l'origine d'accidents aériens [PR97].

Il devient ainsi souhaitable de pouvoir demander à un agent ce qui motive ses comportements et la représentation qu'il a de la situation. Une étude empirique par Stubbs *et al.* [SHW07] a montré que, à mesure qu'un système prend des décisions dans une tâche, la décharge cognitive apportée peut être contrebalancée par un accroissement de la confusion des utilisateurs qui cherchent à comprendre comment le système à pris ses décisions.

Le fait, pour un agent, d'être capable d'expliquer et de justifier ses décisions et ses actions est ce qui s'appelle *l'organisation explicable*[5], d'abord énoncé par Swartout et Moore [SM93] et récemment modernisé par Langley *et al.* [LMMC17]. Elle exige de la part d'un agent de :

— produire un registre des décisions prises lors de l'élaboration et l'exécution des plans pour atteindre ses buts. Ce registre doit inclure les alternatives envisagées, ce qui en était attendu, et les raisons de leurs rejets ;

— produire une synthèse, à différents niveaux d'abstraction et en des termes accessibles par un humain, de ses activités mentales et physiques ;

— expliquer comment le déroulement des événements a divergé du plan initial, quelles ont été les adaptations en réaction, et faire un examen rétrospectif de ces décisions par rapport aux autre possibilités ;

— communiquer ses décisions de manière compréhensible par un humain.

L'organisation explicable est nécessaire pour permettre à l'utilisateur de ne pas perdre le contrôle des agents avec lesquels il collabore. Il reste cependant à développer des mesures pour évaluer sa qualité. Langley *et al.* [LMMC17] proposent un critère subjectif où la clarté et la pertinence des réponses de l'agent aux questions de l'utilisateur seraient évaluées, ainsi que le degré de confiance qu'il suscite. Un critère plus objectif serait de quantifier la capacité d'un utilisateur, après l'interaction, à prédire le comportement de l'agent.

Dans cette section, nous avons vu certaines formes que peut prendre la collaboration humain-machine et quelles caractéristiques un agent doit exposer pour être considéré comme collaboratif. Nous avons notamment mis en avant les lacunes des agents autonomes d'interface et les aspects à améliorer pour enrichir l'interaction avec l'utilisateur pour se rapprocher d'un comportement collaboratif. La *sensibilité à la situation*, à la fois pour l'utilisateur et l'agent, a été présentée comme un élément clé de d'une collaboration réussie. Celle-ci n'est possible que si chacun est capable de percevoir son environnement, de comprendre la situation actuelle et de se projeter dans des états futurs. Pour répondre à ces attentes, il est nécessaire d'élaborer des stratégies spécifiques aux agents collaboratifs.

1.3 Stratégies pour un agent collaboratif

Cette section a pour but d'expliquer les mécanismes mis en œuvre pour doter un agent de capacités suffisantes pour réaliser collaborativement une tâche avec un utilisateur humain.

5. « Explainable agency » en anglais.

Dans la section 1.3.1, nous traitons des approches par plan, qui tentent de situer les actions de l'utilisateur dans un plan d'action permettant de réaliser la tâche. Dans la section 1.3.2, nous présentons les approches cognitives, qui tiennent compte, dès la modélisation de l'agent, des processus cognitifs impliqués dans la réalisation d'une tâche.

1.3.1 Approches par plan

La planification est au cœur de nombreux modèles pour assister un utilisateur dans la réalisation d'une tâche. Un plan est une suite d'actions établie à priori permettant, étant donné un état initial du monde, d'atteindre un état souhaité en appliquant cette suite d'actions. Le TLFi [6] définit un plan de la manière suivante [TLFa] :

Définition 1.5 – Plan : « Projet élaboré, comportant une suite ordonnée d'opérations, en vue de réaliser une action ou une série d'actions. »

Deux approches se distinguent dans l'utilisation de plan pour l'assistance à la réalisation d'une tâche : la *reconnaissance de plan*, une méthode implicite visant à reconnaître les plans de l'utilisateur en contextualisant ses actions dans des plans pré-existants, et la *planification collaborative*, qui inclut l'élaboration d'un plan partagé dans le processus collaboratif.

Reconnaissance de plan

Une approche pour la réalisation de tâches collaboratives se fonde sur la reconnaissance de plan, un procédé permettant d'inférer des intentions à partir d'actions [KA86]. Celle-ci est définie de la manière suivante :

Définition 1.6 – Reconnaissance de plan : « D'une certaine manière, la *reconnaissance de plan* est l'inverse de la *planification*. Lors de la planification, un agent essaie de formuler une séquence d'actions qui permettront d'atteindre un but ; lors de la reconnaissance de plan, un agent essaie de reconstruire, à partir des indices disponibles, un plan précédemment construit par un autre agent. Les indices incluent à la fois les actions observées de l'agent (y compris ses énoncés) et les perspectives par rapport à ses buts probables. » ([Car90] p.18)

L'objectif de la reconnaissance de plan est d'anticiper sur les actions et les besoins des autres participants. De cette manière, il est possible de devancer les attentes de son interlocuteur en ayant reconnu ses buts et en prenant l'initiative d'y contribuer. La reconnaissance de plan évite à l'utilisateur d'avoir à systématiquement annoncer ses buts avant de réaliser une action contribuant à leur réalisation.

La reconnaissance de plan part du principe qu'en plus de l'élaboration et de la mise en exécution de plans individuels, le travail collaboratif nécessite de reconnaître les plans des autres participants. Il requiert d'être capable de contextualiser leurs actions dans des plans tacites entre les interlocuteurs. Du point de vue d'un agent, cela complique la tâche

6. Trésor de la Langue Française informatisé, dictionnaire en ligne de la langue française.

de planification car il doit tenir compte, lors de l'élaboration de ses plans et de leur mise en exécution, des plans de ses interlocuteurs, de leurs capacités et de leurs actions, en plus de la dynamique de l'environnement partagé [Suc07].

Cependant, cette approche suppose que les différents agents ont à leur disposition un même ensemble de plans, qui sont complets et corrects. En effet, la reconnaissance d'un plan suppose préalablement que celui-ci existe avant d'y rattacher des actions réalisées par les participants à l'interaction.

De plus, la reconnaissance de plan part du principe que ses interlocuteurs agissent comme des agents rationnels, s'organisent de manière méthodique et prennent des décisions logiques suite à un raisonnement déductif rigoureux. Cette hypothèse s'applique dans des contextes où les différents agents de l'environnement sont des entités logicielles, comme les systèmes multi-agents [Tam97]. Cependant, son adéquation est beaucoup plus discutable lorsqu'un participant à l'interaction est un être humain. En effet, le comportement d'un être humain ne peut pas être décrit de manière purement logique du fait que celui-ci ne raisonne pas toujours de manière rationnelle [HW83]. Un être humain peut donc avoir un comportement non-uniforme, difficile à prévoir, lent, sujet à erreur et parfois irrationnel [Eps15].

Pour terminer, la reconnaissance de plan est basée sur l'hypothèse que les participants ne possèdent que des plans privés sur lesquels ils ne communiquent pas et que leur seul moyen de connaître les plans de leurs partenaires est de les inférer à partir de leurs actions. Cette hypothèse est en contradiction complète avec les caractéristiques de la collaboration, qui requiert l'établissement de buts communs entre les parties. La reconnaissance de plan n'est donc pas pleinement collaborative car elle ne suppose pas une interaction entre les participants sur les buts à accomplir et la manière de les accomplir.

Planification collaborative

La différence fondamentale entre la reconnaissance de plan et la planification collaborative est le fait que celle-ci inclut la collaboration dans le processus de planification lui-même. Il n'y a donc pas de plan établi à priori et appartenant à un seul agent. Le principe est de communiquer sur les actions à réaliser pour atteindre un but commun, plutôt que d'avoir à inférer un plan pré-établi. De ce fait, il est impossible de n'avoir qu'un échange simple entre les interlocuteurs et l'établissement collaboratif d'un plan nécessite une communication plus longue entre les interlocuteurs. De plus, cela implique que la capacité collaborative d'un agent doit être prise en compte dès sa conception [GK96].

Ce principe a été appliqué dans la conception de COLLAGEN[7], un modèle d'agent collaboratif développé par Rich *et al.* [RSL01, RS06]. L'approche choisie pour cet agent s'inspire de l'interaction humain-humain et des théories développées pour modéliser la réalisation collaborative d'une tâche par deux humains. COLLAGEN part du principe que l'amélioration de l'interaction humain-machine passe par l'étude de l'interaction humain-humain. Pour ce faire, il repose sur la *théorie de la conversation collaborative*, basée sur l'étude de la collaboration humain-humain. Une présentation plus détaillée de cette théorie

7. Pour « COLLaborative AGENt ».

sera faite en section 2.2.3.

Le besoin de gérer une planification collaborative a mené au développement de SharedPlans [GS90], le module COLLAGEN gérant les tâches et sous-tâches à réaliser par les participants à l'interaction. Plus précisément, SharedPlans permet la gestion de croyances et intentions mutuelles des participants. Pour la réalisation des tâches à accomplir, COLLAGEN fait appel à des *recettes*, qui sont des séquences d'actions précises pré-compilées représentant la connaissance de l'agent sur la manière d'atteindre un but donné [Pol90].

Les travaux sur SharedPlans ont par la suite été raffinés par Grosz et Kraus [GK96], avec notamment l'ajout de *plans partiels* et d'opérateurs d'engagement et d'intentionnalité sur la réalisation des actions d'un plan, et la différenciation entre les *plans partagés*, élaborés collaborativement entre les différents interlocuteurs, et les *plans individuels*, propres à chacun et établis à partir des plans partagés et des recettes connues par les agents.

Pour situer les actions de l'utilisateur dans le plan partagé tout en conservant un système tractable dont la complexité permet une exécution en temps réel, différentes stratégies ont été mises en place [LRS99]. La première est l'utilisation d'un *centre d'intérêt*, qui est une restriction du contexte de travail à un élément particulier de l'environnement, restreignant l'interprétation des actions de l'utilisateur. De plus, la structure de données retenue pour représenter les plans est hiérarchique, de manière à réduire les possibilités de contextualisation d'une action. En cas d'ambiguïté, il est aussi possible pour l'agent de demander des clarifications à l'utilisateur. Ces stratégies permettent de réduire la quantité de communication nécessaire au maintien d'une compréhension mutuelle entre l'utilisateur et l'agent.

Il n'est cependant pas envisageable de se contenter d'un plan élaboré collectivement à priori et exécuté sans être ré-évalué au cours de sa réalisation. En effet, au vu du caractère dynamique de l'environnement et des actions réalisées par les autres participants, il est peu probable que le premier plan établi se déroule sans complications. La planification collaborative est donc vue comme un processus itératif de raffinement de plan. De ce fait Grosz et Kraus introduisent le concept de *plans partiels* et de *recettes partielles*, qui sont des plans et recettes amenés à être révisés ou complétés au cours de la réalisation de la tâche.

L'établissement collaboratif de plan a aussi été développé dans le projet TRAINS [ASF+95]. L'objectif de TRAINS est de permettre la planification de résolution de tâche en interagissant avec l'utilisateur en langue naturelle. Une application de ce projet est la planification de trajets ferroviaires. De manière à contourner la complexité computationnelle de la reconnaissance de plans, TRAINS lie la reconnaissance d'énoncés en langue naturelle au processus de planification. Pour l'interprétation d'un énoncé ainsi que la génération d'une réponse, le système fait appel à des arbres de décision [AFM+00]. TRAINS a par la suite été utilisé pour la construction de TRIPS, un système de planification pour la gestion de situation de crise [FA98]. Celui-ci enrichit TRAINS en y ajoutant la possibilité par le système d'afficher des éléments visuels comme des cartes.

Une autre approche a été développée afin d'inférer un plan à partir d'un dialogue entre des humains portant sur la définition d'un plan pour réagir à une situation de crise [KCS15].

Elle est basée sur un système hybride réalisant d'abord une modélisation du plan à partir d'une étude statistique du dialogue puis une validation logique de ce plan. Cette approche permet de réaliser une inférence de plan avec une bonne précision dans un grand espace de solutions en ayant des données en bruitées et en faible quantité. Le principal défaut de cette approche est le fait qu'il n'est possible de la mettre en œuvre qu'une fois le dialogue de planification terminé.

Dans les situations où la planification par une machine prends beaucoup de temps et n'apporte pas de solutions optimales, la collaboration avec un être humain sur l'élaboration de plan peut être bénéfique [KBS17]. La méthode consiste à fournir à l'humain une représentation graphique du problème à résoudre et lui faire formuler des stratégies de résolution de haut niveau qui seront encodées comme des contraintes dans le problème passé au solveur. Cette méthode a montré des améliorations significatives des performances d'optimisation à temps de calcul égal pour des stratégies établies par des non-experts du domaine d'application.

La planification collaborative corrige certains défauts présentés par la reconnaissance de plan. Cependant les approches par plan, qu'elle soient collaboratives ou non, sont limitées sur plusieurs aspects pour modéliser une interaction entre un agent collaboratif et un humain. D'un côté, la réalisation de la tâche est souvent considérée comme planifiable à priori et, même si des modèles comme SharedPlans permettent de se baser sur des plans partiels qui seront raffinés au cours de sa mise en œuvre, il n'est pas possible de contextualiser des événements inattendus déclenchés de manière opportuniste, même si ceux-ci sont localement pertinents. D'un autre côté, la représentation de l'utilisateur se focalise principalement sur ses plans et les actions qu'il doit réaliser. De nombreux aspects sont laissés de côté comme la charge cognitive de l'utilisateur, son centre d'attention et la gestion du référentiel commun, qui sont des éléments primordiaux pour lui permettre de réaliser la tâche. Plus spécifiquement, les approches par plan ne tiennent pas compte de la *sensibilité à la situation* de l'utilisateur, ignorant la manière dont il perçoit l'environnement, la manière dont il le comprend et la projection dans des états futurs qu'il réalise.

1.3.2 Approches cognitives

Les approches cognitives ont pour but d'apporter à l'utilisateur un support personnalisé s'adaptant dynamiquement au cours du temps. Celui-ci évolue avec les fluctuations du volume d'information à traiter qui peut saturer l'opérateur ou au contraire créer un défaut d'activité [GNV06]. Elles tiennent donc compte de l'utilisation des ressources cognitives de l'utilisateur pour adapter leur interactivité.

Fischer [Fis01] résume ce problème en expliquant qu'il n'est pas suffisant de rendre l'information accessible à l'utilisateur n'importe où, à n'importe quel moment et sous n'importe quelle forme, mais qu'il est nécessaire de réduire la surcharge d'information en sélectionnant celle pertinente à la tâche courante et à l'utilisateur. Il s'agit donc de donner la *bonne* information, au *bon* moment et de la *bonne* manière. Intervenir au bon moment nécessite notamment de trouver un compromis entre les interruptions, qui sont intrusives pour l'utilisateur, et le fait de lui donner assez d'information pour ne pas lui faire perdre trop de sensibilité à la situation.

C'est dans ce but que l'ingénierie cognitive a été développée pour représenter l'interaction entre un humain et une machine comme un *système cognitif* [HW83]. Elle approche l'étude des systèmes humain-machine en termes de fonctions cognitives se servant de la connaissance du système sur lui-même et sur son environnement pour lui permettre la planification et la réalisation d'actions.

Différentes approches cognitives ont été développées pour représenter la réalisation d'une tâche. Elles permettent la décomposition de tâches complexes en essayant de tenir compte des processus cognitifs impliqués. *L'analyse hiérarchique de tâche*[8] [Ann03] est une méthode analytique utilisant la décomposition systématique d'une tâche en sous-tâche. Chaque unité d'analyse est une *opération*, spécifiée par un *but*, activée par une *entrée*, atteinte par une *action* et terminée par un *retour*. Cette méthode a pour but d'identifier les sources potentielles de défaillance, physiques ou cognitives, pour y associer des solutions. *L'analyse cognitive de tâche* [CKH06] est une extension de l'analyse hiérarchique de tâche. Elle y ajoute, dans les entrées d'une opération, la représentation de la connaissance et du raisonnement nécessaires pour la réaliser. Une certaine forme d'analyse cognitive de tâche est *l'analyse de tâche orientée but* [End11]. Celle-ci se focalise sur les besoins dynamiques en informations propres à un domaine particulier. Plus précisément, ces besoins en informations sont reliés à la sensibilité à la situation nécessaire à l'opérateur pour réaliser sa tâche. En d'autres termes, l'analyse de tâche orientée but a pour objectif de déterminer ce que l'opérateur a besoin de savoir pour réaliser chaque opération.

Le point commun de toutes ces approches est qu'elles fournissent une représentation de la capacité d'un humain à exécuter une opération. Cependant, elles ne permettent pas directement la réalisation collaborative d'une tâche et nécessitent d'être adaptées dans ce cas spécifique.

Conception coactive

C'est pour combler ce manque que Johnson *et al.* [JBF+14] ont développé la technique de la *conception coactive*[9]. Celle-ci est un processus de conception de systèmes dont les acteurs sont fortement interdépendants. La conception coactive permet cette conception indépendamment de toute architecture interne de la machine ou de tout modèle cognitif de l'humain. Elle donne un modèle abstrait d'interface entre l'humain et la machine spécifiant les critères à respecter pour soutenir la réalisation d'une tâche par des acteurs interdépendants. Son objectif est de décrire les capacités que l'humain et la machine doivent exhiber pour permettre un support mutuel pour la réalisation d'une tâche spécifique. Les auteurs soutiennent que la réalisation d'une tâche au sein de laquelle les acteurs sont interdépendants met en jeu des critères *d'observabilité*, de *prédictibilité* et de *dirigeabilité*. L'observabilité est le fait de permettre aux autres la représentation d'un modèle cohérent de son statut, de ses connaissances, de la tâche à réaliser et de l'environnement partagé. Elle nécessite l'observation et l'interprétation de signaux appropriés. La prédictibilité est le fait de rendre ses actions assez prédictibles pour permettre aux autres participants de les

8. Traduction de « Hierarchical Task Analysis. »
9. Traduction de « coactive design. »

prendre en compte et de s'appuyer dessus dans la réalisation de leurs propres actions. La dirigeabilité est le fait de pouvoir diriger le comportement d'autres participants ou de voir son propre comportement dirigé par d'autres. Elle inclut l'assignation directe de tâches et des instructions plus subtiles comme le conseil, la suggestion ou la mise à disposition d'informations pouvant modifier le comportement du destinataire, comme l'émission d'un avertissement.

La conception coactive permet de synthétiser la réalisation d'un ensemble de tâches sous la forme d'un *tableau d'analyse d'interdépendance*. Elle débute par une analyse hiérarchique de tâche. Il s'agit simplement de décomposer la ou les tâches à réaliser en sous-tâches pour atteindre un niveau de granularité approprié. Les *compétences* requises pour réaliser chaque sous-tâche lui sont ensuite associées, suivant la méthode de l'analyse cognitive de tâche ou de l'analyse de tâche orientée but. Ce champ ne se limite pas à des besoins informationnels et peut contenir de la connaissance, une expertise ou des facultés de perception, de décision ou d'action. Le niveau de compétence de chaque participant lui est associé pour chaque sous-tâche identifiée. Il est ensuite évalué à la fois en tant *qu'exécutant* de la tâche et en tant *qu'assistant* à l'aide d'une échelle à quatre niveaux, présentés dans le tableau 1.2.

	Rôle du participant	
Niveau	Exécutant	Assistant
3	Peut réaliser la tâche seul	Peut augmenter l'efficacité
2	Peut réaliser la tâche avec une fiabilité < 100 %	Peut augmenter la fiabilité
1	Peut contribuer à la tâche mais nécessite une assistance	Assistance nécessaire
0	Incapable de réaliser la tâche	Ne peut pas aider

TABLEAU 1.2 – Niveaux de compétence possibles en tant qu'exécutant et qu'assistant pour une sous-tâche.

Les différentes combinaisons de compétences permettent de déterminer les besoins d'observabilité de prédictibilité et de dirigeabilité pour chaque sous-tâche. De cette manière, il est possible de savoir quel participant a besoin de savoir quoi, de prédire quoi et de diriger qui. Cela permet d'identifier, pour chaque sous-tâche, les éventuelles dépendances mutuelles entre les participants, et comment permettre leur réalisation.

La conception coactive a été à l'origine développée pour aborder la question de l'interaction humain-robot. Elle a été utilisée lors d'une compétition de robotique en environnement virtuel de la DARPA [10] pour la gestion de catastrophe d'origine naturelle ou humaine [JBF+14]. Le système développé était constitué d'un robot humanoïde présent dans l'environnement virtuel assisté à distance par un opérateur humain. L'utilisation de la conception coactive a permis d'identifier les sous-tâches pour lesquelles le robot devait faire appel à l'opérateur, pour valider ou modifier une série d'actions planifiées. En effet, même si le robot devait agir autant que possible en autonomie, il faisait appel à l'opérateur pour les tâche sur lesquelles n'avait pas été identifié fiable à 100%, comme la manipulation

10. Defense Advanced Research Projects Agency, « Agence pour les projets de recherche avancée de défense », faisant partie du Département de la Défense des États-Unis.

d'objets dans l'espace. Lors de ces sollicitations, le robot fournissait à l'opérateur des indications visuelles rendant ses actions à venir prévisibles.

La conception coactive a permis à l'équipe l'ayant appliquée de finir première lors de cette compétition, apportant de la *flexibilité* et de la *résilience* dans la réalisation des épreuves par le robot [JBF+14]. Cependant, un défaut important de cette méthode est le fait qu'elle nécessite, en plus du découpage de la tâche en sous-taches, l'évaluation de la capacité de chaque participant à la collaboration à réaliser chaque sous-tâche. Cette méthode impose donc une ré-évaluation de la répartition des sous-tâches pour chaque groupe désirant résoudre un problème collaborativement. Cela implique donc une importante phase de préparation en amont avant la mise en œuvre de la résolution de la tâche.

Systèmes compagnons

Les approches cognitives ont très récemment fait apparaître le concept de *système compagnon* [BW16, BHSB16, BW17, BLB+17]. Ces systèmes ont pour objectif d'adapter leurs services spécifiquement à chaque utilisateur en tenant compte de leurs besoins, de leurs capacités, de leurs préférences, de leurs demandes, de leur situation et de leur état émotionnel. La conception de ces systèmes se base sur des critères neurophysiologiques et cognitifs pour évaluer la situation de l'utilisateur, ses attentes et l'impact qu'une action du système peut avoir sur lui. Cette vision est centrée sur l'utilisateur et considère que le modèle mental de celui-ci, dans une situation donnée et avec un but donné, a des idées préconçues et des attentes sur la manière dont le système doit se comporter, impactant son propre comportement dans la résolution du problème. Behnke *et al.* [BLB+17] présentent les principes cognitifs sur lesquels se basent les systèmes compagnons :

La planification humaine est dynamique – Les humains n'ont pas une représentation complète du contexte de leur problème, des actions possibles, d'un état initial et d'un état final et utilisent des heuristiques pour atteindre une solution. La planification est vue comme procédurale, hautement dynamique et dépendante de la situation ;

Les humains ont des ressources limitées – La mémoire de travail des humains ne peut contenir qu'un nombre restreint d'éléments. La planification, surtout dans le cadre d'un problème mal défini, est une activité éprouvante sollicitant beaucoup les ressources cognitives. Les humains font appel à des heuristiques pour supporter la charge induite par cette activité ;

La cognition est située – Le comportement des humains dépend du contexte dans lequel il se trouve. Le processus de résolution d'un problème dépend de la situation dans laquelle celui-ci prend place. De ce fait, il est difficile de créer des hypothèses vraies dans toutes les situations.

Les auteurs insistent sur le fait qu'il est primordial de rendre le système *observable* et *dirigeable*. L'observabilité permet à l'utilisateur de se créer une représentation de l'environnement partagé avec le système. La connaissance de l'état du système, de sa perception de l'environnement, de ses actions et de ses buts rends celui-ci prédictible pour

l'utilisateur. La dirigeabilité donne la possibilité à l'utilisateur de changer les buts et les priorités du système.

Les systèmes compagnons semblent être prometteurs de par le nombre et la diversité des outils qu'ils souhaitent utiliser pour supporter l'interaction avec l'utilisateur. Cependant, ceux-ci étant très récents, il restent à l'heure actuelle très prospectifs et il est difficile d'évaluer leur facilité de mise en œuvre. Une expérience a permis de donner des indices sur les pistes à suivre pour leur développement [FRA+17, RFW+17], mais des approfondissements sont nécessaires pour tirer des conclusions sur l'architecture que doit adopter un système compagnon.

1.4 La recherche d'information collaborative

Nous nous intéressons maintenant à un contexte particulier de tâche complexe réalisée par des êtres humains : la recherche d'information (RI). Celle-ci peut présenter des difficultés sur différents aspects et donc amener plusieurs personnes à réaliser une recherche d'information collaborative (RIC). De plus, l'usage de la collaboration pour une tâche de recherche d'information peut être avantageux car il n'est pas rare que le besoin d'information d'une personne soit partagé avec d'autres. Différents systèmes ont vu le jour pour enrichir la RIC et fournir des outils spécifiques à la collaboration sur un tâche de RI. Cela ouvre la voie à la conception d'agents collaboratifs dédiés à l'assistance d'être humains sur une tâche de RI. De ce fait, la RIC sera le contexte applicatif de nos travaux, présentés dans le chapitre 5.

La section 1.4.1 présente les caractéristiques principales du processus de RI. Nous y définissons les recherches auxquelles nous nous intéressons, qui sont *informationnelles*, *complexes* et *divergentes*. La section 1.4.2 présente le processus de RIC. Nous y définissons la recherche d'information collaborative à laquelle nous nous intéressons, qui est *synchrone*, *explicite* de *médiation* et de *localisation quelconques*. La section 1.4.3 présente différentes possibilités de mise en œuvre pour la RIC. La section 1.4.4 présente les avantages et limites de la RIC. Dans la section 1.4.5, nous questionnons la possibilité de réaliser une RIC entre un humain et un agent. Nous y mettons en avant les atouts et limites des systèmes actuels et synthétisons les caractéristiques à développer pour développer un agent collaboratif pour la RI.

1.4.1 Le processus de recherche d'information

Cette section présente les principaux aspects du processus de recherche d'information, au regard de notre travail. Pour une compréhension plus approfondie de la RI, il est conseillé de se tourner vers les travaux de Case [Cas16].

Une définition assez répandue de la recherche d'information est celle de Marchionini :

Définition 1.7 – Recherche d'information : « Ce que nous appelons *recherche d'information* est un processus dans lequel les humains s'engagent délibérément dans le but de changer l'état de leurs connaissances. [...] C'est un processus humain fondamental étroitement lié à l'apprentissage et à la résolution de problèmes. » ([Mar95] p.5-6).

Un *chercheur* est un personne qui s'engage dans un processus de recherche d'information.

Les catégories de recherche d'information

Différentes classifications caractérisent le processus de recherche d'information. Broder [Bro02] décrit trois types de requêtes dans sa taxonomie des requêtes de RI, permettant de déterminer ce qui pousse le chercheur à réaliser une RI :

— les requêtes *navigationnelles*, pour se rendre sur un site. Par exemple la requête « wikipedia » pour se rendre sur le site de l'encyclopédie en ligne Wikipedia ;

— les requêtes *informationnelles*, pour trouver de l'information supposée présente sur internet mais dont on n'a aucune connaissance préalable sur la source. Les requêtes « vendeur de vélo à Rouen », « fabriquer un lombricomposteur » et « recette à base de tofu » sont de ce type ;

— les requêtes *transactionnelles*, pour faire une activité intermédiée par internet, comme télécharger des fichiers et passer une commande. Les requêtes « télécharger iso Ubuntu », « acheter des espadrilles » sont des requêtes de ce type.

Dans une approche différente, Singer *et al.* [SDN12] définissent le concept de *recherche complexe*, qui est un processus nécessitant du temps, plusieurs requêtes, le parcours de plusieurs documents et l'extraction et la synthèse d'informations provenant de différentes sources. À l'inverse, une *recherche simple* ne nécessite qu'une seule requête et la découverte de l'information recherchée est directe et ne nécessite pas de synthèse.

Une classification proche est faite par Kerne et Smith [KS04], qui différencient les recherches *convergentes* des recherches *divergentes*. Les recherches convergentes correspondent à un besoin d'information *spécifié explicitement* pour lequel l'utilisateur trouvera une unique information et dont le critère de validation est clair. À l'inverse, les recherches divergentes correspondent à un besoin d'information pouvant avoir plusieurs formulations sous la forme de *questions ouvertes* et plusieurs solutions valides.

Avoir à réaliser une recherche *complexe* et *divergente* est à rapprocher de la définition d'un problème *confus* ou *épineux* donnée par Roberts [Rob00] et discutée en section 1.1.1. Nous nous intéressons dans la suite de ce manuscrit aux recherches *informationnelles complexes et divergentes*.

Propriétés du processus de recherche d'information

Marchionini présente le processus de RI d'un chercheur comme permettant de répondre à un *besoin d'information* clairement identifié à priori. Le problème à résoudre revient alors à satisfaire ce besoin d'information. La RI est vue comme un *processus itératif* au sein duquel se succèdent différentes étapes [Mar89, MW07, SE98] : reconnaissance du besoin d'information, établissement d'un plan de recherche, réalisation de la recherche, évaluation des résultats, éventuellement reformulation de la requête et répétition du processus jusqu'à satisfaction du besoin d'information de l'utilisateur.

Cette approche est cependant limitée pour deux raisons : le besoin d'information est défini *à priori* et est vu comme *statique* et le chercheur est vu comme itérant le processus de RI jusqu'à tomber sur un ensemble de documents répondant à ce besoin d'information. Or, la découverte de documents pertinents fait évoluer le besoin d'information du chercheur tout au long du processus de RI [OJ93], et le besoin d'information du chercheur est comblé par des documents découverts tout au long de la recherche.

Par conséquent, pour représenter le processus de RI, Bates [Bat89] propose le modèle de la *cueillette de baies*[11]. À l'instar d'un promeneur qui chercherait à récolter des baies en se déplaçant de bosquet en bosquet pour collecter des fruits, le chercheur d'information parcourt l'ensemble des documents à sa disposition et s'arrête lorsqu'il en trouve qui l'intéressent. Cette découverte peut amener son besoin d'information à évoluer, conduisant à une modification de la requête au cours du processus. La découverte de ressources pertinentes peut le mener dans un direction *imprévue*, faisant du besoin d'information du chercheur un objet *dynamique*. Cela amène le déroulement du processus de RI à dépendre des ressources trouvées au fur et à mesure de son avancement, le rendant intrinsèquement *opportuniste*. À la fin du processus, l'utilisateur aura comblé son besoin d'information à l'aide de documents trouvés tout au long de son parcours, et non pas avec un ensemble unique renvoyé par une seule requête.

À cela s'ajoute la personnalité du chercheur lui-même, qui peut être variable. En effet, Iivonen [Iiv95] a montré que les stratégies des chercheurs peuvent être incohérentes et sous-optimales, indépendamment de la réussite de leur recherche et de la qualité des résultats obtenus. Cela contredit l'idée selon laquelle une RI serait un processus indépendant du chercheur dont le déroulement serait un plan statique défini à priori à partir d'un besoin d'information inital.

Bates décrit aussi le processus de RI comme étant *stratégique* et a identifié différentes *tactiques* employées par un chercheur [Bat79]. De ces travaux, quatre types d'activités réalisées au cours d'une recherche ont été identifiés [Bat90] :

Les coups – Un coup est l'unité de base de l'activité. Il s'agit de n'importe quel type d'activité élémentaire associée à la recherche, comme ajouter un mot-clé ou un opérateur à la requête ;

Les tactiques – Une tactique est un ou plusieurs coups réalisés avec l'intention de faire avancer la recherche en tenant compte de l'état courant. Remplacer un mot de la requête par un hyperonyme ou un synonyme est une tactique ;

Les stratagèmes – Un stratagème est un ensemble de coups et/ou de tactiques, impliquant généralement un domaine de recherche particulier et un mode de traitement répétitif des informations obtenues. Par exemple, le domaine peut être un ensemble de références bibliographiques et le stratagème correspond au fait de rechercher tous les travaux d'un auteur ;

Les stratégies – Une stratégie est un *plan complet* pour une recherche complète, et peut contenir les trois types d'activités précédents. Bates précise qu'une stratégie pour

11. Traduction de « berrypicking. »

une recherche complète est difficile à établir en dehors d'une recherche simple du fait de l'aspect opportuniste de la RI.

Bates indique que ces quatre types d'activités sont conceptuellement différents et qu'ils ne sont pas réductibles les uns aux autres : une tactique ne se réduit pas à un ensemble de coups, stratagème ne se réduit pas à un ensemble de tactiques et stratégie ne se réduit pas à un ensemble de stratagèmes. Une tactique peut être un ou plusieurs coups, un stratagème peut utiliser des tactiques ou non, et ainsi de suite.

1.4.2 Caractérisation d'une recherche d'information collaborative

Il est généralement accepté que la recherche d'information est une *activité individuelle* réalisée de *manière isolée*. Ce point de vue est néanmoins remis en question avec l'émergence de la recherche d'information collaborative (RIC) entre humains comme objet d'étude [FMCB04, HJ05, Fos06]. Certains voient la RIC comme une extension du modèle standard de la RI [Her08]. En effet, la RIC peut-être perçue comme la combinaison de la RI individuelle et de l'établissement d'un référentiel commun [12]. Les participants réaliseraient individuellement des recherches puis partageraient leurs résultats avec les autres pour maintenir une représentation cohérente de l'information trouvée par le groupe.

Cependant, le modèle de la RIC est de plus en plus considéré comme un sujet de recherche à part entière et non pas comme un ajout au modèle de RI standard [HHH15, RJ08]. Cela nécessite d'étudier le processus de RIC en tenant compte à la fois de son côté collaboratif et de son côté RI, sans le réduire à une juxtaposition des deux.

La RIC étant une discipline récente, elle n'est pas définie par tous les auteurs de la même manière et n'inclut pas les mêmes concepts pour tout le monde. Nous donnons ici les définitions qui nous intéressent pour la RIC, sans chercher à présenter tous les points de vue possibles.

Golovchinsky [GPB09, GQPG09] donne une taxonomie en quatre catégories décrivant les différentes formes que peut prendre une RIC :

L'intention – elle est *implicite* ou *explicite*. La collaboration implicite repose sur le fait que les moteurs de recherche utilisent les informations trouvées par d'autres utilisateurs cherchant des informations similaires pour organiser leurs résultats. Les utilisateurs ne collaborent donc pas directement les uns avec les autres mais leurs sessions de recherche s'influencent mutuellement. À l'inverse, la collaboration explicite concerne un ensemble d'utilisateurs se regroupant pour partager leur besoin d'information et chercher des documents pertinents ;

La médiation – elle est *d'interface* ou *algorithmique* et définit à quel niveau a lieu la collaboration. La médiation d'interface correspond à tous les outils mis en place au niveau de l'interface utilisateur pour permettre à plusieurs utilisateurs de mener une session de recherche partagée. La médiation algorithmique concerne la prise en compte, dans le fonctionnement du moteur de recherche, des contributions de chaque utilisateur pris individuellement ;

12. Traduction de « grounding. »

La concomitance – elle est *synchrone* ou *asynchrone*. Une collaboration est synchrone quand les participants recherchent de l'information en même temps et peuvent s'influencer en temps réel. Elle est asynchrone quand les participants ne travaillent pas en même temps et peuvent profiter des résultats trouvés précédemment par d'autres collaborateurs ;

La localisation – elle est *co-localisée* ou *à distance*. Une collaboration est co-localisée si les participants sont physiquement au même endroit et ne communiquent pas uniquement par l'intermédiaire d'une machine. À l'inverse, une collaboration à distance donne plus d'opportunités de collaborer mais nécessite des canaux de communication alternatifs comme un clavardage ou une téléconférence.

Nous nous concentrons dans la suite de ce manuscrit sur une recherche collaborative de documents *synchrone*, *explicite* de *médiation et de localisation quelconques*. Le lecteur intéressé pourra trouver un tour d'horizon du reste des situations dans les travaux de Soulier et Tamine [ST17].

La définition de Shah est la plus proche de notre vision de la RIC :

Définition 1.8 – Recherche d'information collaborative : « Un processus intentionnel, interactif et mutuellement bénéfique de recherche d'information qui se déroule au sein d'un projet collaboratif (éventuellement une tâche complexe) dans un petit groupe de participants (potentiellement avec des compétences et rôles différents) » ([Sha14] p. 219).

Cette définition met en avant l'aspect délibéré de la RIC excluant de ce fait l'intention implicite proposée par Golovchinsky. De plus, elle met en avant un *projet collaboratif* au sein duquel les participants peuvent avoir des *compétences et rôles différents.* Les motivations pour commencer une RIC sont diverses et dépendent du chercheur et du besoin d'information [FMCB04]. Plus spécifiquement, Spencer *et al.* [SRH05] ont montré dans une étude portant sur les pratiques de RIC dans le milieu universitaire, qu'une raison principale de sollicitation d'une collaboration est le manque d'expertise du chercheur. Du point de vue des participants à l'étude, la RIC était perçue comme plus facile et apportant plus de résultats pertinents qu'une recherche solitaire.

Reddy et Jansen [RJ08] ont montré, au sein d'une équipe médicale, que des situations de collaboration se créent pour faire face à un besoin d'information complexe. Cela est d'autant plus prononcé lorsque différentes expertises de domaines sont nécessaires pour combler le besoin d'information. Hansen et Järvelin [HJ05] ont étudié le comportement d'une équipe d'ingénieurs travaillant sur des brevets et ont montré que des situations de collaboration se créent tout au long du processus de RI, de la formulation du problème jusqu'à l'exploitation des résultats obtenus. Dans ces contextes, de par la nature complexe et divergente de leurs recherches, les chercheurs font face à un problème difficile et se tournent alors vers leurs semblables pour trouver une solution. Cela montre l'application de l'idée de Roberts [Rob00] selon laquelle la collaboration est appropriée pour résoudre ce type de problèmes.

Dans son étude sur les usages de la RIC, Morris [Mor08] a constaté que la RIC sur le web est très répandue et que dans les cas où elle s'avère nécessaire pour la réalisation d'un problème, elle s'impose même lorsque des outils dédiés ne sont pas à la disposition

des collaborateurs. Dans ce cas, des outils de communication usuels sont détournés pour mener à bien la tâche collaborative [Mor08].

Dans une étude pour mieux cerner les catégories de problèmes qui font l'objet d'une résolution collaborative, Morris [MTP10] a étudié l'utilisation des réseaux sociaux pour poser des questions à ses relations. Les trois premiers types d'information recherchées étaient des recommandations (29%), des opinions (22%) et de la connaissance factuelle (17%). Les quatre premières motivations des utilisateurs étaient la confiance accordée à leurs contacts (24.8%), le besoin d'un point de vue subjectif (21.5%), l'idée que l'utilisation d'un moteur de recherche ne donnerait pas de résultats satisfaisants (15.2%) et le ciblage d'un public spécifique (14.9%). Ce qui motive la collaboration est la *qualité* des résultats et le fait que s'adresser à d'autres personnes soit perçu comme étant la *seule alternative possible* pour trouver une réponse satisfaisante.

1.4.3 Mise en œuvre de la RIC

Pour Foley *et al.* [FSL06, FS10], deux formes d'organisation de groupe cohabitent au sein du processus de RIC : la *division du travail* et le *partage des connaissances*.

Division du travail

La division du travail à pour but de partager la tâche de RIC parmi les collaborateurs de manière à diviser la charge de travail entre tous les participants. Le but de cette stratégie est d'éviter la redondance des recherches entre les participants et de réaliser la tâche plus rapidement. Cependant, sa mise en œuvre n'est pas toujours aisée de par la nature flexible et itérative du processus de RI [FS10].

La division du travail se base sur une attribution, implicite ou explicite, de *rôles* à chaque collaborateur, donnant à chacun une fonction pour la réalisation de la RIC, avec des objectifs propres. Selon les cas, elle peut amener à différencier les outils utilisés par les collaborateurs en fonction de leurs rôles. Dans sa taxonomie, Golovchinsky [GQPG09] décrit les différents rôles qui peuvent être attribués en fonction de la situation, dans le cas d'une RIC à deux :

— Le rôle de *pair* : les collaborateurs utilisant cette méthode ont tous une fonction et des responsabilités similaires, chacun devant explorer un sous-ensemble de l'espace de recherche. La mise en commun des résultats se fait directement par les utilisateurs, sans intermédiaire. Ce type de stratégie est souvent mis en place lors de l'utilisation d'outils existants ne fournissant pas directement d'outil de collaboration. Dans une enquête sur les usages de la RIC, Morris [Mor08] a observé cette méthode en la nommant *diviser pour régner* ;

— Les rôles *d'expert d'un domaine A* et *d'expert d'un domaine B*, où chaque collaborateur possède des connaissances sur un domaine différent et l'interface est la même pour chacun ;

— Les rôles *d'expert en recherche / novice en recherche* ou *d'expert du domaine/novice du domaine* reflètent une asymétrie entre les collaborateurs au niveau de leur exper-

tise de domaine ou de leur familiarité avec les outils de recherche. Cette dernière peut amener chaque collaborateur à utiliser une interface différente ;

— Les rôles *d'expert en recherche / expert du domaine* amènent une forte asymétrie entre les collaborateurs, permettant à un expert du domaine avec un besoin d'information complexe de tirer parti du potentiel des outils de recherche. Dans cette situation, l'expert en recherche est amené à utiliser une interface de recherche avancée alors que l'expert du domaine reste souvent cantonné à une interface rudimentaire ;

— Les rôles de *prospecteur/mineur* amènent les participants à réaliser des activités différentes lors de la recherche. Cette division du travail pousse le prospecteur à faire une recherche en largeur et le mineur à faire une recherche en profondeur. Le prospecteur lance de nombreuses requêtes pour parcourir l'espace de recherche, évalue de manière superficielle les documents renvoyés par le moteur de recherche. Le mineur prends le temps de faire un jugement de pertinence approfondi sur les documents n'ayant pas été examinés par le prospecteur. L'interface de chaque collaborateur peut fournir des outils spécialisés facilitant l'exécution de son rôle. C'est par exemple le cas avec l'outil de recherche de vidéo développé par Pickens *et al.* [PGS+08], où chaque utilisateur a une interface optimisée pour sa tâche.

Shah *et al.* [SPG10] proposent une autre attribution des rôles se rapprochant de la paire prospecteur/mineur. Un rôle est celui du *rassembleur*[13] et l'autre est celui de *surveilleur*[14]. Cette répartition des rôles est réalisée par le système, celui-ci proposant à chaque participant des résultats différents en fonction du rôle qui lui est attribué. Le système propose au rassembleur des documents sélectionnés pour leur précision, de manière à identifier rapidement des documents pertinents. Il propose au surveilleur des documents ayant une plus forte diversité thématique de manière à permettre une exploration plus vaste de l'espace de documents.

L'approche tenant compte des rôles d'expert/novice du domaine a été développée dans les travaux de Soulier [Sou14, STS16]. Ce modèle se sert d'une représentation explicite du profil de chaque utilisateur pour personnaliser la pertinence des documents renvoyés par le moteur de recherche.

Les outils SearchTogether [MH07] et *Coagmento* [SMK09, GS11, MLS18] ont été développés dans le but de soutenir la division du travail. Ils mettent à disposition des utilisateur des moyens de communication comme une messagerie instantanée, la liste des requêtes lancées et des documents visités par les collaborateurs, les évaluations qu'ils en ont fait et le fait qu'ils en aient marqué comme pertinent, ou la possibilité de distribuer les résultats d'une même requête aux différents participants pour une parallélisation de leur utilisation.

Partage des connaissances

Le partage des connaissances permet de communiquer à ses collaborateurs des idées et des informations au cours d'une activité de groupe. Le but de cette stratégie est de per-

13. traduction de « gatherer. »
14. traduction de « surveyor. »

mettre aux participants de bénéficier des connaissances et des expertises des autres [FS10].

Les environnements de travail partagés, physiques ou virtuels, sont un support au partage des connaissances entre collaborateurs. Dans les situations de RIC co-localisée, l'apparition d'outils de visualisation à grande échelle a facilité le partage des connaissance entre les utilisateurs y faisant appel. L'utilisation de tables tactiles pour la RIC [SLFM07, MPW06, MLW10] permet d'augmenter la sensibilité à la situation des collaborateurs et de faciliter leurs échanges en permettant d'utiliser la communication non verbale [FS10].

L'outil SearchTogether [MH07] a été développé dans le but de permettre la sensibilité à la situation et la division du travail. La connaissances des requêtes lancées par ses collaborateurs est permise grâce à une partie de l'interface où chaque requête est associée à la photo du collaborateur l'ayant lancée. La liste des résultats renvoyés par le moteur de recherche affiche le nombre de collaborateurs ayant déjà visité chaque ressource et chaque page affiche la liste des collaborateurs l'ayant déjà visitée et les annotations qu'ils peuvent avoir laissées. De plus, il est possible de recommander des pages à ses collaborateurs, pour mettre en avant une ressource particulière. Ces fonctionnalités permettent d'éviter de répéter un travail déjà réalisé.

1.4.4 Avantages et limites de la RIC

Avantages de la RIC

Nous avons précédemment montré en quoi la collaboration aide à la résolution d'un problème difficile. Cela s'applique dans le cas de la RI, où la mise en commun d'un besoin d'information entre plusieurs collaborateurs peut permettre ou faciliter la découverte de documents pertinents. Nous présentons ici différentes expériences réalisées montrant les avantages d'une RIC.

Certaines situations RI ont montré que la collaboration, en plus d'apporter différentes expertises pour combler le besoin d'information, augmente la couverture du sujet d'étude [SGI11, SHGI15] et enrichit le vocabulaire utilisé lors de la recherche [GQPG09, JHJ08]. Yue *et al.* [YHHJ14] ont observé une incidence significative des actions collaboratives sur la formulation des requêtes dans un environnement où deux personnes réalisaient une tâche de RIC synchrone à distance avec la possibilité de discuter par clavardage et de voir l'historique des recherches de son partenaire. Ils ont notamment remarqué que les termes utilisés au cours d'une communication par clavardage lors de la recherche avaient une forte influence sur la formulation de la requête. De plus, les collaborateurs étaient enclins à utiliser des termes présents dans les requêtes lancées par leurs partenaires, élargissant leur propre champ d'action.

Pickens *et al.* [PGS+08] ont montré que la collaboration au cours d'une tâche de recherche de vidéo en partageant le travail selon les rôles de prospecteur/mineur augmente l'efficacité des chercheurs et permet de trouver des ressources ne pouvant pas être trouvées lors d'une recherche individuelle.

Limites de la RIC

Comme nous l'avons expliqué en section 1.1.2, la collaboration a aussi certains inconvénients. Nous présentons dans cette section des études ayant observé des situations où la RIC peut être désavantageuse par rapport à une RI individuelle.

Dans une étude expérimentale, Joho [JHJ08] a montré que dans un certain contexte de recherche collaborative d'information, le coût de la collaboration n'est pas compensé par ses bénéfices. Dans ce cas, la collaboration n'apporte pas d'amélioration significative à la recherche et ne permet donc pas de résoudre le problème plus efficacement.

Le processus de RIC apporte des contraintes supplémentaires pour les participants. Dans une situation réelle de RIC, les collaborateurs doivent passer du temps à communiquer, peuvent avoir d'autres priorités que de contribuer à la RIC et l'aspect opportuniste du processus de RI peut les faire dériver de leur but initial, sans pour autant répondre au besoin d'information partagé.

González-Ibáñez *et al.* [GSW12] ont observé que la communication permettant aux collaborateurs de s'organiser est limitée par leur connaissance. En effet, des chercheurs travaillant sur un sujet qui ne leur est pas familier peuvent seulement communiquer sur ce qu'ils connaissent, conditionnant leur comportement et les stratégies mises en place par l'information qu'ils possèdent à priori.

Kelly et Payne [KP13] avancent que la division du travail et notamment l'exploitation des documents pose deux problèmes. Premièrement, elle suppose que le tri par les collaborateurs entre les documents pertinents et non pertinents est réalisé sans erreur. Cependant, dans une situation de RI, la méconnaissance du domaine par un utilisateur peut lui faire faire des erreurs d'étiquetage. Deuxièmement, il peut être intéressant pour un utilisateur de savoir ce qui n'est pas pertinent pour mieux cerner son besoin d'information. De ce fait, la compréhension du domaine dans lequel il est en train de réaliser sa recherche pourrait être entravée par la suppression de l'espace de recherche des documents jugés comme non pertinents par son partenaire.

Une des difficultés dans la conception des systèmes de RIC est de faire en sorte que les outils montrant les activités des collaborateurs offrent une bonne sensibilité à la situation, sans pour autant être trop invasif dans le travail de chacun. La gestion simultanée de la recherche d'information et des échanges avec les autres membres du groupe peut augmenter la charge cognitive et être difficile à gérer pour les collaborateurs [APC+07, FS09, FMCB04]. De plus, les outils collaboratifs sont souvent trop rigides pour les utilisateurs qui s'engagent dans une tâche de RIC de manière opportuniste, informelle et non planifiée [GGB+08].

1.4.5 Vers un modèle d'agent collaboratif pour la recherche d'information

Nous essayons maintenant de rapprocher les différents outils de RIC des classes d'agents présentés en section 1.2.

Les systèmes de RIC ayant une *médiation d'interface* peuvent être perçus comme des *agents d'interface* spécialisés pour supporter une tâche collaborative humain-humain. Leur rôle est de simplifier la collaboration en enrichissant l'interface des moteurs de

recherche d'outils permettant aux collaborateurs d'avoir une meilleure conscience des actions réalisées par leurs partenaires et de pouvoir communiquer avec eux. C'est le cas des systèmes SearchTogether [MH07], CoSense [PM09], Coagmento [GS11]. Ceux-ci complètent un moteur de recherche avec des outils comme un clavardage, la liste des requêtes lancées par les collaborateurs, les mots-clés les plus utilisés, les pages visitées, les pages recommandées par les collaborateurs, les évaluations et annotations réalisées sur les pages. Ils peuvent aussi contenir un espace de travail permettant aux collaborateurs de réaliser une agrégation et une synthèse de leur recherche, et d'organiser la division du travail et les tâches à réaliser.

La médiation d'interface est cependant perçue comme un outil *passif* n'apportant pas aux collaborateurs un support personnalisé et leur laissant toute la responsabilité du déroulement de la RIC. Certains systèmes de RIC ne limitent pas leur action à l'interface et réalisent une *médiation algorithmique* de la collaboration [PGS+08]. Celle-ci personnalise activement au cours de la collaboration les résultats renvoyés par le moteur de recherche du système en tirant parti des profils des utilisateurs ainsi que du déroulement de la recherche et des documents trouvés par les chercheurs. Les systèmes réalisant une médiation algorithmique peuvent être rapprochés des *agents autonomes d'interface* car, en plus d'aider les collaborateurs à réaliser leur tâche, ils tentent de faire activement avancer la recherche en réalisant en privé des actions pertinentes dans le contexte dans lequel ils sont exécutés. Dans cette situation, les collaborateurs peuvent faire avancer la RIC à leur rythme tout en ayant leur travail influencé en temps réel par les activités de leurs partenaires. C'est par exemple le cas du système développé par Pickens *et al.* [PGS+08], qui permet une collaboration intentionnelle et synchrone sur une tâche de recherche de vidéo. Le système se base sur des rôles de prospecteur/mineur, et propose une interface avec des outils développés pour chaque rôle. Mais la personnalisation va plus loin et chaque rôle a un algorithme développé spécifiquement pour lui. Ceux-ci tiennent compte en temps réel des documents visités ou non et évalués comme pertinents ou non par chaque participant pour sélectionner et ordonner les documents qui vont être proposés à un participant ayant lancé une requête. Plus spécifiquement, le comportement du prospecteur influence la manière dont les résultats sont ordonnés pour le mineur et le comportement du mineur influence la liste des termes suggérés au prospecteur pour formuler sa requête. Querium [GDD12] est un autre système faisant appel à la médiation algorithmique sur des rôles prospecteur/mineur.

Shah *et al.* [SPG10] proposent aussi une médiation algorithmique pour une collaboration sur des rôles asymétriques de rassembleur/surveilleur. Cette médiation se base sur la fusion et re-division des résultats des requêtes soumises par les collaborateurs. Ce partitionnement des résultats est réalisé de manière à mettre l'accent sur le rôle de chacun, lui proposant une liste personnalisée. La liste du rassembleur est optimisée pour augmenter *l'efficacité* de celui-ci en se focalisant la précision *la précision* des documents qui lui sont proposés. La liste du surveilleur est optimisée pour permettre à celui-ci *d'explorer* l'espace de documents en se focalisant sur *la diversité* des documents proposés.

Une autre approche de la médiation algorithmique pour la RIC humain-humain est l'appariement opportuniste de chercheurs isolées utilisant le même moteur de recherche et

réalisant une recherche sur le même sujet. En ce sens, González-Ibáñez *et al.* [GSW14] ont développé les « collabportunities », proposant de manière opportuniste à deux utilisateurs de collaborer de manière explicite pour trouver ensemble des réponses à leur besoin d'information. Les auteurs appellent cette situation une *pseudo-collaboration*. L'idée de ce système est de solliciter les utilisateurs à réaliser une pseudo-collaboration uniquement dans les situations où celle-ci leur serait bénéfique. La difficulté réside dans l'évaluation à priori des avantages que pourrait apporter une collaboration entre deux chercheurs par rapport à une recherche solitaire. Toutefois, les auteurs proposent une méthode capable d'indiquer à quel moment une collaboration entre deux utilisateurs serait bénéfique.

Les systèmes présentés précédemment sont principalement vus comme des intermédiaires entre des humains réalisant une RIC. Même si les plus avancés peuvent influencer le cours de la recherche en personnalisant les résultats renvoyés à chaque collaborateur, les systèmes actuels ne sont pas vus comme *prenant part activement* au processus de RI sous-jacent à la RIC. Ils restent considérés comme des outils venant en support de la RIC humain-humain, mais ne sont en aucun cas envisagés comme des collaborateurs lors de la RIC.

Marchionini [MW07] met en avant le concept *d'assistant de recheche* pour la réalisation d'une tâche de RI. Ceux-ci seraient des compagnons qui assisteraient les utilisateurs lors de leurs RI, développant leur connaissance de l'utilisateur et de son besoin d'information à mesure que celui-ci réalise sa recherche et interagit avec l'agent. C'est par exemple le cas avec Letizia [Lie95] présentée précédemment. Ces agents viennent en supplément d'une interface de recherche classique et observent le comportement de l'utilisateur, éventuellement de manière multimodale, pour essayer de capter ses intentions. Les agents assistants cherchent ensuite activement des ressources à proposer à l'utilisateur et des modifications à apporter à sa requête.

Ces agents sont ce qui se rapproche le plus des agents collaboratifs, cependant il sont encore lacunaires sur certains points. De notre point de vue, la conception d'un agent collaboratif pour la RI doit reprendre l'idée d'un agent capable d'adapter algorithmiquement en temps réel les propositions qu'il fait aux humains avec lesquels il collabore en tenant compte de leurs actions tout au long du processus de RI. Cependant, celui-ci doit aussi avoir des capacités d'interaction lui permettant de communiquer de manière naturelle avec l'utilisateur. Pour rendre cet agent pleinement collaboratif, celui-ci doit de plus avoir une représentation de la tâche que l'utilisateur est en train de réaliser et pouvoir situer ses actions dans l'avancée de celle-ci.

1.5 Synthèse

Dans ce chapitre, nous avons présenté notre vision de la collaboration et de la manière dont il est envisageable de l'établir entre un humain et une machine.

Nous avons d'abord caractérisé la collaboration et les notions qui s'y rattachent, notamment le modèle en « 5-C ». Ensuite, nous avons présenté les travaux ouvrant la voie à une collaboration entre un humain et un système informatique appelé *agent*. Par la suite, nous avons identifié les stratégies que peut mettre en œuvre un *agent collaboratif*

pour assister un utilisateur dans la réalisation d'une tâche. Enfin, nous avons présenté le processus de *recherche d'information* et son application dans le contexte collaboratif.

Nous avons montré que la collaboration humain-machine est une perspective prometteuse pour la réalisation de tâches difficiles à réaliser par des humains seuls. Nous avons présenté des concepts généraux donnant des lignes directrices pour la conception et l'évaluation de systèmes collaboratifs.

Cependant, il reste à montrer de quelle manière une collaboration humain-machine peut être mise en œuvre. Plus particulièrement, il est nécessaire d'expliciter l'articulation entre l'interaction humain-machine et la réalisation de la tâche sous-jacente à cette interaction.

Chapitre 2

Modèles formels des interactions dialogiques

« On a tous les même réponses aux même questions
Puisqu'on matte tous les même émissions sur les même chaînes.
On fait tous le même discours sur le même ton,
Et les héros du feuilleton, eux, rejouent toujours la même scène. »

Hocus Pocus, « Géométrie », *73 Touches*

Sommaire

Comme nous l'avons montré dans le précédent chapitre, la communication est un support central pour la collaboration. Dans ce chapitre, nous présentons certains modèles existant dédiés au dialogue humain-machine. Notre vision est que, pour rendre un agent collaboratif, il faut le doter d'une capacité d'interaction structurant à la fois l'interaction elle-même et la tâche sous-jacente à cette interaction.

Dans la section 2.1, nous présentons les fondements sur le dialogue. Nous distinguons ensuite deux approches principales : les approches *intentionnelles* et les approches *conventionnelles et sociales*. Les approches intentionnelles, présentées en section 2.2, présentent le dialogue comme une manifestation de plans établis par les interlocuteurs. Celles-ci sont intéressantes car elles permettent de modéliser le support collaboratif qu'un agent doit fournir [Ort14]. Les approches conventionnelles et sociales, présentées en section 2.3, expliquent la structure du dialogue par un ensemble de motifs d'interaction normatifs. Nous terminons ce chapitre en section 2.4 en présentant le modèle DOGMA et défendant l'idée selon laquelle les jeux de dialogue sont des approches mixtes permettant de créer un agent à la fois réactif et délibératif.

2.1 Fondements sur le dialogue

Avant de nous intéresser à l'interaction humain-machine, nous présentons dans cette section ce qui caractérise le dialogue comme type d'interaction entre deux être humains à propos d'une tâche. La section 2.1.1 fait une caractérisation générale du dialogue. La section 2.1.2 présente la théorie de Clark [Cla96] selon laquelle le dialogue est un *projet conjoint opportuniste*. La section 2.1.3 présente la théorie des *actes de langage* d'Austin [Aus62], qui présente le langage comme un moyen de changer l'état du monde. La section 2.1.4 présente la théorie des *actes de dialogue*, une extension des actes de langage interprétés dans le contexte du dialogue.

2.1.1 Caractérisation du dialogue

Brennan donne la définition suivante pour le dialogue :

Définition 2.1 – Dialogue : « La conversation est une activité conjointe au sein de laquelle deux participants ou plus utilisent des formes linguistiques et des signaux non-verbaux pour communiquer de manière interactive. Les dialogues sont des conversations entre deux participants. » ([Bre13] p.22)

Comme l'indique Pasquier [Pas05], ce qui différencie le dialogue du discours est la recherche d'une *inter-compréhension*, contraignant les interlocuteurs à s'assurer qu'ils se comprennent pour co-construire des interprétations communes. Cette co-construction ne signifie cependant pas que les interlocuteurs doivent être d'accord, ceux-ci pouvant avoir une interprétation commune de leur désaccord.

Pasquier liste les facteurs pouvant faciliter l'inter-compréhension des interlocuteurs :

— la recherche d'un langage commun ;

— la détection des ambiguïtés et des incohérences dans le discours d'autrui ;

— le production et la recherche de retours de compréhension (feedback), en fournissant et cherchant chez l'autre des indices de compréhension ;

— les échanges correctifs, permettant de corriger sa compréhension ou celle d'autrui en cas d'erreur d'interprétation ;

— la capacité à méta-communiquer, c'est-à-dire à dialoguer sur le dialogue en cours.

De ces caractéristiques découle une propriété plus générale du dialogue : *l'interactivité*, qui est l'influence mutuelle des actions des participants sur celles des autres. De ce fait, la structure du dialogue est *imprévisible*, celle-ci étant le fruit d'un processus de co-construction développé tout au long du dialogue.

Jucker [Juc92] distingue deux types de conversations : les conversations orientées *structure* et les conversations orientées *processus*. Les conversations orientées structure sont des conversations contraintes, orientées et ayant lieu au sein d'une activité précise, en lien avec une tâche particulière ou devant mener à certains résultats. À l'inverse, une conversation orientée processus est une conversation libre, n'ayant pas de but général. Une comparaison de leurs caractéristiques est donnée dans le tableau 2.1. De manière à mieux cerner le contexte d'un dialogue orienté structure, nous retenons la définition du TLFi d'une tâche [TLFb] :

Définition 2.2 – Tâche : « Travail défini et limité, imposé par autrui ou par soi-même, à exécuter dans certaines conditions. »

Caractéristiques	**Orienté processus**	**Orienté structure**
Contraintes sur le type de contribution	Peu	Beaucoup
Buts	Multiples et locaux	Peu nombreux et globaux
Ordre de prise de parole	Plutôt libre	Plutôt fixe
Rôle des participants	Pas très bien défini	Bien défini
Contribution dépendante des contributions passées	Souvent	Rarement
Principe d'organisation	Local	Global

TABLEAU 2.1 – Comparaison des types de conversation orienté processus et orienté structure [Juc92]. Repris et enrichis par Vongkasem et Chaib-draa [VCd00].

Grice [Gri75] a défini le *principe de coopération*, permettant aux interlocuteurs d'échanger de l'information lors d'un dialogue. Celui-ci est défini de la manière suivante :

Définition 2.3 – Principe de coopération : « Rendre votre contribution conversationnelle telle que requise, au moment atteint par celle-ci, par le but ou la direction acceptée de la discussion dans laquelle vous êtes engagé. » ([Gri75] p.45)

Ces travaux ont mené à l'élaboration des « maximes de Grice » :

Maxime de quantité – rendre sa contribution aussi informative que nécessaire et ne pas rendre sa contribution plus informative que nécessaire ;

Maxime de qualité – ne pas dire ce que l'on pense être faux et ne pas dire ce pour quoi l'on manque de preuve ;

Maxime de relation – être pertinent ;

Maxime de manière – s'exprimer clairement, de manière ordonnée, concise et non ambiguë.

Le respect de ces maximes aide l'interlocuteur d'interpréter correctement les énoncés du locuteur.

Lors d'un dialogue, les participants peuvent ne pas respecter ces règles et en *violer* une ou plusieurs. Ce phénomène mène à une *implicature*, qui est le processus d'interprétation d'un tel énoncé.

2.1.2 Le dialogue comme projet conjoint opportuniste

Clark [Cla96] présente l'usage social du langage comme un *type d'activité collective* ayant lieu au sein d'un *projet conjoint*. Un projet conjoint est une action commune proposée par un participant et acceptée par tous menant à l'accomplissement de buts collectifs, en plus d'éventuels buts individuels. Ces buts collectifs sont réalisés par des actions dont la plupart sont des *actions conjointes*. Une action conjointe n'est pas simplement la somme des actions individuelles des participants. En effet, elle implique des *actions autonomes* et des *actions participatives*. Les actions autonomes sont réalisées par un participant sans tenter de se coordonner avec d'autres participants. À l'inverse, une action participative implique une coordination entre les participants sur le processus et le contenu de l'action. Clark fait le parallèle entre la réalisation d'un dialogue comme une action conjointe et la réalisation d'activités comme jouer un duo en musique, manœuvrer un canoë ou se serrer la main.

Clark précise que le dialogue en tant qu'activité conjointe a les caractéristiques suivantes :

Participants – il implique au moins deux participants ;

Rôles – les participants ont un rôle public au sein de cette activité déterminant leur répartition des tâches ;

Buts publics – les participants cherchent à établir et réaliser des buts collectifs ;

Buts privés – les participants peuvent chercher à réaliser des buts privés ;

Hiérarchies – il est composé d'une hiérarchie d'actions ou d'activités conjointes ;

Procédures – les participants sont susceptibles d'utiliser des procédures conventionnelles et non-conventionnelles pour atteindre leur but ;

Dynamique – les activités conjointes peuvent être simultanées ou séquencées ;

Limites – les participants s'accordent sur son début et sa fin.

Pour Clark, il est possible de coordonner une action conjointe car on peut lui associer une *entrée*, un *corps* correspondant à l'action elle-même, et une *sortie*. Cette représentation permet aux participants d'aller de l'état où ils ne sont pas engagés sur l'action conjointe à l'état où ils sont engagés sur l'action conjointe, puis d'en ressortir. Pour résoudre le

problème de coordination qui se pose au cours du dialogue, il est nécessaire de faire appel à un *outil de coordination*[1], permettant de connaître les actions que chaque participant est sensé réaliser à un instant donné du dialogue. Cet outil peut être une convention, une habitude, un accord réalisé explicitement, ou n'importe quelle autre structure partagée entre les participants leur permettant de connaître les actions attendues à n'importe quel moment de l'action conjointe et donc notamment de synchroniser les temps d'entrée dans cette action.

De plus, pour Clark, le dialogue est une activité opportuniste car les interlocuteurs qui s'y engagent ne peuvent pas anticiper ce qu'ils vont y faire. Les actions réalisées par les participants y sont vues comme *locales* et *opportunistes* et son évolution dépend des projets conjoints établis par les participants. Ceux-ci se combinent alors pour former des *projets conjoints étendus* et font évoluer le cours de la discussion dans des directions non anticipées. Cela renforce l'aspect imprévisible du dialogue évoqué précédemment.

2.1.3 Les actes de langage

Nous avons précédemment discuté et caractérisé les principes et la structure globale du dialogue. Nous nous intéressons maintenant à sa structure locale, et présentons la décomposition locale qui en a été faite. Les travaux d'Austin [Aus62] en philosophie du langage ont fait apparaître la notion *d'acte de langage*. Ceux-ci ont pour principe que le langage n'est pas uniquement un moyen de *décrire* le monde, mais aussi un moyen de *d'agir sur* le monde. Un énoncé n'est pas forcément *constatatif*, c'est-à-dire vrai ou faux, mais peut être *performatif*, c'est-à-dire étant ou prenant part à la réalisation d'une action. Par exemple, dans les circonstances appropriées, l'énoncé « je baptise ce bateau le *Reine Elizabeth* » n'est pas une description de ce qui est en train d'être réalisé, mais la réalisation elle-même de l'action consistant à baptiser un bateau. Cet énoncé n'est ni vrai ni faux mais correspond à l'exécution d'une action changeant l'état du monde.

Austin définit les actes de langage sous trois aspects :

L'acte locutoire – l'action physique de dire quelque chose ;

L'acte illocutoire – la représentation des intentions du locuteur envers ses interlocuteurs. Par exemple, informer, donner un ordre, avertir ou s'engager peuvent être des actes illocutoires d'un énoncé ;

L'acte perlocutoire – les effets que le locuteur produit sur ses interlocuteurs. Ils peuvent être désirés ou non. Par exemple, convaincre, persuader, surprendre ou duper peuvent être des actes perlocutoires d'un énoncé.

Par exemple, dire « range ta chambre » à un interlocuteur correspond à énoncer cette phrase (acte locutoire), donner un ordre (acte illocutoire), et déclencher la réalisation de l'action de ranger sa chambre par son interlocuteur (acte perlocutoire).

La vision d'Austin des actes de langage a par la suite été étendue par Searle [Sea69] dans son aspect illocutoire, renforçant l'approche *intentionnelle* des actes de langage. Pour Searle, la réalisation d'un acte de langage est un engagement sur ce qu'il contient. De

1. Traduction de « coordination device. »

ce point de vue, il décrit les actes de langage sous la forme fonctionnelle $F(p)$, où p est le *contenu propositionnel* de l'acte et F la *force illocutoire* de l'acte. La force illocutoire contient notamment un but illocutoire qui indique comment, du point de vue de l'émetteur, la proposition doit être prise en compte par les interlocuteurs. En d'autres termes, c'est l'effet perlocutoire publique voulu. Il existe cinq types de buts illocutoires que peut revêtir un énoncé [SV85, Van90] :

— le but descriptif : le locuteur exprime des faits.

— le but promissif : le locuteur s'engage sur la réalisation d'une action.

— le but directif : le locuteur fait réaliser une action par son ou ses interlocuteurs.

— le but déclaratif : le locuteur change le monde en énonçant ce changement.

— le but expressif : le locuteur exprime un état d'esprit.

Cette liste est complète car elle décrit toutes les *directions d'ajustement*[2] entre le monde et le discours, c'est-à-dire le contenu propositionnel du performatif :

— le discours s'ajuste au monde pour les performatifs descriptifs ;

— le monde s'ajuste au discours pour les performatifs promissifs et directifs ;

— l'ajustement est bidirectionnel pour les performatifs déclaratifs (le locuteur fait correspondre le monde au discours en énonçant que le discours correspond au monde) ;

— la direction d'ajustement est vide pour les performatifs expressifs.

Un acte de langage est associé aux notions de *succès* et de *satisfaction*. Le succès d'un acte de langage tient uniquement compte de la réussite de l'acte de langage en lui-même. Il dépend de son but illocutoire et de son contenu propositionnel. Il est conditionné par des *conditions préparatoires*, qui sont des préalables à leur application (p.ex. celui qui donne un ordre a l'autorité pour le faire), et par des *conditions de sincérité*, contraignant le locuteur à vouloir l'application du but illocutoire de son énoncé. La satisfaction d'un acte de langage ne dépend que de son contenu propositionnel. Si celui-ci devient vrai dans la direction d'ajustement du but illocutoire associé, alors l'acte de langage est satisfait.

Le modèle des actes de langage qui vient d'être présenté est principalement basé sur des fondements philosophiques et théoriques. De ce fait, il présente différentes limites lorsque l'on souhaite l'appliquer à un système dialogique. Une limite importante de ce modèle est qu'il se restreint à l'occurrence d'un seul énoncé de la part d'un locuteur. Cette vision est trop restreinte pour permettre de représenter un dialogue complet, ou même l'enchaînement de quelques énoncés. De plus, comme nous le verrons dans la section 2.1.4, un énoncé peut être *multifonctionnel* et donc réaliser plusieurs actes communicatifs à lui tout seul, contredisant l'une des hypothèses de la théorie des actes de langage. Un aspect important de cette multifonctionnalité est la possibilité d'utiliser le dialogue pour gérer l'interaction elle-même, en plus de la tâche sous-jacente. Cette vision présente le dialogue comme étant une interaction *multidimensionnelle*, contredisant une autre hypothèse de la théorie des actes de langage.

2. Traduction de « directions of fit. »

2.1.4 Les actes de dialogue

L'approche *contextuelle* [BTNB00] des actes de langage a été introduite en linguistique computationnelle comme variante de la théorie classique des actes de langage, y ajoutant le fait qu'un acte de langage n'est pas produit ou interprété de manière isolée. Dans cette approche, l'interprétation d'un acte de langage réalisé au cours d'un dialogue ne peut être faite que dans le contexte même de ce dialogue. De plus, les effets de ces actes de langage, appelés actes de dialogue, sont considérés comme modifiant le contexte du dialogue. Un acte de dialogue est donc vu comme étant une fonction du contexte vers le contexte. Ce contexte est appelé *état d'information*[3] [LT00, TL03].

Selon Bunt, le contexte du dialogue est « la totalité des conditions qui influencent l'interprétation ou la génération d'énoncés dans le dialogue » ([Bun11a], p.215). Il peut être divisé en cinq catégories :

— le contexte linguistique, représentant l'état du dialogue, c'est-à-dire les événements communicatifs passés et les événements communicatifs en cours ;

— le contexte sémantique, représentant l'état de la tâche sous-jacente au dialogue et les propriétés du domaine ;

— le contexte cognitif, représentant le but qui a motivé l'entrée dans le dialogue (c-à-d. la *tâche sous-jacente*). Il inclut aussi les autres participants au dialogue ;

— le contexte physique et perceptuel, représentant les canaux de communication et de perception disponibles ;

— le contexte social, représentant les droits, les obligations, les conventions et les contraintes de communication de chacun.

Chacune de ces dimensions doit être distinguée selon son aspect global, considéré comme figé, et son aspect local, considéré comme dynamique. Pour Bunt, le contexte global ne peut pas être modifié par un acte de dialogue, alors que le contexte local est « la totalité des conditions qui peuvent être changée par l'interprétation d'énoncés du dialogue » ([Bun11a] p.216).

La définition de Bunt d'un acte de dialogue est donnée en 2.4.

Définition 2.4 – Acte de dialogue : « Un acte de dialogue est une section de la description sémantique du comportement communicatif dans le dialogue, spécifiant comment le comportement est sensé changer l'état d'information d'un participant au dialogue qui interprète correctement le comportement. » ([Bun09] p.13)

Les actes de dialogue sont considérés comme des unités fonctionnelles utilisées par le locuteur pour modifier son contexte. Un acte de dialogue s'applique à un *contenu sémantique*, une information ayant un sens particulier dans le contexte résultant de l'application de l'acte de dialogue. Un acte de dialogue est associé à une *fonction communicative*, définissant le sens de cette information en spécifiant de quelle manière le contexte doit être mis à jour pour en tenir compte. Formellement, un acte de dialogue est un opérateur de

3. Traduction de « information state. »

mise à jour de l'état d'information produit par l'application d'une fonction communicative à un contenu sémantique.

Les unités fonctionnelles des actes de dialogue ne correspondent pas directement à un énoncé en langue naturelle, une phrase ou une autre unité linguistique. En effet, les énoncés sont souvent *multifonctionnels* [Bun11b] et un acte de dialogue n'est souvent qu'une partie d'un énoncé. En effet, le dialogue est *multidimensionnel*, c'est-à-dire qu'il permet de gérer en parallèle différents processus (notamment la tâche sous-jacente et le dialogue lui-même). Le dialogue est alors vu comme la réalisation de plusieurs activités en parallèle de la part des interlocuteurs. C'est dans le but de prendre en compte cette *multidimensionalité* du dialogue que Bunt a introduit DIT++ [Bun09], une extension de la taxonomie DIT (pour « Dynamic Interpretation Theory ») [BTNB00]. DIT++ est une taxonomie de fonctions communicatives indépendantes du domaine pour l'annotation, l'analyse et la génération de dialogue. Cette taxonomie ne tient pas uniquement compte des énoncés linguistiques et inclut le comportement communicatif non-verbal comme les mouvements de la tête et les expressions faciales.

La taxonomie DIT++ se décompose en un ensemble de *dimensions*. Chaque activité ayant lieu au sein du dialogue est réalisée dans une dimension. Le concept de dimension est défini de la manière suivante :

Définition 2.5 – Dimension : « Une dimension est un aspect de la participation au dialogue :

— au sein duquel les participants peuvent intervenir en utilisant des actes de dialogue ;

— pouvant être traité indépendamment des autres aspects de la participation au dialogue.

» ([Bun11b] p.227)

DIT++ sépare l'interaction entre dix dimensions [Bun09], chacune focalisée sur un type d'information déterminant son type de contenu sémantique :

Task/Activity — Les actes de dialogue dont la réalisation contribue à l'avancée de la tâche ou de l'activité sous-jacente au dialogue ;

Auto-Feedback — Les actes de dialogue apportant des informations sur le traitement par le locuteur de l'énoncé précédent ;

Allo-Feedback — Les actes de dialogue utilisés par le locuteur pour exprimer ses opinions sur le traitement d'un de ces précédents énoncés par son interlocuteur, ou pour solliciter des informations à propos de ce traitement ;

Contact Management — Les actes de dialogue permettant d'établir et d'assurer le contact entre les interlocuteurs ;

Turn Management — Les actes de dialogue permettant de s'accorder sur le rôle de locuteur ;

Time Management — Les actes de dialogue signalant que le locuteur a besoin de temps pour formuler sa contribution ;

Discourse Structuring — Les actes de dialogue pour explicitement structurer la conversation ;

Own Communication Management — Les actes de dialogue indiquant que le locuteur modifie sa contribution courante ;

Partner Communication Management — Les actes de dialogue du participant n'étant pas le locuteur par lesquels il assiste ou corrige le locuteur dans sa contribution ;

Social Obligation Management — Les actes de dialogues dédiés aux conventions sociales, comme les salutations et excuses en cas d'erreur.

Les fonctions communicatives utilisées dans DIT++ sont séparées en deux catégories : les fonctions *générales* et les fonctions *spécifiques à une dimension*. Nous ne traitons ici qu'une sous-partie des fonctions générales de DIT++, leur intégralité ainsi que les fonctions spécifiques sont disponibles dans les travaux de Bunt [Bun09] ainsi que sur le site internet de DIT++ : `https://dit.uvt.nl/` [4]. Les fonctions générales de DIT++ sont données en figure C.1. La hiérarchie des fonctions de DIT++ est divisée en deux grandes catégories : les fonctions de *transfert d'information* et les fonctions de *discussion d'action*. Les fonctions de transfert d'information regroupent d'un côté les fonctions de *recherche d'information* et de l'autre les fonctions permettant de *fournir de l'information*. Les fonctions de discussion d'action sont soit *promissives*, soit *directives*. Les fonctions présentes dans cette hiérarchie sont soit mutuellement exclusives (comme les fonctions *Confirm* et *Disconfirm*), soit une spécialisation de leur fonction parente (la fonction de vérification positive *PosiCheck* est une spécialisation de la fonction de question oui/non *PropositionalQuestion*).

2.2 Approches intentionnelles

Cette section présente différentes approches, dites « intentionnelles », développées à partir des actes de langage pour permettre à des agents de communiquer. Les approches intentionnelles présentent les énoncés produits par les interlocuteurs comme le résultat de l'instanciation d'un plan établi à partir des intentions de l'agent [Hul00]. La section 2.2.1 présente les langages de communication agent et montre leurs limites pour l'interaction humain-agent. La section 2.2.2 présente les approches par planification du dialogue pour une interaction humain-agent. La section 2.2.3 présente la théorie de la conversation collaborative.

2.2.1 Langages de communication agent

Les actes de langage ont servi de point de départ à la création de modèles de communication pour les systèmes multi-agents, portant le nom de *langages de communication agent* [5]. L'un des premiers apparu est KQML (Knowledge Query and Manipulation Language) [FFMM94]. Il décrit un format d'encapsulation des messages avec lequel un agent

4. Consulté le 22/06/2018
5. Traduction de « Agent Communication Langage. », ou ACL.

logiciel peut explicitement énoncer la force illocutoire d'un message, indépendamment du contenu du message. Chaque message est composé d'un *performatif*, correspondant à un acte de langage, et d'un ensemble d'arguments correspondant au contenu du message.

Malgré les apports de KQML, celui-ci fut critiqué sur plusieurs aspects [Woo02]. Le manque de définition stricte des performatifs et des mécanismes de transport d'un message a amené à différentes implémentations de KQML n'étant pas interopérables. De même, le manque de sémantique rigoureuse des messages KQML rend impossible la validation d'une utilisation de KQML par un agent. Le manque de performatif promissif diminue fortement l'expressivité de KQML, empêchant l'engagement d'un agent envers un autre. De plus, l'ensemble des performatifs de KQML est vu comme trop important et *ad hoc*.

Ces critiques ont mené à l'introduction par FIPA [6] de FIPA-ACL, un langage proche de KQML[ON98]. Un message FIPA-ACL est défini par un ensemble de paramètres [FA02a]. Les paramètres nécessaires à l'interpértation du message varient en fonction du contexte et le seul paramètre obligatoire est `performative`. La liste des paramètres est donnée dans le tableau 2.2.

Paramètre	Catégorie
`performative`	Type de l'acte de communication
`sender`	Participant à la communication
`receiver`	Participant à la communication
`reply-to`	Participant à la communication
`content`	Contenu du message
`language`	Description du contenu
`encoding`	Description du contenu
`ontology`	Description du contenu
`protocol`	Contrôle de la conversation
`conversation-id`	Contrôle de la conversation
`reply-with`	Contrôle de la conversation
`in-reply-to`	Contrôle de la conversation
`reply-by`	Contrôle de la conversation

TABLEAU 2.2 – Paramètres d'un message FIPA-ACL [FA02a].

Les performatifs (correspondant au paramètre `performative` d'un message FIPA-ACL) ne peuvent prendre de valeur que dans un ensemble pré-défini d'actes de communication : *Accept Proposal*, *Agree*, *Cancel*, *Call for Proposal*, *Confirm*, *Disconfirm*, *Failure*, *Inform*, *Inform In*, *Inform Ref*, *Not Understood*, *Propagate*, *Propose*, *Proxy*, *Query If*, *Query Ref*, *Refuse*, *Reject Proposal*, *Request*, *Request When*, *Request Whenever* et *Subscribe* [FA02b].

De manière à ne pas répéter les lacunes sémantiques de KQML, FIPA-ACL est basé sur la théorie de Cohen et Levesque voyant les actes de langage comme des *actions*

6. Foundation for Intelligent Physical Agents (`http://www.fipa.org`), intégrée à l'IEEE en 2005 en tant que comité de standards.

rationnelles [CL90a, CL90b], par la suite étendue par Sadek [Sad91]. La sémantique est donnée dans le langage SL (pour Semantic Language) [FA02c] qui permet de décrire les états mentaux des interlocuteurs en termes de *croyances*, de *croyances incertaines*, de *désirs* et de buts persistants. Plus précisément, un message FIPA-ACL décrit à l'aide de SL les conditions de *faisabilité* d'un message, qu'il faut respecter pour pouvoir l'envoyer, ainsi que *l'effet rationnel* de ce message, c'est-à-dire ce que l'agent souhaite réaliser en l'envoyant (correspondant à l'acte perlocutoire publique d'Austin).

L'approche de FIPA-ACL permet de s'éloigner de la vision locale d'un simple performatif en formalisant ses pré- et post-conditions. L'acte de langage auquel correspond un message FIPA-ACL est contextualisé dans le cadre plus général d'un échange entre plusieurs agents, ouvrant la voie à la description d'enchaînements de messages et donc d'une conversation entre agents.

Même si FIPA-ACL est une norme conçue indépendamment de toute architecture d'agent, sa sémantique est alignée sur le modèle d'agent de type BDI [7] de Rao et Georgeff [RG91]. En pratique, cela contraint les agents utilisant FIPA-ACL de manière sémantiquement cohérente à être conçus dans une architecture proche de BDI [Pas04].

Les approches qui, comme KQML et FIPA-ACL, se réfèrent à des états mentaux des agents, sont qualifiées de *mentalistes* [Sin98]. Elles considèrent les structures conversationnelles comme un simple épiphénomène, en partant du principe que les structures de la conversation émergent d'une succession d'actes de communication et plus spécifiquement des conséquences de ces actes sur les états mentaux des participants. Cette vision, bien qu'attirante par la flexibilité qu'elle permet et par l'indépendance vis-à-vis d'un modèle structurel particulier est contestable à la fois sur le plan pratique et sur le plan théorique [CDLBP06] :

— d'un point de vue pratique, la sémantique des actes de langage est si riche qu'il est très compliqué de déterminer les réponses possibles simplement en inférant les états mentaux des autres, car il y a trop de possibilités sémantiquement cohérentes pour poursuivre le dialogue. La restriction de l'espace des réponses possibles à un unique acte dans ce contexte est généralement un problème difficile, pouvant mener les agents à de nombreux raisonnements déductifs dans le but de choisir la « bonne » poursuite au dialogue ;

— d'un point de vue théorique, les approches mentalistes supposent deux hypothèses qui peuvent être remises en question : les états mentaux des agents sont « vérifiables » et les agents sont sincères. Or, ces hypothèses sont difficilement assurées dans un environnement ouvert, où les architectures d'agents peuvent être hétérogènes et où il n'est pas forcément possible d'accéder directement à leurs états mentaux et de vérifier leur conformité à des règles.

Cette dernière remarque est d'autant plus vraie lorsqu'il s'agit de faire interagir un agent avec un être humain. En effet, il n'est pas concevable de se représenter les états mentaux d'un être humain de manière exacte et formelle, et les êtres humains ne sont pas rationnels comme peuvent l'être des agents logiciels.

7. « Belief, Desire and Intention » ou « Croyance, Désirs et Intentions » en français.

2.2.2 Approches par planification du dialogue

Nous avons défini dans le chapitre précédent les concepts de plan et de reconnaissance de plan (cf. section 1.3.1). Nous présentons ici leur utilisation pour le dialogue humain-agent. Cette vision considère que les actes de dialogue sont des actions qui sont contextualisables dans un plan de l'interaction que l'on peut relier avec la tâche sous-jacente au dialogue.

La première approche pour la structuration et la planification du dialogue a utilisé un outil proche de STRIPS [FN71]. STRIPS est un système de planification générique qui à partir d'une représentation de l'environnement initial e, d'un but à atteindre E et d'un ensemble d'actions réalisables Ac, génère une séquence d'actions $a \in Ac^*$ permettant, quand exécutée à partir de l'état e, d'atteindre l'état E. e, E et Ac sont représentés en logique du premier ordre.

Cohen et Perrault [CP79] et Allen et Perrault [AP80] ont utilisé cette approche pour le dialogue, en considérant qu'un acte de langage correspond à un opérateur de plan, c'est-à-dire à une action réalisable présente dans Ac. De cette manière, les actes de langage sont considérés comme des actions physiques. Cette approche est rendue possible par le fait que les actes de langage dans la vision de Searle [Sea69] possèdent des préconditions et des effets. Les interlocuteurs possèdent comme états mentaux des buts (appelés *wants*) et différents types de croyances.

Un exemple est donné dans le tableau 2.3, qui présente l'opérateur de plan INFORMER correspondant à la formalisation sous forme d'action de l'acte de langage d'information sur une proposition P, ou L est le locuteur et I est l'interlocuteur.

INFORMER(L, I, P)	
Préconditions	L VEUT INFORMER(L, I, P)
	L SAIT P
Effets	P I SAIT P
Corps	L DIRE P à I

TABLEAU 2.3 – Représentation d'un opérateur INFORMER.

Cette formalisation permet d'appliquer une technique de reconnaissance de plan sur les observations faites sur son interlocuteur. La reconnaissance de plan s'appuie sur les règles suivantes :

Règle de Précondition-Action – L VEUT P ⇒ L VEUT A, avec P une précondition de plan et A une action ;

Règle de Corps-Action – L VEUT C ⇒ L VEUT A, avec C le corps d'un plan et A une action ;

Règle d'Action-Effet – L VEUT A ⇒ L VEUT E, avec A une action et E un effet.

La reconnaissance de plan est utilisée pour permettre à un participant au dialogue de déterminer les buts non implicites de son interlocuteur, prédire ses actions futures et anticiper les difficultés pouvant empêcher la réalisation de ses actions. De cette manière, il peut avoir un comportement coopératif en *adoptant le ou les buts* de son interlocuteur.

Cette approche présente deux problèmes : d'un côté, elle restreint l'analyse à un seul énoncé, alors que Carberry [Car90] a montré que la reconnaissance de plan se fait sur des échanges de plusieurs énoncés. De l'autre côté, elle considère que la structure du dialogue est isomorphe à celle de la tâche sous-jacente. En effet, les interlocuteurs peuvent par exemple s'engager au cours du dialogue dans un sous-dialogue de clarification qui n'est pas prévu dans la tâche.

Cette seconde critique a amené Litman et Allen [LA90] à montrer que deux types de plans distincts sont exécutés au cours d'un dialogue :

Les plans du domaine – une représentation des tâches extra-linguistiques qui doivent être réalisées dans le domaine sous-jacent à un dialogue orienté-tâche ;

Les plans du discours – des plans indépendants du domaine pouvant être générés par d'autre plans au cours de n'importe quel dialogue orienté tâche.

2.2.3 Théorie de la conversation collaborative

Les objectifs de segment de conversation

La structuration du dialogue par une planification classique a été contestée car elle ne tient compte que des plans *individuels*, restreints à un agent. En effet, le dialogue étant une activité collaborative, sa planification est vue comme une activité collaborative [GS90]. Cependant, certains comme Grosz et Kraus [GK96], ont montré que la collaboration ne peut pas être modélisée comme une simple combinaison de plans individuels. Rich et Sidner [RSL01] partent du principe que la collaboration et la conversation [8] sont deux concepts qui s'étendent mutuellement dans le modèle de la *conversation collaborative* [9]. Celui-ci repose sur le modèle de conversation de Grosz et Sidner [GS86], séparant celle-ci en trois parties :

— la structure *intentionnelle*, qui contient les croyances et intentions des participants au dialogue vis-à-vis des tâches et sous-tâches à réaliser ;

— la structure *attentionnelle*, qui est une abstraction dynamique du centre d'intérêt des participants au cours de la conversation ;

— la structure *linguistique*, qui consiste en des séquences d'énoncés (appelés segments) contribuant à une tâche ou sous-tâche particulière.

La structure attentionnelle est essentielle pour expliquer le traitement des énoncés de la conversation et la structure intentionnelle joue un rôle central dans la gestion de la structure de la conversation, lui donnant une cohérence et se rattachant directement au concept de conversation. La structure linguistique est constituée de *segments de conversation* [10] intégrant la relation les liant les uns aux autres. Ces segments peuvent correspondre à plusieurs énoncés successifs au cours de la conversation.

Grosz et Sidner partent du principe que les participants à une conversation ont un objectif qui va jouer un rôle fondamental dans la structure de la conversation. Ces

8. Traduction de « discourse. »
9. Traduction de « collaborative discourse. »
10. Traduction de « discourse segments », ou DS.

objectifs sont appelés *objectifs de conversation*[11] et représentent l'intention sous-jacente à la participation à une discussion. De plus, ce sont ces objectifs qui expliquent le contenu des contributions réalisées par les participants au dialogue. Cette intention est déclinée à l'échelle d'un segment de conversation, amenant à l'introduction du concept *d'objectif de segment de conversation*[12], ou DSP. Un DSP exprime la contribution d'un segment à l'atteinte des objectifs globaux de la conversation. Il traduit aussi une intention de l'émetteur, pouvant par exemple être :

— faire réaliser une action physique par un autre agent ;

— amener un autre agent à croire en un fait ;

— amener un autre agent à croire qu'un fait en appuie un autre ;

— faire identifier un objet par un autre agent ;

— faire connaître à un autre agent les propriétés d'un objet.

Deux relations structurant le discours ont été identifiées entre les DSP :

La domination – DSP1 domine DSP2 quand la satisfaction de DSP2 contribue à la satisfaction de DSP1 ;

La satisfaction-précédence – DSP1 pré-satisfait DSP2 si DSP1 doit être satisfait avant DSP2.

Grosz et Sidner [GS90] avancent que les participants à une conversation doivent identifier les DSP et leurs relations pour être capables de traiter les énoncés suivants du dialogue. Cette vision rejoint celle de Grice [Gri69] selon laquelle les intentions sous-jacentes à un énoncé visent à être identifiées et prises en compte par son destinataire. Elle implique un lien fort entre la structure intentionnelle sous-jacente à l'activité dialogique et la structure du dialogue lui-même.

Il est donc nécessaire, dans ce modèle, de mettre en œuvre des mécanismes de reconnaissance de plan de manière à inférer le plan du locuteur sur la base des segments d'information qu'il fournit à ce sujet. La reconnaissance de plan suppose que les plans reconnus par l'interlocuteur sont ceux que le locuteur veut voir reconnus. Elle requiert une hypothèse de coopération, qui suppose une volonté de chaque participant d'élaborer un plan partagé de la tâche. Chaque participant au dialogue établit des plans pour poursuivre ses propres buts et collabore à l'élaboration de plans pour réaliser les buts des autres. Les participants à une conversation sont donc à la fois des acteurs et des agents inférant mutuellement leurs plans. Leurs actions conversationnelles doivent donc fournir assez d'information sur leurs croyances et leurs intentions pour que leurs interlocuteurs soient capables de les situer dans le plan collaboratif représentant la conversation. Les énoncés émis par les participants sont alors perçus comme des opérateurs de plan, modifiant l'état de la tâche sous-jacente et donc le contexte du dialogue et amenant de ce fait la possibilité d'émettre de nouveaux énoncés.

Cette structure intentionnelle du discours a été utilisée pour l'élaboration des systèmes basés sur SharedPlans [GS90], déjà évoqué en section 1.3.1. Le formalisme de SharedPlans

11. Traduction de « discourse purpose », ou DP.
12. Traduction de « discourse segment purpose. »

modélise comment les intentions et les croyances mutuelles se cumulent au cours du dialogue. Il part du principe que les participants à l'interaction possèdent des plans individuels et qu'ils élaborent des plans partagés pour réaliser une action en groupe. Ces plans partagés peuvent être partiels et complétés tout au long de l'interaction. Le lien entre les plans de domaine élaborés avec SharedPlans et la planification du dialogue en langue naturelle a été mis en œuvre par Lochbaum [Loc94, Loc98]. Elle voit notamment le dialogue comme un outil pour compléter les plans partagés, ceux-ci modélisant exactement les DSP.

COLLAGEN et Disco for Games

Les systèmes collaboratifs humain-machine COLLAGEN [RSL01] et son successeur Disco for Games [RS12] se basent sur la théorie de la conversation collaborative et se servent du formalisme de SharedPlans. Ceux-ci représentent la tâche à réaliser sous la forme d'un modèle abstrait hiérarchique et partiellement ordonné d'actions contribuant à la réalisation de buts dans le domaine d'application. L'état de la conversation est l'ensemble des croyances et intentions des participants dans le contexte courant. Il comprends un mécanisme de *centre d'intérêt* permettant de suivre les changements dans le contexte de la tâche et de la conversation. Il contient un arbre représentant le plan global décomposé en sous-buts, ainsi qu'une pile de centre d'intérêts, chaque centre d'intérêt représentant un contexte particulier. L'élément au sommet de cette pile est le *but actuel* de la discussion.

L'interprétation des nouveaux événements dialogiques consiste à mettre à jour l'état de la conversation en expliquant comment chaque nouvel événement contribue au but actuel. Une contribution au but actuel peut prendre l'une des formes suivantes :

— accomplit le but actuel ;

— rentre dans un modèle de tâche pour accomplir le but actuel ;

— sélectionne une alternative parmi plusieurs pour accomplir le but actuel ;

— identifie quel participant doit accomplir le but actuel ;

— identifie un paramètre du but actuel.

Si l'événement dialogique interprété contribue au but actuel de la discussion, l'arbre représentant les plans est mis à jour pour inclure cet événement et la pile de centre d'intérêts peut être empilée ou dépilée. Pour la génération de discours, le principe est sensiblement l'inverse de l'interprétation : en fonction de l'état actuel du discours, COLLAGEN crée une liste priorisée d'énoncés et d'actions contribuant au but actuel selon les cinq critères qui viennent d'être présentés.

COLLAGEN a été utilisé comme outil de représentation de modèles de tâche pour DiamondHelp [RS07], un système collaboratif générique assistant un utilisateur sur une tâche particulière (utilisation d'une machine à laver, d'un lecteur DVD ou d'un thermostat par exemple).

TRAINS et TRIPS

TRAINS [ASF+95, FAM96, AFM+00] est un autre système développé dans les années 90 pour gérer un dialogue avec un humain dans le cadre d'une résolution collaborative de problème. Il y a eu plusieurs versions successives du système TRAINS, dont le but était de permettre à un utilisateur d'interroger une base de données d'horaires de trains en langue naturelle. Le modèle TRAINS considère aussi les actes de langage effectués par les participants comme des opérateurs de plan. Son gestionnaire de dialogue est conçu pour interpréter les actes de dialogue dans le contexte courant, invoquer les actions de résolution du problème, formuler des réponses et assurer la représentation du dialogue par le système. À l'instar de la structure attentionnelle de Grosz et Sidner [GS86], il contient une pile dont chaque élément représente un contexte pour interpréter les énoncés composés de [AFM+00] :

— le but du domaine ou du discours motivant le segment actuel ;

— le nœud sur lequel le système est actuellement focalisé dans la hiérarchie de la résolution du problème ;

— l'objet de focalisation et son historique pour le segment ;

— des informations sur le statut de résolution du problème.

Cette structure permet de contextualiser un énoncé formulé par un utilisateur humain dans le plan de la tâche associée au dialogue.

TRAINS a évolué pour intégrer, en plus de la gestion du dialogue en langue naturelle, l'utilisation d'éléments graphiques comme des cartes, donnant naissance au système TRIPS [FA98].

Critique des approches intentionnelles

Les approches intentionnelles apportent une grande souplesse à la structure du dialogue, permettant de dynamiquement situer un énoncé dans le contexte d'une conversation dont la structure s'articule directement autour de l'action menée collaborativement. Le réseau hiérarchique de tâches permet par exemple de contextualiser dans des sous-tâches des sous-dialogues ouverts de manière opportuniste.

Cependant, dans le cadre humain-machine les approches intentionnelles considèrent que les êtres humains sont des agents rationnels, utilisant des plans déterministes clairement organisés et détaillés à priori. Cela laisse de côté les aspects dynamique et opportuniste de la structure dialogique [Cla96], qui pourtant sont au centre du dialogue, rendant celui-ci difficilement planifiable sur le long terme de manière définitive.

En outre, pour Grosz et Sidner [GS86], si la structure linguistique traduit la structure intentionnelle, cette structure n'est *ni identique, ni isomorphe* à la tâche sous-jacente. En effet, la structure intentionnelle contient des sous-structures propres à l'exécution de la tâche du dialogue. Le dialogue ne peut donc pas se structurer autour d'un plan de la tâche.

De plus, la reconnaissance d'une intention sous-jacente à tout énoncé n'est pas pertinente dans des échanges ritualisés comme les salutations. C'est en fait toute la structure sociale du dialogue qui est ignorée dans ces approches [TA94].

2.3 Approches conventionnelles et sociales

L'utilisation directe d'actes de langage présentée en section précédente est limitée par le fait qu'elle a recours à des actes de langage *isolés*. En effet, il sont insuffisants pour exprimer des règles sur des séquences de messages, qui peuvent avoir lieu au cours du dialogue. Or, ces *conversations* ne se limitent pas à l'émission par les interlocuteurs de messages correspondant à leurs intentions car ceux-ci doivent aussi prendre en compte dans leur processus décisionnel les messages et actions des autres participants. Paradoxalement, la prise en compte de ce contexte peut *simplifier* la complexité combinatoire de la sélection de messages pour un locuteur, en appliquant des *motifs* pré-compilés ou stéréotypés. Le respect de ces motifs restreint l'espace des possibles pour l'interprétation et la génération d'énoncés tout en garantissant la cohérence sémantique des messages du langage utilisé [CDLBP06].

Dans le cas d'une communication agent-agent, différentes structures ont été utilisées pour représenter les protocoles d'interaction entre des agents logiciels, comme les automates finis [GB99] les réseaux de Pétri [CCF+00] et AUML, une version modifiée d'UML dédiée à l'interaction entre agents introduite par Odell [OPB00]. Plus particulièrement, le langage FIPA-ACL intègre une spécification de protocoles dialogiques sous la forme de diagrammes de séquence AUML utilisant les actes de communication spécifiques à FIPA-ACL [FA02d].

Cependant, l'extension de l'interaction agent-agent à l'interaction humain-agent est contestable et l'usage d'un langage de communication agent explicite n'est pas adapté pour modéliser une interaction humain-agent [Nwa96]. Nous nous limitons donc dorénavant à des structure d'interaction dialogique applicables pour une communication impliquant au moins un être humain. La section 2.3.1 présente les protocoles et politiques de conversation. La section 2.3.2 présente le tableau de conversation. La section 2.3.3 présente les engagements sociaux. La section 2.3.4 présente les approches par jeux de dialogue.

2.3.1 Protocoles et politiques

Protocoles pour les communications humaines

Pauchet [Pau06] utilise des automates temporisés pour modéliser la dynamique du discours entre deux humains dans un contexte de planification en connaissances incomplètes. Ce modèle s'appuie sur la théorie des actes de langage pour modéliser les énoncés. Ceux-ci sont associés à des performatifs appliqués à des états mentaux. Ce schéma s'intègre dans une architecture BDI appelée BDIGGY qui articule l'interaction, la planification et les connaissances au sein du système.

Les automates temporisés sont une extension des automates finis [AD94] permettant de modéliser l'ordonnancement temporel d'actions en fournissant des informations quantitatives sur le délai séparant deux actions. Il fait appel à des horloges prenant des valeurs réelles positives, avançant toutes à la même vitesse et mises à zéro à l'initialisation du système. Une transition dans un automate temporisé est faite de trois éléments :

— une *garde*, qui est une condition sur la valeur des horloges ;

— une *étiquette*, représentant une action comme dans les automates finis ;

— des *actions de remise à zéro* de certaines horloges.

Lors de l'occurrence d'une action, une transition ayant une étiquette correspondante ne peut être franchie que si la garde est respectée par les valeurs courantes des horloges, déclenchant alors les éventuelles actions de remise à zéro.

Pauchet associe deux automates à chaque motif d'interaction identifié : un pour l'initiateur et un pour l'interlocuteur. Un seul automate pourrait suffire à représenter l'intégralité du motif mais cette séparation permet de mettre en avant le point de vue de chaque interlocuteur. L'ensemble PER des performatifs identifiés lors de cette expérimentation est le suivant : PER = { *inform*, *reply*, *notUnderstood*, *thank*, *query*, *acceptProposal*, *refuseProposal*, *cancel*, *refine*, *propose*}. Ces performatifs permettent de représenter les étiquettes des transitions des automates, associés à la direction de la transition. Avec $p \in$ PER, l'étiquette *Receive*(p) représente un message reçu et *Send*(p) représente un message envoyé. La figure 2.1 présente l'automate de demande d'information du point de vue de l'initiateur du modèle de Pauchet.

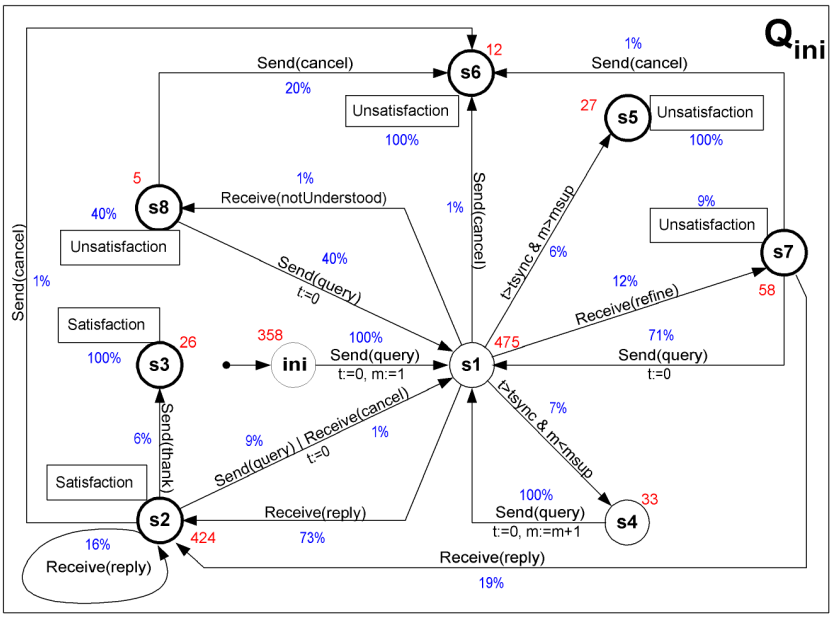

FIGURE 2.1 – Automate de demande d'information du point de vue de l'initiateur du modèle de Pauchet [Pau06].

Les politiques de conversation

Cette vision de la conversation en tant que contexte normatif pour l'enchaînement d'actes de communication a été décrit par Greaves *et al.* [GHB00] comme étant des *politiques de conversation* [13].

13. Traduction de « conversation policies. »

Définition 2.6 – Politique de conversation : « Les politiques de conversation sont des *contraintes* publiquement partagées sur l'espace potentiellement illimité des séquences de messages ACL pouvant être utilisées pour réaliser un but. » ([GHB00] p. 123)

Les auteurs donnent les caractéristiques que doivent respecter les politiques de conversation :

Séparation politique publique/implémentation privée – la séparation doit être nette entre une politique de conversation et la théorie régissant le comportement d'un agent, distinguant les politiques *publiques* des implémentations *privées* ;

Abstraction des politiques – les politiques de conversation doivent donner un niveau d'abstraction permettant l'identification de classes de conversation parmi différents formalismes de spécification. Un but important de la théorie des politiques de conversation est la production de critères permettant d'identifier des politiques de conversation identiques d'une implémentation à une autre ;

Composition des politiques – les politiques de conversation doivent pouvoir être composées, permettant d'en assembler différentes pour permettre de mener des conversations spécifiques à un domaine ;

Flexibilité des politiques – les politiques de conversation doivent être assez flexibles pour permettre l'interopérabilité entre des agents ayant différents niveaux de sophistication.

Les politiques de conversation sont habituellement représentées à l'aide de d'automates à états finis. L'exemple le plus répandu est le « request for action » de Winograd et Flores [WF87]. Sur ce type d'automates, un état représente un état de la conversation et une transition entre deux états représente un acte de communication permettant de passer de l'un à l'autre.

Ces politiques de conversation présentent cependant certains désavantages [CDLBP06] :

— plus la taille de l'automate associé à la politique de conversation est élevée, moins celle-ci est flexible et plus elle risque de poser un problème de temps de calcul ;

— les politiques de conversation ne se basent pas sur des structures sociales et se limitent à un ensemble d'états mentaux des participants sans faire appel à une organisation sociale plus proche du dialogue [Sin98].

2.3.2 Le tableau de conversation

Clark [Cla96] explique que l'étude linguistique du dialogue n'est pas suffisante et qu'il est nécessaire pour les participants au dialogue de partager un *terrain commun*[14], qui est un contexte cognitif élaboré en commun au cours du dialogue. En effet, celui-ci considérant que le dialogue est une activité conjointe (cf. section 2.1.2), le partage d'un terrain commun est la condition *sine qua non* du dialogue, permettant de garder une trace du passé du dialogue. La synthèse de ce terrain commun lors d'un dialogue est l'une des activités

14. Traduction de « common ground. »

principales du dialogue et l'action *d'établissement*[15] [CS89] est dédiée à la vérification par les interlocuteurs qu'ils partagent bien le même terrain commun. L'établissement permet la construction d'un terrain commun de manière ordonnée et cohérente.

Clark avance que le terrain commun est une *base partagée* ou de la *connaissance partagée* au sens de Lewis [Lew69] et donnée en définition 2.7.

Définition 2.7 – Terrain commun : « p fait partie du terrain commun pour les membres d'une communauté C si et seulement si :

— chaque membre de C a l'information que la base b est établie ;

— b indique à chaque membre de C que chaque membre de C possède l'information que b est établie ;

— b indique aux membres de C que p.

» ([Cla96] p.94)

Le terrain commun est à rapprocher de la notion *d'état d'information* évoqué lors de la présentation des approches contextuelles (cf. 2.1.4). La définition de Larsson et Traum [LT00] de l'état d'information est donnée en définition 2.8.

Définition 2.8 – État d'information : « L'état d'information d'un dialogue représente l'information nécessaire pour le distinguer d'autres dialogues, représentant les additions cumulées issues d'actions précédentes dans le dialogue, et motivant des actions futures. » ([LT00] p.327)

La théorie de Larsson et Traum du dialogue basé sur l'état d'information est composée de [TL03] :

Une description des constituants infomationnels – incluant les aspects du contexte partagé et les facteurs internes ;

Une représentation formelle de ces constituants – utilisant des structures données comme des ensembles, des listes, des propositions ou des opérateurs modaux associés à une logique ;

Un ensemble de coups dialogiques – déclenchant la mise à jour de l'état d'information. Ils sont généralement associés à des actions telles que des énoncés en langue naturelle ;

Un ensemble de règles de mise à jour – régissant la mise à jour de l'état d'information compte tenu de l'état courant de l'état d'information et des coups dialogiques joués ;

Une stratégie de mise à jour – permettant de décider quelle règle appliquer à quel moment.

Il est courant de désigner la modélisation du dialogue par des approches conventionnelles en parlant de *tableau de conversation*. Plus précisément, le tableau de conversation est la partie du terrain commun représentant l'état du dialogue entre les participants. Il

15. Traduction de « grounding. »

est donc considéré comme la *partie publique* de l'état d'information modélisant le *contexte dialogique*. Ginzburg [Gin12] présente différentes combinaisons possibles entre les parties publiques et privées des états d'information des participants au dialogue (A et B étant les participants) :

1. A : ⟨A.privée⟩, B : ⟨B.privée⟩
2. A : ⟨publique⟩, B : ⟨publique⟩
3. A : ⟨A.privée, A.B.privée⟩, B : ⟨B.privée, B.A.privée⟩
4. A : ⟨publique, A.privée⟩, B : ⟨publique, B.privée⟩
5. A : ⟨A.publique, A.privée⟩, B : ⟨B.publique, B.privée⟩

Le premier modèle est une vision extrême considérant qu'il n'existe pas de partie publique dans l'état d'information. Il nie donc le concept de tableau de conversation partagé entre les interlocuteurs. À l'inverse, le deuxième modèle considère que la modélisation du dialogue est réalisable en employant uniquement des informations publiques (c-à-d. strictement partagées entre les interlocuteurs) et que celles-ci suffisent à expliquer les comportements dialogiques des participants. Dans le troisième modèle, A.B.privé est la représentation de A des attitudes privées de B. Ce modèle rejette l'idée d'une partie publique entre les deux participants et divise la partie privée en deux sous-parties : la première étant leurs propres attitudes privées et la seconde représentant les attitudes privées de leur interlocuteur. Le quatrième modèle part du principe que la partie publique est strictement partagée entre les interlocuteurs et qu'ils ont chacun une partie privée. Dans ce modèle, les interlocuteurs n'essaient pas de se représenter la partie privée de leur partenaire. Le dernier modèle présente est une vision plus centrée sur les agents, où chacun à sa propre version de la partie publique de l'état d'information. Cette vision donne la possibilité d'ajouter une information incertaine ou inexacte au tableau de conversation, amenant à une divergence entre les parties publiques des états d'information des interlocuteurs. La correction de cette divergence est réalisée à l'aide d'un processus de *grounding*.

Ginzburg [Gin94, Gin96, Gin12] a développé une structure représentant l'état d'information des participants au dialogue. Le tableau de conversation constitue la partie publique de cet état d'information. Il est *quasi-partagé*, c'est-à-dire que chaque participant en possède sa propre copie qu'il met à jour [Gin96]. Il s'agit donc de la partie publique du cinquième modèle d'état d'information introduit plus haut. Ginzburg y fait référence comme étant le « dialogue gameboard » [16] abrégé DGB, et le structure de la manière suivante :

FACTS – Un ensemble de faits communément établis ;

QUD – Les « questions under discussion », un ensemble partiellement ordonné spécifiant les questions actuellement discutées au cours du dialogue. Si une question q est maximale dans QUD, alors il est possible de fournir des informations spécifiques à q. Larsson [Lar02] précise que les questions présentes dans le QUD sont ouvertes à la discussion, résolvables par une ellipse, posées de manière explicite et pas encore résolues ;

16. Littéralement « planche de jeux. »

LATEST-MOVE – Contient le *dernier coup joué* : il est possible de jouer n'importe quel coup autorisé comme réaction au dernier coup.

La partie privée de l'état d'information de Ginzburg est appelée *unpublicized mental situation*, noté UNPUB-MS(DP) [17]. Elle représente les états mentaux privés des participants au dialogue, et contient typiquement leurs buts, et capacités inférentielles. Formellement, Ginzburg représente un participant au dialogue comme un ensemble de triplets $<GB$, ms, $t>$, représentant un tableau de conversation GB, avec des états mentaux ms à un temps t. L'évolution du tableau de conversation de chaque participant au dialogue dépend du contenu de son UNPUB-MS. En effet, la résolution d'une question présente dans QUD est conditionnée par la capacité de l'agent à décider si un coup dialogique apporte une réponse résolvante à une question de QUD.

La structure de l'état d'information et la théorie QUD de Ginzburg ont servi de base pour le développement des systèmes de dialogue GoDiS et IBiS par Larsson [LT00, LLC+00, Lar02]. Les différentes version de IBiS ont itérativement raffiné la structure de l'état d'information, tout en conservant la distinction public/privé établie par Ginzburg. Nous ne détaillons pas ici la structure finale établie par Larsson mais celle-ci est disponible dans ses travaux de thèse [Lar02]. Cependant, nous insistons sur le fait que les approches basées sur un tableau de conversation permettent de gérer le dialogue indépendamment de la tâche.

Un défaut majeur des approches donnant un rôle central au champ QUD de leur tableau de conversation est le fait qu'elles se focalisent sur le motif question/réponse. Même si certaines sont assez perfectionnées et dépassent le motif de la paire adjacente question/réponse, elles négligent d'autres types d'énoncés comme les requêtes, suggestions, salutations, etc. [Dub14].

2.3.3 Les engagements sociaux

Théorie de l'engagement social

Singh [Sin91, Sin98] a mis en lumière le besoin d'avoir une *interaction sociale* entre les agents utilisant des ACL. La théorie de l'interaction sociale se base sur le fait que la capacité à communiquer d'un agent dépend de son obéissance aux *conventions* ayant cours dans la société dont il fait partie. Cette vision rejette l'idée d'une communication basée sur des états mentaux, au sein de laquelle les agents communiqueraient principalement en termes de concepts mentaux, comme des croyances et des intentions. En effet, celle-ci revient à *lire dans l'esprit* des autres agents et ne permet pas la communication au sein d'un ensemble d'agents dont les architectures sont hétérogènes. Singh considère que les actions communicatives des agents font partie de l'interaction sociale ayant lieu entre eux et fait la distinction entre les *états internes* privés d'un agent et son *comportement externe* public. De ce fait, même s'il n'est pas possible de déterminer les états mentaux d'un agent, il est certain qu'il doit être capable d'interagir socialement pour communiquer.

Dans le modèle de l'interaction sociale, les agents jouent différents rôles au sein d'une société. Ce rôle est associé à des *engagements sociaux* ou des *obligations* envers les autres

17. « DP » pour « Dialogue Participant. »

rôles, restreignant le champ d'action et de communication de celui qui l'endosse. Cependant, ces engagements ne sont pas figés et un agent est capable de les faire évoluer ou de les supprimer. Ce fonctionnement offre plus de flexibilité aux comportements des agents et permet de définir les protocoles d'interaction comme des ensembles d'engagements plutôt que comme des automates à états finis, comme c'est le cas dans les approches par états mentaux [Sin00]. Plus formellement, la définition de Singh d'un engagement est donnée en définition 2.9.

Définition 2.9 – Engagement social : « Un engagement c est une relation quadripartite impliquant une proposition p et trois agents x, y et G. $c = \mathrm{C}(x, y, G, p)$ représente un engagement sur la proposition p de la part de x envers y dans un contexte G. On appelle x le *débiteur*, y le *créditeur*, G le *groupe contextuel* et p la condition de décharge de c. » ([Sin99] p. 100)

Une partie de la littérature présente les engagements comme étant contractés par un agent envers lui-même [CL90a]. À l'inverse, un engagement social est contracté *envers une communauté*, capturant les obligations des agents les uns envers les autres. La définition de Singh d'un engagement social inclut son *contexte social*, c'est-à-dire le groupe d'agents au sein duquel l'agent communique et réalise ses actions, perçu comme un agent à part entière.

Singh ajoute à cette définition les opérations réalisables sur les engagements :

La création – instancie un engagement ;

La décharge – satisfait l'engagement. Singh suppose que la décharge est réalisée en même temps que l'action satisfaisant la condition de l'engagement ;

L'annulation – révoque l'engagement ;

L'affranchissement – supprime l'engagement. Différent de la décharge et de l'annulation car n'impliquant pas une réussite ou un échec de l'engagement en question ;

La délégation – transfert le rôle de débiteur à un autre agent du même groupe contextuel ;

L'assignation – transfert le rôle de créditeur à un autre agent du même groupe contextuel.

À l'exception de la décharge, ces opérations correspondent à certains des performatifs donnés par Austin.

Engagements en action et engagements propositionnels

On distingue habituellement les engagements *en action* des engagements *propositionnels* [MCD02]. Un engagement en action correspond à un engagement de la part d'un agent sur l'occurrence d'une action *dans le futur* alors qu'un engagement propositionnel est dirigé *vers le présent* et porte sur l'état du monde. Notons que certains auteurs considèrent que les engagements propositionnels et en action ne sont pas deux catégories d'engagements distinctes mais plutôt que les engagements propositionnels sont une catégorie spécifique d'engagements en action engageant le débiteur à justifier sa proposition dans le cas où l'on

le lui demanderait [Kib06]. La définition de Chaib-Draa *et al.* [CDLBP06] est donnée en définition 2.10.

Définition 2.10 – Engagement en action : « La notion d'engagement est sociale, et ne doit pas être confondue avec son autre pendant basé sur des notions de psychologie. Fondamentalement, les engagements sont contractés envers un partenaire ou un groupe. Il sont exprimés à l'aide de prédicats d'arité 7. Un engagement en action a la forme : C(x, y, α, t, s_x, s_y, `Sta`), où l'agent x (le débiteur) est engagé envers y (le créditeur) sur la réalisation de α au temps t (temps de création), contraint par les sanctions s_x et s_y. » ([CDLBP06] p. 69-70)

La sanction s_x précise ce qu'il se passe si x annule ou viole l'engagement, et s_y précise ce qu'il se passe si y tente de se retirer de l'engagement *après l'avoir accepté*. Une fois l'engagement socialement établi, le rôle de ces sanctions est de décourager une perturbation des règles sociales.

Le paramètre `Sta` correspond à *l'état* de l'engagement. Le modèle d'engagement de Chaib-Draa *et al.* comprend les états suivants :

- *Inactif* (`Ina`) : état par défaut d'un engagement.
- *Créé* (`Crt`) : un engagement est dans cet état s'il a été « socialement établi » au cours de l'interaction.
- *Annulé* (`Cnl`) : un engagement est dans cet état s'il a été révoqué par son débiteur.
- *Affranchi* (`Rel`) : un engagement est dans cet état s'il a été révoqué par son créditeur.
- *Accompli* (`Ful`) : un engagement est dans cet état si le débiteur a satisfait sont contenu.
- *Violé* (`Vio`) : un engagement est dans cet état si le débiteur n'a pas respecté son contenu.
- *Échoué* (`Fal`) : un engagement est dans cet état si la tentative de l'établir socialement a échoué.

Il n'est pas possible d'aller de n'importe quel état vers n'importe quel état et les transitions possibles sont présentées en figure 2.2. La définition des transitions possibles entre les engagements contractés au cours d'une conversation est nécessaire pour la définition d'une politique de conversation basée sur les engagements sociaux. Cependant, la présentation qui en est faite en figure 2.2 ne décrit pas les *actions* véhiculant ces changements d'état. Il est donc nécessaire de trouver une structure capturant le *cycle de vie* des engagements sociaux contractés au cours du dialogue afin d'associer une action dialogique au changement d'état d'un engagement.

La vision de Chaib-Draa des engagements est tournée vers les systèmes multi-agents. Cette orientation permet de décider lors de l'établissement de possibles sanctions en cas de non respect de l'engagement par l'une ou l'autre des parties. Cependant, ce modèle d'engagement semble trop riche pour modéliser une interaction avec au moins un humain comme interlocuteur et le type de sanction qui pourrait s'appliquer dans ce contexte paraît flou et nécessite des investigations supplémentaires. Cela invite, comme nous le verrons plus loin, à une nouvelle définition des engagements sociaux, adaptée à ce contexte.

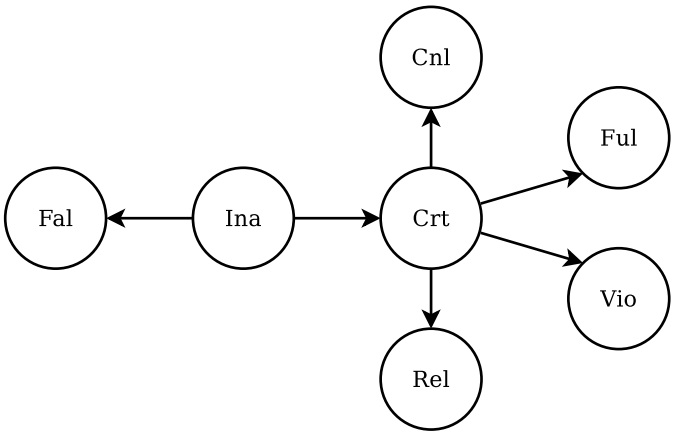

FIGURE 2.2 – Diagramme d'état des engagements du modèle de Chaib-Draa [CDLBP06] (p.71).

Notons qu'il existe, au sein de la communauté des systèmes multi-agents, d'autres modèles utilisant leurs propres version des engagements sociaux afin de construire des protocoles d'interaction [FC04, FVC07, DvdTYS17]. Chaque modèle présente des variations quant au nombre d'états dans lesquels un engagement peut se trouver ainsi que leur cycle de vie. Nous laissons ces approches de côté pour nous concentrer sur les théories impliquant des interlocuteurs humains.

2.3.4 Les jeux de dialogue

Les *systèmes dialectiques*, introduits par Hamblin [Ham70], sont des structures présentant des caractéristiques intéressantes pour la gestion des engagements au cours du dialogue. Un système dialectique est un modèle normatif du dialogue structuré en plusieurs champs : un ensemble de coups dialogiques, un tableau d'engagement [18] pour chaque participant, un ensemble de règles régissant les coups dialogiques, un ensemble de règles régissant les effets des coups dialogiques sur les tableaux d'engagement. Le tableau d'engagement contient tous les engagements contractés par les participants au dialogue. Un système dialectique représente le contexte dialogique et vérifie la cohérence des coups dialogiques dans ce contexte. Cependant il ne traite pas de la production de coups dialogiques.

Maudet [Mau01] propose comme système dialectique les *jeux de dialogue*, des structures *normatives* capturant le lien entre les engagements sociaux et les coups dialogique joués par les interlocuteurs. Nous nous intéressons dans cette section aux jeux de dialogue modélisant le dialogue humain et la génération de dialogue à destination de l'humain. Nous laissons donc de côté l'utilisation des jeux de dialogue pour les systèmes multi-agents et nous concentrons sur la formalisation de conversations humaines.

Maudet retient les propriétés suivantes pour les engagements sociaux dans le cadre d'un dialogue humain-humain [Mau01] :

18. Traduction de« commitment store. »

— Les engagements sont *sociaux*. Ils sont pris envers d'autres membres d'une communauté.

— Les engagements sont *publics* et accessibles à tous les membres de la communauté. On considère alors qu'ils sont stockés dans une structure de données appelée *tableau d'engagement*. Dans ce modèle, cette structure est vue comme strictement partagée et accessible à tous les participants au dialogue.

— Les engagements sont *propositionnels* ou *en action*.

— Les engagements peuvent être *directs* ou *conditionnels*. Un engagement conditionnel tient si sa condition est remplie. Le contenu de l'engagement peut alors s'exprimer sous la forme d'une règle traduisant par exemple « si tu fais α, alors je fais β ».

— Les engagements sont *dialogiques* ou *extra-dialogiques*. Un engagement dialogique est contracté dans le contexte local du dialogue en cours. Par exemple les conventions du dialogue engagent un interlocuteur à répondre à une question qui lui est posée. Un engagement extra-dialogique est contracté dans le contexte de la tâche sous-jacente au dialogue.

— Les engagements sont *explicites* ou *implicites*. Un engagement contracté explicitement l'est par le biais d'un acte communicatif, comme un acte de dialogue. Un engagement implicite résulte de l'application d'une règle ou d'un protocole partagé entre les interlocuteurs.

— Les engagements sont *datés*, c'est-à-dire pris à un moment donné.

— Les engagements peuvent être *ordonnés*. Un engagement en action peut être prioritaire sur un autre. Par exemple, un engagement en action peut nécessiter la satisfaction d'un autre engagement en action pour être réalisable.

La vision de Maudet diffère donc légèrement de celle de Hamblin du fait qu'elle considère un unique tableau d'engagement partagé par tous les participants (correspondant au quatrième modèle d'état d'information de Ginzburg) au lieu d'un tableau d'engagement par participant.

Pour sa formalisation des jeux de dialogue, Maudet se base sur les travaux de Singh [Sin00] pour introduire sa propre définition des engagements sociaux, $\mathrm{C}(x, p)$, dénotant l'engagement propositionnel de x envers y sur la proposition p. Les engagements en action sont rattachés au jeu j dans lequel ils sont contractés. Différentes formes d'engagement en action sont possibles :

— $\mathrm{C}_j(x, \alpha)$: x s'engage sur l'occurrence de l'action α ;

— $\mathrm{C}_j(x, \alpha_1|\alpha_2)$: x s'engage sur l'occurrence de α_1 ou de α_2 ;

— $\mathrm{C}_j(x, \neg\alpha)$: x sur le fait que α n'ait pas lieu ;

— $\mathrm{C}_j(x, \alpha \Rightarrow \beta)$: x s'engage à ce que β soit réalisé sur le talbeau de conversation si α a lieu ;

— $\mathrm{C}_j(x, \alpha \stackrel{*}{\Rightarrow} \beta)$: x s'engage à ce que β soit réalisé sur le tableau de conversation *à chaque fois* que α a lieu.

Ces deux derniers engagement sont *conditionnels* et permettent de modéliser la conséquence d'une action sur le tableau de conversation. Un type particulier d'engagement en action est *l'engagement conjoint* $\mathrm{C}(\{x, y\}, j)$. Celui-ci engage les interlocuteurs sur le jeu j.

Les jeux définis par Maudet sont des *structures conventionnelles* associant des actes de dialogue à des engagements sociaux selon des règles partagées entre les interlocuteurs. Il distingue deux catégories de jeux : les *jeux de dialogue* et les *jeux de communication*. Les jeux de dialogue sont des activités *temporaires*, réalisées *conjointement* pendant le dialogue dans un *but spécifique*. Ils peuvent être *ouverts*, *fermés* ou *proposés* et doivent être établis par les participants. Les jeux de communication sont *toujours* activés, sont dédiés à la gestion de l'interaction et capturent les critères communicatifs supposés valides pour n'importe quel type de dialogue.

Un jeu de dialogue s'exprime de plus en termes de conditions sur le tableau de conversation. Il est défini selon trois conditions : les *conditions d'entrées*, nécessaires pour entrer dans le jeu de dialogue, les *conditions de succès*, définissant le succès du jeu, les *conditions d'échec*, définissant l'échec du jeu. Il est à noter que Maudet simplifie les conditions d'échec d'un jeu en les associant à l'énonciation d'un coup dialogique vide. Un jeu est dit *collaboratif* si les conditions de succès sont les mêmes pour les deux interlocuteurs.

Les jeux de dialogue contiennent de plus des *règles dialogiques* spécifiant les engagements conversationnels contractés lorsque les interlocuteurs entrent dans un jeu. Ces engagement contiennent notamment les actes de dialogue attendus ou interdits au sein de ce jeu.

Jeux de communication Les jeux de communication sont écrits à l'aide d'engagements conditionnels persistants : $\mathrm{C}_j(-, \alpha \overset{*}{\Rightarrow} \beta)$. Ils sont donc activés en permanence et n'ont donc ni condition d'entrée, ni condition de sortie.

L'intégralité de ce modèle est défini à l'aide d'engagements, même les règles dialogiques définies au sein des jeux de dialogue.

Le statut des jeux de dialogue est géré à l'aide d'un jeu de communication de *contextualisation*, utilisant des méta-actes de dialogue pour gérer la structure du dialogue. Les fonctions associées sont présentées dans le tableau 2.4. Elles se situent dans la dimension *Dialogue Structure* de la taxonomie DIT++, au niveau de contextualisation.

Dimension	Niveau	Fonctions
Dialogue Structure	Contextualisation	*prop.in ref.in acc.in cont prop.out ref.out acc.out*

TABLEAU 2.4 – Fonctions des actes de dialogue des contextualisation [Dub14].

La spécification du jeu de communication de contextualisation est donnée en tableau 2.5. Le coup *prop.in* correspond à la proposition par x d'entrer dans le jeu j. Il crée l'engagement pour le partenaire d'accepter ($acc.in$) ou de refuser ($ref.in$) de rentrer dans j et fait passer j dans l'état *proposé*. L'entrée dans le jeu le fait passer dans l'état *ouvert* et crée l'engagement conjoint $\mathrm{C}(\{x, y\}, j)$ des deux interlocuteurs sur leur participation

à j. Tant que le jeu est *ouvert*, il est possible de continuer (*cont*) de jouer des coups dialogiques dans celui-ci. On suppose alors que n'importe quel coup dialogique attendu (c'est-à-dire satisfaisant un engagement) dans le contexte du jeu joue en même temps un coup de poursuite (*cont*) dans le jeu de contextualisation. Enfin, il est possible de proposer (*prop.out*) de quitter le jeu, cette proposition pouvant être acceptée (*acc.out*) ou refusée ($ref.out$). Accepter de quitter le jeu fait passer celui-ci dans l'état *fermé*. Les différentes transitions entre les états ainsi que les coups dialogiques associés sont représentées en figure 2.3.

Coup	Effet
$prop.in(x, j)$	$create(x, \mathrm{C}(y, acc.in(y, j) \mid ref.in(y, j)))$
$acc.in(x, j)$	$create(x, \mathrm{C}(\{y, x\}, j))$
	$create(x, \mathrm{C}(y, cont(y, j)))$
$cont(x, j)$	$create(x, \mathrm{C}(y, cont(y, j) \mid prop.out(x, j)))$
$prop.out(x, j)$	$create(x, acc.out(y, j) \mid ref.out(y, j))$
$acc.out(x, j)$	$create(x, \mathrm{C}(\{y, x\}, j))$

TABLEAU 2.5 – Jeu de communication de contextualisation [MCD02].

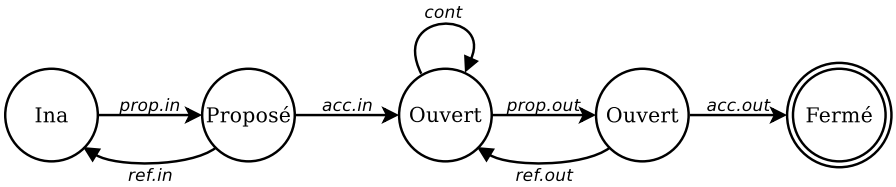

FIGURE 2.3 – Diagramme des états d'un jeu de communication de contextualisation [Mau01].

Les *jeux de dialogue* et les *jeux de communication* sont des motifs normatifs capturant les engagements contractés par les interlocuteurs au cours du dialogue. Ceux-ci apportent des contraintes structurelles au dialogue, en régulant les coups dialogiques attendus et réalisables par les interlocuteurs. Ces contraintes diminuent la complexité combinatoire de l'interprétation ou de la génération d'un coup dialogique via la définition d'engagement sur des actions dialogiques.

Cette vision présente l'interaction comme subordonnée à des contraintes sociales et basée sur des conventions et des règles. Les jeux de dialogue donnent la possibilité à un agent d'avoir un comportement social réactif pertinent avec les normes régissant l'interaction. Cependant, les règles définies par les jeux ne suffisent pas à définir les motivations qui vont pousser l'agent à prendre la décision d'un comportement dialogique.

2.4 Les jeux de dialogue comme approche mixte

Nous développons dans cette section une vision des jeux de dialogue comme des *approches mixtes*, réconciliant intentionnel et conventionnel en une seule structure. Plus

particulièrement, nous présentons le gestionnaire de dialogue DOGMA développé par Dubuisson Duplessis [DCK+13, Dub14, DPCK17] et le formalisme qui lui est associé. La section 2.4.1 présente le modèle de dialogue utilisé par DOGMA et notamment son modèle d'engagement social et de tableau de conversation. La section 2.4.2 donne le modèle de jeux de dialogue développé pour DOGMA. La section 2.4.3 montre que les approches intentionnelles et conventionnelles ne sont pas mutuellement exclusives et que les jeux de dialogues sont des structures permettant de les réconcilier.

2.4.1 Modèle de dialogue de DOGMA

À l'instar de Maudet (cf. section 2.3.4), Dubuisson Duplessis distingue, dans son modèle *standard*[19], les engagements *propositionnels* des engagements *en action*. Ces derniers sont séparés en deux catégories : les engagement *en action dialogiques* et les engagements *en action extra-dialogiques*.

Engagement propositionnel

La définition d'un engagement propositionnel prend la forme d'un quintuplet :

Définition 2.11 – Engagement propositionnel : « Un engagement propositionnel capture le fait qu'un interlocuteur s'engage au présent sur une proposition envers un autre interlocuteur. Un tel engagement prends la forme C(x, y, p, t, s) signifiant que l'engagement "x est engagé envers y sur la proposition p" est dans l'état s depuis le temps t. » ([Dub14] p.146)

Les engagements en action ne peuvent être que dans deux états, présentés en figure 2.4 : initialement, un engagement propositionnel est *inactif* (**Ina**), il peut être créé, passant dans l'état *créé* (**Crt**), dit *actif*. Un engagement créé peut être annulé par un participant au dialogue, retournant alors à l'état *inactif*.

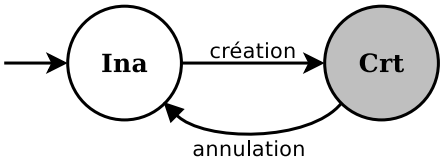

FIGURE 2.4 – Diagramme des états d'un engagement propositionnel [Dub14].

Engagement en action

Un engagement en action est défini de la manière suivante :

Définition 2.12 – Engagement en action : « Un engagement en action capture le fait qu'un interlocuteur s'engage au présent à ce qu'une action survienne dans le futur. » ([Dub14] p.146)

19. Celui-ci exclut les engagements conjoints sur les jeux de dialogue, un cas particulier qui ne sera pas traité ici.

Un engagement en action peut se trouver dans différents états, comme présenté en figure 2.5. Un engagement en action est initialement inactif (**Ina**). Un interlocuteur peut

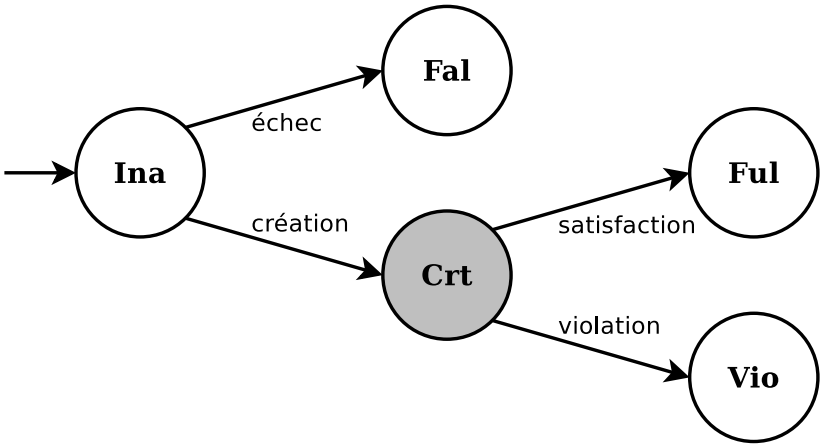

FIGURE 2.5 – Diagramme des états d'un engagement en action [Dub14].

alors tenter de le créer. Cette tentative de création peut échouer (l'amenant dans l'état **Fal**) ou réussir (l'amenant dans l'état **Crt**). Un engagement dans l'état **Crt** est dit *actif*. Dans cet état, l'engagement peut être *violé* (l'amenant dans l'état **Vio**), si les conditions de satisfaction de l'action associée ne peuvent plus être remplies. Autrement, il peut être *satisfait* (l'amenant dans l'état **Ful**), si l'action associée est effectuée.

Même si la distinction est faite entre les engagements en action dialogique et les engagements en action extra-dialogique, ces deux types d'engagements sont aussi définis à l'aide d'un quintuplet. Les définitions des engagements en action extra-dialogique et dialogique sont donnés en définition 2.13 et 2.14 respectivement.

Définition 2.13 – Engagement en action extra-dialogique : « Un engagement en action *extra-dialogique* prend la forme C(x, y, α, t, s) signifiant que l'engagement "x est engagé envers y sur la réalisation de l'action α" est d'ans l'état s depuis un temps t. » ([Dub14] p.147)

Définition 2.14 – Engagement en action dialogique : « Les engagements en action *dialogiques* sont au cœur de la spécification des jeux de dialogue et de communication. Ceux-ci prennent la même forme que les engagements en action *extra-dialogiques* à l'exception du fait qu'ils sont contextualisés dans un jeu j noté en indice : $C_j(x, y, \alpha, t, s)$. » ([Dub14] p.147)

Le terme « engagement » désigne indifféremment les engagements propositionnels, les engagements en action dialogique et les engagements en action extra-dialogique. Les engagements retenus pour ce modèle se restreignent à un petit nombre d'états. Cependant, il est envisageable de l'étendre à d'autres états de manière à enrichir les descriptions qui peuvent être faites des engagements sociaux. Le modèle de Chaib-Draa [CDLBP06] donné en figure 2.2 présente une extension possible de ce modèle.

Tableau de conversation

Le tableau de conversation est une structure enregistrant *l'état du dialogue* à un instant donné. Il est considéré comme étant *strictement partagé* entre les participants au dialogue et fonctionne en temps discret. On note T_i pour symboliser le tableau de conversation à l'instant i (que l'on considère comme le temps courant).

Le tableau de conversation contient le *tableau d'engagement*. Celui-ci est un ensemble d'engagements *partiellement ordonné*, c'est-à-dire qu'il est possible que la satisfaction d'un engagement soit conditionnée par la satisfaction d'un autre engagement. Trois fonctionnalités relatives aux engagements existent sur le tableau de conversation, présentées dans le tableau 2.6. Il est possible de vérifier *l'appartenance* ou la *non-appartenance* d'un engagement au tableau de conversation. Il est aussi possible de vérifier si un engagement est *prioritaire* sur un autre, le tableau étant partiellement ordonné. La priorisation d'engagement est un phénomène qui peut être engendré par des jeux de dialogue, ceux-ci spécifiant que certains engagements en action doivent être réalisés avant d'autres.

Nom	Fonctionnalité	Préconditions	Résultat
Appartenance	$\mathrm{T}_i \ni c$ c : engagement		vrai si $c \in \mathrm{T}_i$, faux sinon
Non-appartenance	$\mathrm{T}_i \not\ni c$ c : engagement		équivalent à $\neg(\mathrm{T}_i \ni c)$
Priorité	$\mathrm{T}_i \ni c_1 \prec c_2$ c_1, c_2 : engagement	$\mathrm{T}_i \ni c_1$, $\mathrm{T}_i \ni c_2$	vrai si c_1 est prioritaire sur c_2, faux sinon

Tableau 2.6 – Fonctionnalités du tableau de conversation [Dub14].

Contraintes des engagements sur le comportement communicatif Les engagements en action dialogique réalisés par les interlocuteurs contraignent leur *comportement communicatif* en désignant des *règles de production* des actions dialogiques. Cela permet de définir les événements dialogiques qui :

— sont *attendus*, c'est-à-dire qu'il existe dans le tableau de conversation un engagement sur la réalisation de cette action ;

— *violent* une règle, c'est-à-dire qu'il existe dans le tableau de conversation un engagement sur la réalisation de la négation de cette action ;

— sont *réguliers*, c'est-à-dire qu'il ne viole aucun engagement du tableau de conversation ou, s'il viole un engagement c_1 du tableau de conversation, il est attendu par un engagement c_2 prioritaire sur c_1 ;

— sont *interdits*, c'est-à-dire qu'ils ne sont pas réguliers.

2.4.2 Représentation des jeux dans DOGMA

Modèle de jeu de dialogue

Dans DOGMA, les jeux de dialogue s'écrivent sous la forme type(sujet), où *type* appartient à l'ensemble des jeux de dialogue existants et *sujet* correspond au but du jeu dans le langage d'expression du sujet du jeu [Dub14]. Par exemple, requête(α) réfère au jeu de requête dont le but est l'action α. La définition des jeux de dialogue est la suivante :

Définition 2.15 – Jeu de dialogue : « Un jeu de dialogue est défini en terme d'engagements sociaux. C'est un quintuplet caractérisé pour *l'initiateur* et le *partenaire* par des :

conditions d'entrée : celles-ci spécifient l'état dans lequel doit se trouver le tableau de conversation permettant l'entrée dans le jeu de dialogue. Cet état réfère à des engagements *extra-dialogiques*. Par exemple, les conditions d'entrée du jeu de requête d'action spécifient que le partenaire ne doit pas être déjà engagé sur l'action demandée ;

conditions de sortie subdivisées en :

- **conditions de succès** : celles-ci spécifient l'état du tableau de conversation qui correspond au succès. Cet état réfère à des engagements extra-dialogiques ;
- **conditions d'échec** : celles-ci spécifient l'état du tableau de conversation correspondant à l'échec du jeu. Cet état réfère à des engagements extra-dialogiques.

règles : elles spécifient les enchaînement d'actes attendus ou interdits dans le contexte du jeu en termes d'engagements en action *dialogiques* ;

effets : il s'agit de règles particulières qui spécifient les *effets contextualisés des actes de dialogue* en termes de production d'engagements extra-dialogiques.

» ([Dub14] p.158)

Un exemple de jeu de dialogue est développé dans la suite de ce manuscrit dans le tableau 2.10.

Modèle de jeu de communication Les jeux de communication utilisés dans ce modèle sont identiques à ceux de Maudet présentés en section 2.3.4. Ils sont spécifiés sous la forme d'engagements en actions dialogiques persistants : $C_j(-,\alpha \overset{*}{\Rightarrow} \beta,\mathbf{Crt})$. Nous ne considérons ici que deux jeux de communication : le jeu de contextualisation (voir tableau 2.5 et figure 2.3) et le jeu d'évaluation (tableau 2.8).

Jeux extraits d'un corpus humain-humain

Les jeux de dialogue intégrés à DOGMA sont basés sur une réalité empirique. En effet, ils sont issus de l'extraction de motifs d'interaction présents dans le corpus de dialogues humain-humain du projet COGNI-CISMEF.

Le corpus COGNI-CISMEF Le corpus COGNI-CISMEF [Loi08], qui a servi à l'élaboration de DOGMA, est constitué de dialogues sur une tâche de RI médicale. Chaque dialogue a lieu entre un utilisateur ayant un besoin d'information spécifique et un spécialiste du moteur de recherche de CISMEF, une plate-forme d'indexation médicale.

Actes de dialogue Les actes de dialogue utilisés dans ce modèle sont *multidimensionnels* et *multiniveaux*, et sont issus de la taxonomie DIT++ (cf. section 2.1.4). L'analyse faite pour établir ce modèle s'est restreinte à trois dimensions de DIT++ : la dimension *Task*, limitée au niveau *standard*, la dimension *Auto-feedback*, limitée à l'acte *ExecNegativeAutoFB* du niveau *exécution*, et la dimension *Dialogue Structure*, limitée au niveau *Contextualisation*. Les actes de dialogue des deux premières dimensions sont donnés dans le tableau 2.7. Les actes de dialogue de la *Dialogue Structure* sont donnés dans le tableau 2.4.

Dimension	Niveau	Fonctions
Task	Standard	*Inform*, *Answer*, *Disconfirm*, *Confirm*, *Agreement*, *Disagreement*, *Correction*, *AcceptCorrection*, *DeclineCorrection*, *PropositionalQuestion*, *CheckQuestion*, *PosiCheck*, *NegaCheck*, *SetQuestion*, *ChoiceQuestion*, *Offer*, *AcceptOffer*, *DeclineOffer*, *Request*, *AcceptRequest*, *DeclineRequest*, *Suggestion*, *AcceptSuggestion*, *DeclineSuggestion*
Auto-feedback	Exécution	*ExecNegativeAutoFB*

TABLEAU 2.7 – Fonction des actes de dialogue considérés dans DOGMA [Dub14].

Les jeux de dialogue et de communication présents dans DOGMA utilisent ces fonctions pour capturer les engagements que les interlocuteurs contractent pendant le dialogue. Ces engagements peuvent porter sur des propositions ou des actions dialogiques ou extra-dialogiques.

Jeux de communication DOGMA intègre deux jeux de communication : le jeu de contextualisation et le jeu d'évaluation. Le jeu de contextualisation est celui présenté en section 2.3.4.

Les effets d'un jeu de communication sont exprimés à l'aide d'un engagement sur une conditionnelle persistante, formellement : $C_j(-, \alpha \overset{*}{\Rightarrow} \beta, \mathbf{Crt})$, signifiant que chaque interlocuteur est engagé sur les règles du jeu de communication j de manière persistante.

Le *jeu d'évaluation* capture les effets des actes de transfert d'information supposés valides dans toute conversation. Il est noté « *ev* » dans les engagements dialogiques. L'objectif de ce jeu est double : premièrement, il permet de définir les *effets directs* de l'occurrence de certains actes de dialogue sur le tableau de conversation. Ces effets sont présentés dans le tableau 2.8.

Les effets des actes *Confirm* et *Disconfirm* dépendent du jeu de dialogue dans lequel ils prennent part. Ceux-ci ne sont donc pas définis dans le jeu d'évaluation.

α	β
f(x, p)	C(x, p, **Crt**)
disagreement(x, p)	C(x, $\neg p$, **Crt**)

f $\in$ {inform, answer, agreement}

TABLEAU 2.8 – Jeu de communication d'évaluation : effets directs(adapté de [Dub14]).

Le second objectif du jeu d'évaluation est de modéliser les motifs dialogiques d'accord et de correction, permettant aux interlocuteurs de s'accorder sur l'état du monde. Ces motifs sont présentés dans le tableau 2.9. Ils décrivent le fait que l'émission d'un acte informatif (*Inform* ou *Answer*) engage son interlocuteur à exprimer son accord (*Agreement*), désaccord (*Disagreement*) ou à proposer une correction (*Correction*).

Notons ici l'utilisation de la relation de correction, **correction**(p, s), dépendante du domaine et qui est vraie lorsque s est une correction de p. Celle-ci introduit une contrainte sémantique sur le contenu de l'acte de correction.

α	β
f(x, p)	C_{ev}(y, agreement(y, p) | disagreement(y, p) | correction(y, p, s), **Crt**) avec **correction**(p, s)

f $\in$ {inform, answer}

TABLEAU 2.9 – Jeu de communication d'évaluation [Dub14].

Jeux de dialogue Lors de l'analyse du corpus COGNI-CISMEF, trois catégories de jeux de dialogue ont été extraites [Dub14].

Vérification d'une proposition – Quatre motifs de vérification on été identifiés dans le corpus : question oui/non, vérification, vérification positive et vérification négative Ceux-ci servent à vérifier la valeur de vérité d'une proposition. Les différentes variantes de ces jeux viennent des à priori que peut avoir le locuteur sur le contenu propositionnel du jeu ;

Interrogation ouverte et interrogation à choix multiples – Ces deux motifs permettent de poser une question ouverte ou une question à choix multiples à son partenaire ;

Discussion d'action – Trois motifs portant sur des actions ont été identifiés dans le corpus : deux directifs la suggestion et la requête, et un promissif, l'offre. Ils servent à proposer ou a demander à son partenaire de réaliser une action.

Pour illustrer l'écriture des jeux de dialogue, le jeu d'offre est présent dans le tableau 2.10. Il permet à son initiateur de proposer de réaliser une action. Ses conditions d'entrée indiquent que le locuteur ne doit pas être engagé sur l'occurrence de l'action ni sur sa non-occurrence. Ses conditions de succès et d'échec sont les mêmes pour les deux interlocuteurs : dans le premier cas l'initiateur s'est engagé à réaliser l'action, dans le

second cas, le partenaire a refusé que l'initiateur s'engage sur l'action. Les règles stipulent que l'initiateur est engagé à jouer un acte *offer*. Cet acte a pour conséquence d'engager le partenaire sur l'acceptation de cette offre, correspondant à un acte *acceptOffer*, ou le refus de cette offre, correspondant à un acte *declineOffer*. Les effets associent à l'acte *acceptOffer* le fait que l'initiateur est engagé sur la réalisation de cette action, et à l'acte *declineOffer* le fait que l'engagement de l'initiateur sur cette action a échoué.

j=**offre**(α)		
	Initiateur (x)	Partenaire (y)
Entrée	C(x, α, **Ina**) et C(x, $\neg\alpha$, **Ina**)	
Succès	C(x, α, **Crt**)	C(x, α, **Crt**)
Échec	C(x, α, **Fal**)	C(x, α, **Fal**)
Règles	offer(x, α)	offer(x, α) $\Rightarrow$ C$_j$(y, acceptOffer(y, α) | declineOffer(y, α), **Crt**)
Effets	acceptOffer(y, α) $\Rightarrow$ C(x, α, **Crt**) declineOffer(y, α) $\Rightarrow$ C(x, α, **Fal**)	

TABLEAU 2.10 – Jeu de dialogue d'offre [Dub14].

2.4.3 Réconcilier intentionnel et conventionnel

Les approches intentionnelles et conventionnelles sont souvent présentées comme opposées et mutuellement exclusives, se focalisant pour la première sur les états mentaux des interlocuteurs et pour la seconde sur une structure normalisée du dialogue. Cependant, cette séparation n'est pas aussi franche qu'il peut le sembler à priori. En effet, il semble dans une certaine mesure possible de tirer parti du meilleur des deux approches et de construire une architecture d'agent mixte réactive/délibérative en considérant qu'elles sont complémentaires [Mau01, Dub14]. Cela rejoint la vision de Clark [Cla96] qui considère le dialogue comme une action conjointe. Une action conjointe est *coordonnée* et composée par des *actions participatives*. Cette approche est aussi celle avancée par Traum [TA92], qui présente les actes de dialogue comme ayant des effets à la fois sur les états mentaux des interlocuteurs et sur l'état du dialogue lui-même. Cela traduit le fait qu'au cours du dialogue, différents types de processus sont mis en jeu, certains délibératifs, liés aux intentions et désirs des interlocuteurs, et d'autres réactifs traduisant les structures conventionnelles de certaines phases d'interaction.

Il est effectivement peu réaliste de penser que l'approche intentionnelle seule suffit, c'est-à-dire que nous planifions toutes les interventions que nous faisons au cours d'un dialogue. Certains échanges sont en effet stéréotypés et ritualisés, comme les salutations en début et fin de dialogue ou les remerciements et la reconnaissance d'une intention sous-jacente à ces motifs est superflue [Mau02]. De plus, la mise en œuvre des approches intentionnelles est techniquement difficile du fait qu'elle peut rapidement mener à une explosion combinatoire même si l'utilisation d'un *centre d'intérêt* définissant un contexte local permet de réduire ce problème. Dans le même sens, Clark [Cla96] présente le dialogue comme une activité

opportuniste le rendant intrinsèquement imprévisible. De plus Pulman [Pul98] souligne que certaines séquences d'actes ne peuvent pas être planifiées. L'idée même de la possibilité de planifier un dialogue a aussi été remise en cause par Suchman [Suc07], pour qui, vu le contexte dynamique et évolutif du dialogue, les plans du dialogue ne sont pas des schémas établis d'avance mais des *reconstructions* réalisées à posteriori.

L'intention seule n'est donc pas pleinement satisfaisante pour modéliser le dialogue, et ne permet pas de représenter certains phénomènes impossibles à planifier, amenant à l'introduction d'une structure normative de l'interaction. C'est ce qu'apportent les jeux de dialogue, en permettant la régulation nécessaire à la coordination des participants tout en préservant la flexibilité du dialogue et sa nature opportuniste. En effet, les jeux de dialogue traduisent les conventions de l'interaction langagière tout en véhiculant l'engagement d'un locuteur envers son interlocuteur. Comme le formule Maudet [Mau02] (p. 94) : « l'ancrage avec le niveau intentionnel est réalisé à travers la notion de but(s) du jeu, qui représente(nt) les motivations qui poussent les interlocuteurs à s'engager dans un jeu donné, afin de permettre la manipulation de ces structures lors du processus de délibération. »

2.5 Conclusion

Dans ce chapitre, nous avons présenté quelques une des méthodes de dialogue humain-machine.

Nous avons dans un premier temps présenté le dialogue comme un *projet conjoint opportuniste* puis discuté de la théorie des *actes de langage* d'Austin, par la suite étendue dans la théorie des *actes de dialogue*. Dans un second temps, nous avons développé les approches présentant le dialogue comme une structure *intentionnelle*, c'est-à-dire résultant de la mise en œuvre d'un plan par les interlocuteurs. Dans un troisième temps, nous avons présenté les approches *conventionnelles*, c'est-à-dire considérant que le dialogue est une activité contrainte imposant aux interlocuteurs de respecter certains motifs d'interaction conventionnellement établis entre eux. Ces approches se manifestent notamment sous la forme de *jeux de dialogue*, décrivant les règles régissant le dialogue à l'aide d'engagements sociaux et de jeux de dialogue. Enfin, nous avons montré que les jeux de dialogue constituent une structure intéressante pour la mise en œuvre d'une approche mixte réactive/délibérative, donnant la possibilité de concevoir des agents réconciliant approches intentionnelles et conventionnelles. Nous avons notamment présenté DOGMA, un gestionnaire de dialogue basé sur la théorie des jeux de dialogue.

L'utilisation des jeux de dialogue pour la conception d'un gestionnaire de dialogue présente différents avantages [Mau01, Dub14] :

— les jeux sont basés sur des *réalités empiriques* et peuvent être spécifiés à partir de motifs dialogiques. C'est notamment le cas avec ceux implémentés dans DOGMA, qui sont issus de l'analyse du corpus d'interaction humain-humain du projet COGNICISMEF ;

— les jeux sont des structures *prédictives*. La vision du dialogue comme un enchaînement de motifs dialogiques pré-défini accroît les capacités *interprétative* et *générative* de l'agent en l'aidant à contextualiser les actes de dialogue issus de son interlocuteur et

à sélectionner les réponses pertinentes à lui apporter, renforçant son comportement collaboratif ;

— les jeux ont été décrit et formalisés rigoureusement comme nous l'avons monté en section 2.4 ;

— les jeux permettent la *gestion du cycle de vie* des engagements sociaux. L'utilisation des engagements sociaux pour représenter les obligations contractées par les agents les uns par rapport aux autres est accompagnée de la nécessité de gérer l'évolution des états de ces engagements [CDLBP06]. Les jeux de dialogue apportent un élément de réponse en permettant de gérer une partie du cycle de vie des engagements.

Ces arguments nous amènent à considérer DOGMA comme un outil adapté au développement d'un agent collaboratif interagissant avec un utilisateur pour l'aider à réaliser une tâche.

La vision des jeux en tant que structures mixtes réactive/délibérative permet d'articuler les buts des interlocuteurs (qui peuvent être conflictuels) et les attentes des interlocuteurs les uns par rapport aux autres en termes de participation à l'interaction (par le respect des normes régissant le déroulement du dialogue). Il semble cependant que le lien interaction-tâche n'est pas clairement établi dans les jeux de dialogue. En effet, il n'existe à l'heure actuelle aucune structure permettant de lier explicitement la génération d'un acte de dialogue par un agent dans le cadre d'un jeu de dialogue et l'intention sous-jacente de l'agent. La relation entre les processus intentionnels et conventionnels reste donc à préciser et nécessite la définition d'un formalisme tenant compte à la fois des contraintes dialogiques induites par les jeux de dialogue et des processus délibératifs de l'agent.

À cela s'ajoute une capacité réduite des jeux à gérer le cycle de vie des engagements sociaux. En effet, les jeux de dialogue ne tiennent compte que du contexte dialogique pour la mise à jour des engagements sociaux et ignorent le contexte de la tâche sous-jacente au dialogue. En en outre, ils ne prévoient aucun mécanisme pour la gestion des effets que peut avoir un acte de dialogue sur les engagements autres que ceux relatifs au motif dialogique lui-même.

Deuxième partie

Contribution

Chapitre 3

Modèle formel pour la collaboration humain-machine

« – Shakespeare, dit-il à Maren [. . .]. Il profère des citations. – La machine puise dans sa banque de mots, de phrases, de citations. Qu'attendais-tu ? Un sonnet qu'elle aurait elle-même composé ? Elle peut choisir, jamais inventer. »

Philip K. Dick, Le zappeur de mondes

Sommaire

Ce chapitre présente le volet théorique de notre contribution. Nous y développons un modèle d'agent collaboratif conçu pour assister un utilisateur dans la résolution d'un problème confus ou épineux, c'est-à-dire caractérisé par une complexité technique et dont la définition peut évoluer en cours de résolution.

Nous nous rattachons au concept de *système compagnon* [BW17], des systèmes intelligents qui ont pour but de s'adapter à un utilisateur en présentant des compétences comme l'autonomie et la collaboration. Leur but est de supprimer les obstacles qui entravent la bonne utilisation de systèmes compliqués par l'utilisateur en lui rendant accessibles leurs fonctionnalités avancées.

Nous utilisons les motifs dialogiques pour proposer une approche formelle de la collaboration humain-machine, menée par l'interaction [LDC+17]. Un point-clé de cette approche est qu'elle repose sur des comportements conventionnellement attendus et localement cohérents pour permettre au système de réaliser des actions pertinentes pour faire avancer la tâche et pour contextualiser les actions de l'utilisateur.

La section 3.1 donne notre représentation de l'interaction humain-machine. La section 3.2 décrit les raisonnements de notre système réalisés à partir de la représentation de l'interaction. La section 3.3 présente les spécificités de la gestion de l'interaction avec l'utilisateur. Enfin, nous discutons en section 3.4 de l'intérêt de notre représentation et des processus décisionnels qui y sont liés.

3.1 Représentation de l'interaction humain-machine

Nous donnons dans cette section le formalisme que nous utilisons pour représenter l'interaction humain-machine. Celui-ci tire parti de l'aspect formel des engagements sociaux et des jeux de dialogue (cf. sections 2.3.3 et 2.3.4) pour faire le lien entre les actions réalisées par l'utilisateur et la structure conventionnelle de l'interaction. Nous introduisons le concept *d'état*, une représentation des interactions conventionnellement attendues et localement cohérentes permettant de faire avancer une partie de la tâche. Nous donnons de plus une représentation des états sous forme de *tables d'état*, organisant l'écriture des états pour les rendre plus lisibles.

La section 3.1.1 donne le cadre de nos travaux et la structure de la tâche collaborative que nous essayons de représenter. La section 3.1.2 présente notre formalisation des interactions permettant le déroulement de la tâche. La section 3.1.3 développe le concept de table d'état en précisant certains aspects de celles-ci.

3.1.1 Structure d'une tâche collaborative

Nous donnons maintenant le cadre interactif pour lequel notre système est conçu, ainsi que nos hypothèses de travail sur le type d'interaction humain-machine concerné et la tâche sous-jacente à cette interaction.

Type d'interaction

Nous nous focalisons sur une interaction entre un humain et une machine. Nous excluons donc la possibilité d'avoir plusieurs interlocuteurs humains au sein de la même interaction. Cette hypothèse, bien que réductrice, permet de simplifier grandement la représentation de l'interaction et la gestion de la communication. En effet, le fait de conserver une interaction avec seulement deux interlocuteurs restreint la complexité d'interprétation d'un acte de langage car celui-ci n'a qu'un seul destinataire et ne sera donc interprété que d'une seule manière. L'introduction d'autres interlocuteurs apporterait la possibilité d'avoir des interprétations multiples d'un même acte de langage, ce qui complexifierait le maintien d'une représentation cohérente de l'interaction.

Suchman [Suc07] explique que « l'interaction entre les humains et les machines requiert principalement le même travail interprétatif que celui qui caractérise l'interaction entre les personnes, mais avec des ressources foncièrement différentes. En particulier, les humains utilisent une palette riche de ressources linguistiques, non verbales, et inférentielles pour rendre intelligibles des actions et des événements, pour donner du sens à leurs propres actions et pour gérer les problèmes de compréhension qui arrivent inévitablement. En contraste, les machines actuelles on recours à une palette fixe d'entrées sensorielles, associées à un ensemble pré-déterminés d'états internes et de réponses. » (p. 178-179).

De manière à réduire ce fossé entre l'humain et la machine, nous utilisons une modélisation conventionnelle du dialogue humain qui nous semble être un compromis idéal entre les ressources utilisées par les humains et celles accessibles à la machine. À ce titre, nous nous basons sur le modèle de Dogma (cf. section 2.4.1), qui donne une représentation formelle et cohérente du dialogue tout en étant basé et validé sur des exemples de dialogues humains. Plus spécifiquement, ce modèle de dialogue décrit, à l'aide d'engagements sociaux, de jeux de dialogue et de jeux de communication, la structure conventionnellement attendue des échanges ayant lieu au cours d'un dialogue.

Nous prenons donc le parti de représenter les interactions, qu'elles soient verbales ou non verbales, à l'aide d'un modèle formel de dialogue. Cela implique que toute action réalisée au cours de l'interaction est véhiculée par un acte de dialogue, que cette action soit dialogique ou extra-dialogique. Par exemple, une question posée verbalement (« Tu viens au sport ce midi ? ») peut être répondue de manière non verbale (un hochement de tête). Dans notre cas, cette réponse sera tout de même représentée à l'aide d'un acte de dialogue. Nous faisons ici l'hypothèse qu'il existe un outil de traitement automatique de la langue permettant d'extraire des actes de dialogue à partir d'énoncés en langue naturelle et des comportements interactifs extra-dialogiques associés. Nous utilisons une approche fonctionnelle du langage où chaque acte de dialogue est représenté par une fonction et un contenu sémantique.

Selon la classification de Jucker, donnée dans le tableau 2.1, nous nous situons dans une conversation *orientée structure*, ce qui correspond à une conversation contrainte, ayant lieu au sein d'une activité précise, en lien avec une tâche particulière ou devant mener à certains résultats.

Notre objectif est de fournir un soutien à l'utilisateur et de l'aider à résoudre un problème difficile. Notre système a donc pour but de se mettre au service de celui-ci et de

l'assister tout au long du processus de résolution. L'interaction entre l'utilisateur humain et la machine sera donc *asymétrique*, l'utilisateur étant totalement libre de faire ce qu'il souhaite et le système étant beaucoup plus contraint dans ses actions. Nous considérons que c'est au système de s'adapter à l'utilisateur et qu'il ne doit pas entraver les activités de celui-ci.

Type de tâche sous-jacente

Nos travaux se focalisent sur une interaction humain-machine dans le cadre de la résolution d'un problème *épineux* ou *confus* (cf. section 1.1.1). Nous avons montré dans les sections 1.3.1 et 1.3.2, que le processus de résolution de ce type de problème est dynamique, interactif et opportuniste, ce qui le rend difficilement planifiable. C'est ce qui nous pousse à nous rapprocher des *systèmes compagnons* [BLB+17] et à proposer une approche ne s'appuyant pas sur de la planification pour tenter de résoudre le problème sous-jacent à l'interaction.

Nous faisons l'hypothèse que le problème sur lequel porte l'interaction peut être décomposé en plusieurs sous-problèmes, non inter-dépendants ou ordonnés de manière stricte et que chaque sous-problème représente une unité d'action cohérente permettant de faire avancer la résolution du problème.

3.1.2 Formalisation d'un état de l'interaction

Nous définissons maintenant le formalisme utilisé pour décrire la structure de l'interaction entre le système et l'utilisateur. Ce formalisme utilise les engagements sociaux (cf. section 2.3.3), les jeux de dialogue (cf. section 2.3.4). Plus précisément, nous utilisons le modèle de dialogue et de jeux de dialogue de DOGMA (cf. sections 2.4.1 et 2.4.2). Le système s'organise autour d'un état d'information dont la partie publique est le tableau de conversation (cf. section 2.3.2). La partie privée de l'état d'information contient une représentation de la tâche à réaliser sous forme *d'états*, chacun étant une sous-tâche représentant une unité d'action localement cohérente.

Chaque *état* est représenté sous la forme d'une quadruplet $\langle n, Pe, Re, C\rangle$ où :

- n est le *nom* de l'état ;
- Pe est les *prérequis* de l'état, définis à l'aide de prédicats déduits sur l'état d'information ;
- Re est les *règles* de l'état décrivant les conséquences de l'entrée dans cet état, en termes d'engagements sociaux enregistrés dans le tableau de conversation.
- C est l'ensemble des *comportements possibles* de l'état.

Les *comportements possibles* sont différenciés en deux catégories : les *comportements réalisables* et les *comportements attendus*. Un comportement réalisable est initié par le système et un comportement attendu est initié par l'utilisateur.

Un comportement réalisable c_s est un quadruplet $\langle d, Pc, Rc, F\rangle$ où :

- d est le *motif dialogique* du comportement, correspondant à un jeu de dialogue ou un jeu de communication du modèle de DOGMA ;

— Pc est la *précondition* du comportement, définie à l'aide de prédicats déduits sur l'état d'information. Cette condition doit être remplie pour que le système puisse proposer l'ouverture du motif dialogique d ;

— Rc est l'ensemble des *règles* décrivant les conséquences de l'atteinte de certains états de d (entrée dans d, succès ou échec de d), en termes d'engagements sociaux enregistrés dans le tableau de conversation ;

— F est les *effets* de d et Rc en termes d'engagements sociaux quand d est terminé.

Un comportement attendu c_u est un triplet $\langle d, Rc, F\rangle$ ou d, Rc, et F sont définis comme ci-dessus.

Le principe d'un comportement possible est *d'enrichir le motif dialogique* d qui lui est associé en ajoutant les règles Rc correspondant au contexte local de l'interaction conventionnelle d'une sous-tâche donnée.

Un prérequis ou une précondition utilise la logique des prédicats du premier ordre avec les opérateurs classiques de négations, conjonctions et disjonctions (respectivement $\neg$, $\wedge$ et $\vee$). Les prédicats sont déduits à partir de l'état d'information courant du système. De même, les engagements présents dans les effets sont décrits à l'aide de conjonctions et de disjonctions pour pouvoir préciser les différentes issues possibles d'un même comportement. En effet, la formule décrivant les effets d'un comportement possible est toujours vraie à la sortie du motif dialogique associé. De ce fait, ils peuvent être vus comme des *post-conditions* du comportement associé.

Nous utilisons la représentation du tableau de conversation du modèle de dialogue de Dogma donné en section 2.4.1. Celui-ci a la spécificité d'être défini à à chaque instant de l'interaction. L'état du tableau à un instant i est noté T_i. Il est donc possible, lors du calcul d'un prédicat E, d'accéder à l'état courant T_i du tableau de conversation T et à ses états antérieurs $\mathrm{T}_0, \ldots, \mathrm{T}_{i-1}$.

Un motif dialogique correspond à un jeu de dialogue ou de communication (cf. section 2.3.4). Ceux-ci sont des unités d'interaction élémentaires organisant de manière cohérente une séquence d'actes de dialogue et les liant à des engagements sociaux. L'ouverture d'un motif dialogique correspond à l'action de jouer le premier acte de dialogue attendu dans sa séquence. Par extension, l'ouverture d'un comportement possible correspond à l'ouverture de son motif dialogique.

Une règle s'écrit de la manière suivante :

$$\text{Événement} \Rightarrow \mathrm{C}(z, p, s)$$

Où Événement est un événement atteint lors du déroulement du motif dialogique (Entrée, Succès ou Échec), z est n'importe quel interlocuteur, p est une proposition et $\mathrm{C}(z, p, s)$ l'engagement de z sur p dans l'état s. La règle décrite ci-dessus indique que si l'événement Événement a lieu au cours du déroulement du motif dialogique, alors l'engagement $\mathrm{C}(z, p, s)$ est créé.

Pour l'écriture des règles Re d'un état, celles-ci n'étant pas associées à un motif dialogique particulier mais simplement au fait de rentrer dans un état, on utilise une notation anonyme de la règle précédente en l'événement par « _ » :

$$\_ \Rightarrow \mathrm{C}(z, p, s)$$

Pour faciliter la lisibilité des états, nous introduisons une mise en forme de leur écriture à l'aide de tables, appelées *tables d'état*. Un exemple générique de table d'état est donné dans la table 3.1.

⟨**Nom de l'état** (n)⟩	
Prérequis (Pe)	E_0
Règles (Rc)	$\_ \Rightarrow \mathrm{C}(z, \gamma \mid r, s)$
Comportement réalisable (c_s)	
Motif dialogique (d)	$\mathrm{MD}(y, \alpha \mid p)$
Préconditions (Pc)	E_1
Règles (Rc)	Événement $\Rightarrow \mathrm{C}(z, \alpha \mid p, s)$
Effets (F)	$\mathrm{T}_j \ni \mathrm{C}(z, \alpha \mid p, s)$
Comportement attendu (c_u)	
Motif dialogique (d)	$\mathrm{MD}(x, \beta \mid q)$
Règles (Rc)	Événement $\Rightarrow \mathrm{C}(z, \beta \mid q, s)$
Effets (F)	$\mathrm{T}_j \ni \mathrm{C}(z, \beta \mid q, s)$

TABLE D'ÉTAT 3.1 – Table d'état générique. MD est un motif dialogique. α, β et γ sont des actions, p, q et r sont des propositions et E_0 et E_1 sont des prédicats déduits sur l'état d'information. x représente l'utilisateur, y représente le système et z représente l'un ou l'autre. s représente n'importe quel état que peut prendre un engagement. j représente l'instant de fermeture du motif dialogique associé à un comportement réalisable.

Par convention, nous différencions les jeux de dialogue des actes de dialogue en les faisant commencer respectivement par une lettre capitale et une lettre minuscule. Par exemple, Request(x, α) est un jeu de dialogue dans lequel l'acte de dialogue request(x, α) est joué par l'utilisateur.

Même si la table d'état 3.1 se veut être aussi générique que possible, elle ne décrit pas avec exactitude toutes les règles d'écriture possibles. L'idée ici est de donner un aperçu global de la structure d'une table d'état sans énumérer toutes les réécritures possibles pour éviter de surcharger inutilement la table. Il est donc possible que, lors de l'écriture d'une table pour décrire un état de la tâche, quelques petites variations aient lieu, en fonction notamment de certains jeux de dialogue ayant des règles d'écriture légèrement différentes en termes d'implication sur les engagements. Cependant, les variations n'altèrent pas la lisibilité des tables et leur structure globale reste inchangée.

Dans les paragraphes précédents et la table d'état générique, nous avons utilisé deux symboles différents (« | » et « ∨ ») pour représenter la condition « ou ». Cette distinction est réalisée de manière délibérée pour faire la différence entre d'un côté des règles d'écriture d'une table d'état (c'est l'usage du « | ») et d'un autre l'écriture d'une formule sur des prédicats déduits sur l'état d'information (c'est l'usage du « ∨ »).

3.1.3 Organisation d'une table d'état

Cette section a pour but de développer certains aspects spécifiques aux états et tables d'état. Dans un premier temps, nous détaillons les propriétés des comportements attendus au sein d'un état. Ensuite, nous présentons les conventions d'écriture des règles des comportements possibles. Pour finir, nous introduisons une hypothèse simplificatrice sur les engagements en action.

Remarque sur les comportements possibles Rien n'oblige l'interaction à être linéaire ou exhaustive dans le suivi des comportements possibles d'un même état :

— il n'y a à priori pas d'ordre entre les comportements possibles [1] ;

— tous les comportements possibles n'ont pas forcément à être joués (si plusieurs sont possibles, on peut en jouer uniquement un) ;

— plusieurs comportements possibles peuvent être ouverts en parallèle dans un état ;

— en plus des éventuelles préconditions précisées dans le tableau, il faut respecter les conditions d'entrée du motif dialogique pour pouvoir rentrer dans celui-ci.

La définition des comportements possibles a pour but de faire le lien entre les interactions entre les utilisateurs et l'avancement de la tâche sous-jacente. Les comportements possibles décrivent, dans un état donné, la manière de faire avancer la tâche à l'aide d'interactions dialogiques.

Remarque sur les règles Les règles décrites dans un comportement possible associent l'occurrence d'un événement interne au motif dialogique à un ou plusieurs engagements sur le tableau de conversation. Les règles permettent de spécifier un comportement dialogique en enrichissant ses implications dans le contexte local d'un état tout en maintenant la cohérence des engagements concernant la tâche. Elles s'ajoutent aux règles données dans la définition du motif dialogique associé. Par exemple, dans le cadre d'un jeu de dialogue de vérification CheckQuestion(y, $?p$), il est inutile de spécifier que Succès $\Rightarrow$ C(x, p, **Crt**) et que Échec $\Rightarrow$ C(x, $\neg p$, **Crt**) car ces règles ont déjà été décrites dans les effets du jeu de dialogue de vérification CheckQuestion(y, $?p$) [2]. Les tableaux 3.2 et 3.3 illustrent cette règle d'écriture : le premier intègre les règles propres au motif dialogique alors que le second applique la règle d'écriture en les omettant.

S'il n'y a pas de règle à ajouter par rapport à celles apportées par le comportement possible, on met le symbole $\varnothing$ dans la ligne règles. Il est en sus possible préciser que le système à l'interdiction de jouer, au sein du motif dialogique associé, des actes de dialogue menant à un certain événement. Typiquement, pour empêcher le système de faire échouer un motif dialogique, on peut ajouter la règle ¬Échec. Pour rester cohérent avec l'interdiction d'un acte de dialogue, on supprime des effets du comportement les engagements qui auraient été contractés par cet acte de dialogue. L'acte de dialogue associé

1. Il est cependant possible de les séquencer si on le souhaite à l'aide des préconditions.

2. Les règles régissant les jeux de dialogue du modèle DOGMA sont décrites dans la section 6.3 des travaux de thèse de Dubuisson Duplessis [Dub14].

Table avec les règles complètes		
Prérequis	E_0	
Règles	$\varnothing$	
Comportement réalisable		
Motif dialogique	CheckQuestion(y, $?p$)	
Préconditions	E_1	
Règles	Succès $\Rightarrow$ C(x, p, **Crt**) Échec $\Rightarrow$ C(x, $\neg p$, **Crt**)	
Effets	$\vee$	$T_j \ni$ C(x, p, **Crt**) $T_j \ni$ C(x, $\neg p$, **Crt**)

TABLE D'ÉTAT 3.2 – Table d'état exhibant les règles du motif dialogique CheckQuestion.

Table avec les règles condensées		
Prérequis	E_0	
Règles	$\varnothing$	
Comportement réalisable		
Motif dialogique	CheckQuestion(y, $?p$)	
Préconditions	E_1	
Règles	$\varnothing$	
Effets	$\vee$	$T_j \ni$ C(x, p, **Crt**) $T_j \ni$ C(x, $\neg p$, **Crt**)

TABLE D'ÉTAT 3.3 – Table d'état omettant les règles du motif dialogique CheckQuestion.

à une règle est issu du motif dialogique du comportement possible. Il peut correspondre à l'exécution d'un engagement en action dialogique faisant partie du motif. Cependant, il ne se limite pas à cette catégorie d'actes de dialogue et peut aussi correspondre à celui ouvrant le motif, par exemple.

Règles générales d'un état Ces règles ne s'appliquent pas exactement de la même manière que les règles des comportements possibles. En effet, elles ne sont rattachées à aucun motif dialogique. Ces règles ont pour principe d'être plus générales qu'un unique comportement attendu. Elles seront donc directement appliquées au moment où le système décide de passer dans l'état auquel elles sont associées, indépendamment de ce qui a déclenché la transition vers cet état.

L'idée sous-jacente à ce mécanisme est le fait que le contexte local d'un état correspond à une situation particulière de la tâche. Ce contexte local implique une configuration spécifique du contexte de l'interaction permettant le déroulement des comportements possibles de l'état. Notamment, si le passage dans cet état se fait de manière forcée, c'est-à-dire que l'utilisateur « court-circuite » les prérequis d'un état, cela signifie que certaines actions sont sous-entendues et ne sont donc pas réalisées par des actes dialogiques.

L'intérêt d'ajouter des règles aux prérequis d'un état est de capturer ce phénomène en rattrapant le retard accumulé par le système suite à un raccourci de l'utilisateur. Il permet alors de modifier la représentation du monde du système de manière à valider à posteriori les prérequis de l'état. Ce mécanisme donne plus de souplesse à l'interaction car il tente de compenser des ellipses faites par l'utilisateur dans le déroulement de sa tâche sans le solliciter explicitement.

Conditionnement des règles Il est possible de définir une condition de garde sur les règles d'un état ou d'un motif dialogique. Celle-ci se définit à l'aide de conditions sur le tableau de conversation. Elle s'écrit sous la règle qu'elle conditionne, en la précédant du symbole « ⌊ ». Par exemple la règle suivante est conditionnée :

$$\begin{array}{l}\text{Entrée} \Rightarrow \mathrm{C}(x, q, \textbf{Ina}) \\ \lfloor\ \mathrm{T}_i \ni \mathrm{C}(x, q, \textbf{Crt})\end{array}$$

Cette écriture veut dire que la règle Entrée ⇒ C(x, q, **Ina**) n'est appliquée que si le tableau de conversation contient l'engagement C(x, q, **Crt**).

Cette possibilité est décrite dans le tableau 3.4. Dans cet exemple, les deux règles sont conditionnées. La règle générale de l'état indique que l'entrée dans cet état implique le passage de l'engagement sur q dans l'état s si celui-ci est dans l'état s'. La règle du comportement attendu indique que si, au moment où il rentre dans le motif d'Inform, l'utilisateur est engagé sur la proposition q ($\mathrm{T}_i \ni \mathrm{C}(x, q, \textbf{Crt})$), alors cet engagement est désactivé (il passe dans l'état **Ina**).

Table avec conditionnement des règles		
Prérequis	E_0	
Règles	$\_ \Rightarrow \mathrm{C}(z, q, s)$ $\lfloor\ \mathrm{T}_i \ni \mathrm{C}(z, q, s')$	
Comportement attendu		
Motif dialogique	Inform(x, p)	
Règles	Entrée ⇒ C(x, q, **Ina**) $\lfloor\ \mathrm{T}_i \ni \mathrm{C}(x, q, \textbf{Crt})$	
Effets	$\wedge$	$\mathrm{T}_j \ni \mathrm{C}(x, p, \textbf{Crt})$ $\mathrm{T}_j \ni \mathrm{C}(x, q, \textbf{Ina})$

TABLE D'ÉTAT 3.4 – Table d'état avec une condition sur une règle.

Remarque sur les engagements en action Les engagements en action (suggestion, requête et offre) sont contractés sur des actions qui *se dérouleront dans le futur*. Donc lorsque ces engagements sont contractés, on ne sait pas quand les actions sur lesquels ils portent seront réalisées. En ce qui concerne les actions promises par l'utilisateur, il n'y a aucune possibilité de savoir quand elles auront lieu. À l'inverse, pour les actions promises par le système, il a tout à gagner à réaliser cette action au plus tôt. On considère donc que

tout engagement en action du système implique la réalisation directe de cette action. On aura donc dans la partie « effets » du tableau les engagements résultants de *la réalisation* de l'action sur laquelle le système s'est engagé (on simplifie en partant du principe que les conditions nécessaires sont toujours réunies au moment de l'engagement en action [3]). Par contre pour l'utilisateur, c'est uniquement son engagement en action qui sera dans la partie « effets » et non pas les engagements qui en résultent.

3.2 Processus décisionnels d'un agent collaborant avec un humain

Cette section a pour but de décrire la manière dont il est prévu que l'interaction se déroule. Les processus décisionnels de l'agent sont décrits ici, de même que la contextualisation d'un acte de dialogue de l'utilisateur et la méthode permettant de passer d'un état à un autre.

Nous présentons dans un premier temps (section 3.2.1) les définitions nécessaires à la description des processus décisionnels de l'agent. Nous décrivons ensuite (section 3.2.2) le processus de contextualisation d'un acte de dialogue de l'utilisateur. La troisième partie (section 3.2.3) donne la méthode de sélection d'un comportement réalisable par le système. Enfin, la dernière partie (section 3.2.4) montre quels sont les différents types d'échecs que le système peut rencontrer et comment il les traite.

3.2.1 Définitions

Il est nécessaire dans un premier temps de définir les concepts que nous allons employer pour organiser les processus décisionnels du système. Nous définissons notamment le concept de *maturité* qui peut être associé à un comportent réalisable ou à un état et qui sert de base pour le raisonnement du système.

Définition 3.1 – Maturité d'un comportement réalisable : à tout comportement réalisable on associe une valeur de « maturité ». Le calcul de la maturité tient compte de l'état d'activation des préconditions du comportement et de la maturité du comportement au pas de temps précédent. La formule définissant la maturité $m_{c,n+1}$ d'un comportement réalisable c à un rang $n+1$ est la suivante :

$$m_{c,0} = 0.$$
$$m_{c,n+1} = \begin{cases} \frac{\alpha + m_{c,n}}{\alpha + 1} & \text{si les préconditions de } c \text{ sont satisfaites ;} \\ 0 & \text{sinon.} \end{cases}$$

Avec $\alpha > 0$ un paramètre déterminant le taux de croissance $m_{c,n}$: plus α est élevé et plus ce taux est important.

3. Cette simplification est réductrice et dépend de la tâche sous-jacente à l'interaction. Cependant elle semble raisonnable dans le cas où l'interaction a lieu dans un environnement où l'état des objets est déterministe, comme c'est le cas pour les environnements purement numériques.

On a donc la formule suivante pour la maturité d'un comportement réalisable c dont les préconditions sont satisfaites depuis n pas de temps :

$$m_{c,n} = 1 - \frac{1}{(\alpha + 1)^n}.$$

La valeur de maturité d'un comportement réalisable sera donc toujours positive et comprise dans l'intervalle $[0, 1[$ et plus un comportement sera activé longtemps et plus sa maturité sera grande.

Nous généralisons maintenant la définition de la maturité à un état.

Définition 3.2 – Maturité d'un état : la maturité d'un état correspond à la moyenne des maturités des comportements réalisables de l'état. La maturité $M_{E,i}$ d'un état E à un instant i est donc donnée par la formule suivante :

$$M_{E,i} = \frac{1}{\#E.C_s} \sum_{c \in E.C_s} m_{c,i}.$$

Avec $E.C_s$ l'ensemble des comportements réalisables (donc uniquement ceux du système) de E et $\#E.C_s$ le nombre de comportements réalisables dans cet état.

La maturité d'un état n'a de sens que si cet état contient au moins un comportement réalisable. Associer une maturité à un état ne contenant que des comportement attendus (donc initiés par l'utilisateur) ne signifie rien.

Remarque 3.1 – Maturité d'un état : on peut voir la maturité d'un état comme représentant le potentiel d'interaction du système dans cet état-là : si l'état est mature, c'est que le système est capable d'être à l'initiative d'interactions avec l'utilisateur. Un état mature représenterait donc un état au sein duquel, si on y entrait, il pourrait se dérouler des actions à l'initiative du système, sollicitant l'utilisateur pour faire avancer la tâche.

Définition 3.3 – État courant : l'état courant est l'état dans lequel l'interaction se trouve au temps courant i.

Définition 3.4 – États candidats : les états candidats sont les états différents de l'état courant dont les prérequis sont satisfaits au temps courant i.

L'utilisation de la maturité d'un état est prépondérante dans la prise de décision du système pour le passage d'un état à un autre, comme présenté dans les sections suivantes.

3.2.2 Contextualisation d'un acte de dialogue de l'utilisateur

La contextualisation a pour but d'intégrer un nouvel acte de dialogue de son interlocuteur à la représentation que l'on a actuellement de l'interaction tout en conservant la cohérence de celle-ci. Étant donné que l'interaction est menée par l'utilisateur, c'est au système de contextualiser au mieux les actes de dialogues de celui-ci de manière à

réagir aussi bien que possible à ses actions. Quand un nouvel acte de dialogue est joué par l'utilisateur, le système doit le situer dans sa représentation courante de l'interaction (c'est-à-dire l'état d'information). Il essaie d'abord de situer l'acte de dialogue dans le contexte dialogique courant en tentant de le contextualiser dans un jeu de dialogue ouvert. S'il ne trouve pas de jeu de dialogue ouvert attendant l'acte de dialogue de l'utilisateur, il tente de le situer dans un comportement attendu de l'état courant s'ouvrant par l'acte de dialogue de l'utilisateur. Si ce n'est pas possible, il essaie de trouver dans un autre état un comportement s'ouvrant par l'acte de dialogue de l'utilisateur. De cette manière, il essaie de contextualiser l'acte de dialogue de l'utilisateur du contexte le plus restreint (l'état courant du dialogue) au contexte le plus général (l'ensemble des états) de l'interaction et le passage dans un autre état n'est considéré qu'en dernier recours.

L'algorigramme 3.1 illustre le fonctionnement de la contextualisation d'un acte de dialogue de l'utilisateur par le système. Il décrit en sus le mécanisme de transition entre les différents états de la tâche. Nous en détaillons le fonctionnement dans la suite de cette section.

Nous considérons que l'ouverture d'un comportement possible n'est effective que lorsque l'utilisateur a joué un acte de dialogue de ce comportement. Dans le cas d'un comportement attendu, celui-ci étant initié par l'utilisateur, il est ouvert dès que l'utilisateur joue son premier acte. Dans le cas d'un comportement possible, celui-ci étant initié par le système, il *suggéré* lorsque le système joue le premier acte et n'est ouvert que si l'utilisateur joue un acte de réponse.

Passage d'un état à un autre

Les tables d'état contiennent toute l'information nécessaire pour passer d'un état à un autre. En effet, dès qu'un état a ses prérequis satisfaits, il devient candidat et il est donc possible de rentrer dans celui-ci. Cependant, le passage d'un état à un autre mérite d'être plus subtil et il n'est pas nécessaire de se « précipiter » dans un état dès que celui-ci est accessible. Il semble plus approprié d'attendre d'avoir une raison de passer dans un nouvel état plutôt que d'y passer dès que possible. Ces raisons sont provoquées soit par l'utilisateur, soit par le système :

— l'utilisateur peut provoquer un changement d'état s'il joue un comportement inattendu dans l'état courant mais attendu dans un état candidat ;
— le système peut provoquer un changement d'état s'il propose un comportement réalisable dans un état candidat et que l'utilisateur accepte de rentrer dans le motif dialogique associé au comportement (il joue un *acc.entrée* dans le jeu de communication de contextualisation du motif dialogique associé).

Cela correspond au losange 3 dans l'algorigramme 3.1. Dans les deux cas, la transition entre les états est déclenchée uniquement par un acte de dialogue de l'utilisateur.

Cas d'un conflit de passage

Le fonctionnement décrit dans la partie précédente nécessite d'être raffiné dans le cas où l'utilisateur joue un coup dialogique attendu dans deux états candidats. Même si à

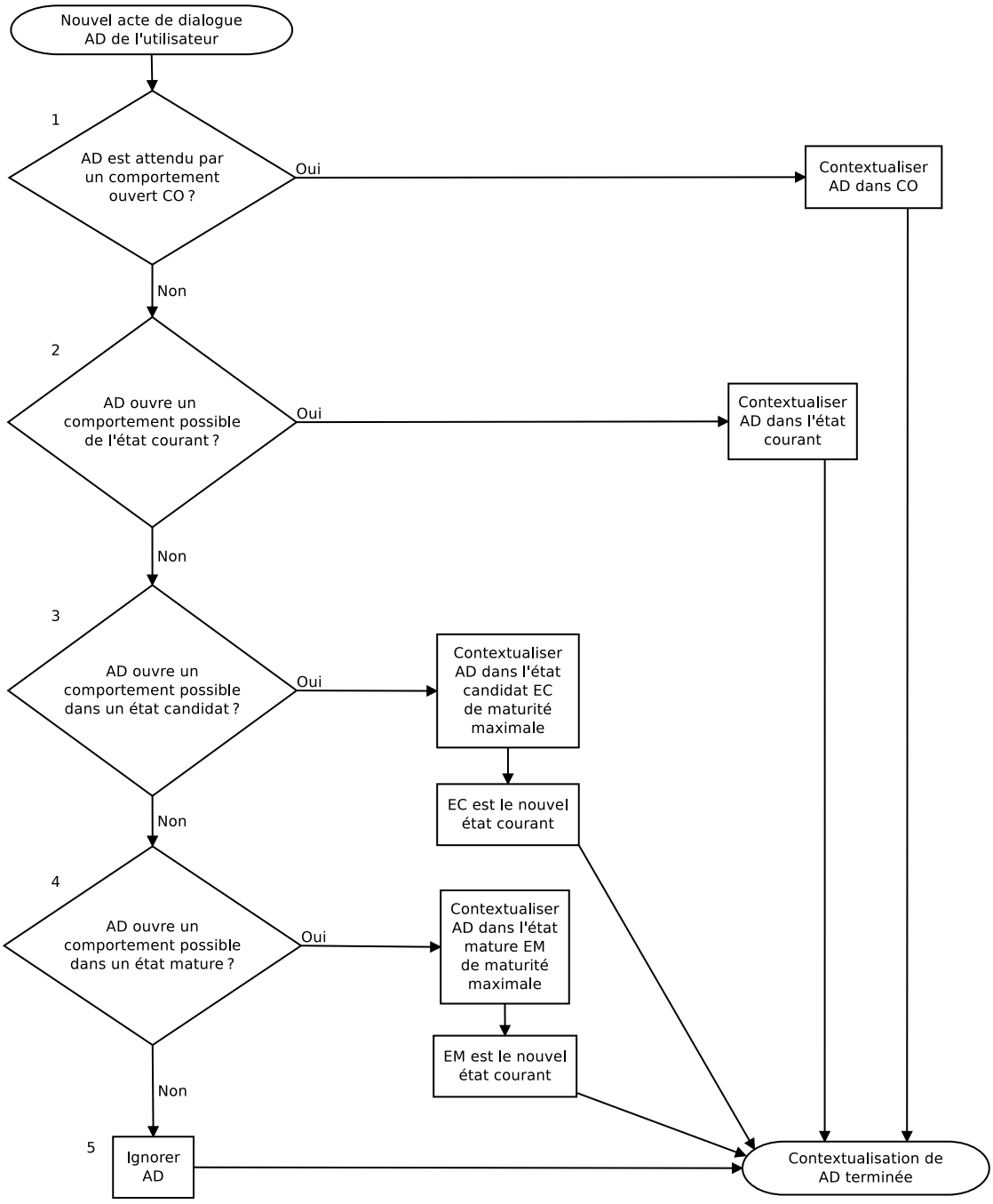

FIGURE 3.1 – Algorigramme décrivant la contextualisation d'un acte de dialogue de l'utilisateur.

priori ce cas a peu de chances d'arriver, rien ne l'empêche et c'est la manière dont seront définis les prérequis des tables d'état qui permettront ou non d'éviter ce type de situation. Nous proposons un mécanisme spécifique pour résoudre ces conflits dans le cas où ils auraient lieu.

Dans cette situation, il semble préférable de se diriger vers l'état où le système aura la capacité de contribuer à l'interaction et faire avancer la tâche. Si l'on s'appuie sur la définition 3.2 de la maturité d'un état et la remarque 3.1 associée, la maturité d'un état est un indicateur pertinent pour permettre de choisir de rentrer dans l'état où le système sera capable de prendre le plus d'initiatives. Dans cette situation, on privilégiera donc l'état dont la maturité est la plus élevée.

Dans le cas, où plusieurs états auraient la maturité la plus élevée, on peut envisager de tirer au hasard parmi ces états celui qui sera choisi comme destination. Une autre solution est d'ouvrir le comportement attendu sans changer d'état, et prendre cette décision en fonction des interactions suivantes de l'utilisateur.

Transition mature

Si l'on s'en réfère à la description donnée précédemment, pour passer d'un état à un autre suite à une action de l'utilisateur, il faut que cette action soit inattendue dans l'état courant et attendue dans un état candidat. Cependant, permettre uniquement ce type de transition est très restrictif et rendrait le fonctionnement du système très rigide. En effet, il serait alors incapable de contextualiser une action de l'utilisateur dans un état tant que les préconditions de celui-ci ne seraient pas satisfaites, menant à de nombreux échecs de contextualisation. Nous introduisons donc le mécanisme de *transition mature*, qui permet, faute d'état candidat disponible, de contextualiser une action de l'utilisateur dans un comportement attendu d'un état dont les prérequis ne sont pas satisfaits mais dont la maturité est non-nulle. Cela correspond au losange 4 dans l'algorigramme 3.1.

L'idée ici est de faire primer la décision de l'utilisateur sur la définition des prérequis et de s'adapter à ses actions quitte à ignorer les règles en place. Cependant, l'assouplissement du passage d'un état à un autre sur un coup de l'utilisateur ne doit pas se faire au prix de la capacité du système à initier des interactions avec l'utilisateur. Ce mécanisme reste donc conditionné par l'état destination, qui doit alors avoir une maturité strictement positive ($E.C_s \neq 0$ et $M_{E,i} > 0$) ou qui doit contenir uniquement des comportements attendus ($E.C_s = 0$).

Dans le cas où le système n'est pas capable de contextualiser une action de l'utilisateur, même à l'aide d'une transition mature, la contextualisation échoue et l'action est ignorée. Ce cas correspond au bloc 5 dans l'algorigramme 3.1.

Lien entre la contextualisation et l'accommodation

Dans un article fondateur, Lewis [Lew79] compare le maintien d'une représentation cohérente de l'état de la conversation à la comptabilisation des scores lors d'un match de baseball. Dans un match de baseball, le tableau des scores est mis à jour à chaque tour de jeu en appliquant les règles de ce sport. De la même manière au cours d'une

conversation, le tableau de conversation évolue afin d'intégrer les coups dialogiques joués par les interlocuteurs.

Dans certains cas, il est nécessaire pour le destinataire d'un acte de dialogue de modifier sa représentation de la conversation et des sujets qui s'y rapportent afin d'y intégrer un acte de son interlocuteur. Ce phénomène est appelé *accommodation* et a pour principe de faire évoluer l'état actuel du dialogue afin de rendre pertinent l'acte à contextualiser.

Cependant, à l'inverse d'un match de baseball où deux équipes s'affrontent, au cours d'une conversation les deux interlocuteurs coopèrent afin de favoriser leur compréhension mutuelle. C'est ce que Lewis décrit en disant que « le score conversationnel tend à évoluer de manière à permettre à n'importe quel acte d'être un coup correct »([Lew79], p.347). L'accommodation amène celui qui reçoit l'acte à faire des *présuppositions*, apportant un contexte sous-entendu rendant certaines actions *acceptables* et permettant ainsi la contextualisation de l'acte.

Roberts [Rob15] renforce cette vision du dialogue comme une activité coopérative en présentant l'accommodation comme un *principe de coopération* mis en œuvre au cours d'une conversation et qui « ajuste la représentation du dialogue pour éliminer les obstacles au plan détecté chez son interlocuteur »([Rob15], p.354). Cette vision va plus loin en expliquant que l'accommodation consiste à reconnaître les plans, buts et intentions de son interlocuteur afin de modifier ses propres plans, buts et intentions pour faire avancer la conversation.

Même si notre agent ne réalise pas de reconnaissance de plan ou de but, celui-ci réalise une forme d'accommodation lorsqu'il reçoit un acte de dialogue de la part de l'utilisateur. En effet, c'est à ce moment qu'il peut décider de modifier le cadre dans lequel il se trouve et de changer d'état afin de trouver un comportement attendu permettant de contexualiser l'acte de dialogue de l'utilisateur. Ce changement d'état lui permet de se situer dans un nouveau cadre où il est possible de lier l'acte de dialogue de l'utilisateur à un contexte pertinent à la fois au niveau de l'interaction et du dialogue.

3.2.3 Sélection d'un comportement réalisable

Comme défini en section 3.1, les comportements réalisables sont ceux initiés par le système. Cette section présente comment celui-ci prend l'initiative de proposer un comportement réalisable en utilisant le concept de maturité donné en définition 3.1. Il est d'abord nécessaire de préciser le concept de *comportement candidat*.

Définition 3.5 – Comportement candidat : un comportement candidat est un comportement réalisable de l'état courant ou d'un état candidat dont les préconditions sont satisfaites.

Pour sélectionner le comportement qu'il souhaite proposer le système va, à chaque état, récupérer l'ensemble des comportements candidats et conserver celui dont la maturité est la plus élevée. Dans le cas où plusieurs comportements candidats auraient la maturité maximale, on prend parmi eux ceux de l'état courant et on en pioche un au hasard. Dans le cas où il n'y aurait pas de comportement candidat de l'état courant parmi ceux de

maturité maximale, on pioche au hasard parmi ces derniers. Le comportement sélectionné par le système devient alors un *comportement proposé*.

Définition 3.6 – Comportement proposé : un comportement proposé est un comportement réalisable dans lequel le système a proposé à l'utilisateur de rentrer, en jouant un coup *prop.in*.

On peut considérer sans perte de généralité que le temps de calcul pour que le système sélectionne un comportement réalisable est négligeable. Cela implique le besoin de limiter le nombre de comportements réalisables que le système peut proposer en parallèle. En effet, si aucune limite n'est fixée, le système proposerait tous les comportements candidats en même temps, ce qui risque de surcharger l'utilisateur d'information plutôt que de lui suggérer des actions pertinentes. Pour prévenir ce type de situations, nous proposons de limiter le nombre de comportements proposés en parallèle a une valeur $\tau_{parallèle}$ et de limiter la somme des maturités des comportements proposés à une valeur maximum $\tau_{maturité}$.

De cette manière, le système sera limité à proposer une quantité limité de comportements. Le nombre de comportements qu'il proposera sera d'autant plus limité que leur maturité est élevée. Cela envisage la maturité d'un comportement réalisable sous l'aspect de la *confiance* que le système à en sa réussite : si un comportement est très mature, c'est qu'il a de fortes chances d'être pertinent dans le contexte local (ses préconditions sont vraies depuis longtemps). Le système peut donc être optimiste et ne proposer que celui-ci ou celui-ci et un nombre restreint d'autres comportements. À l'inverse, si le système n'a à sa disposition que des comportements réalisables dont la maturité est faible, il semble raisonnable d'en proposer un plus grand nombre à l'utilisateur et de laisser celui-ci choisir celui qui lui convient le mieux.

De manière à éviter que le système ne se retrouve bloqué en ayant proposé des comportements qui sont restés sans réponse par l'utilisateur, un mécanisme de décroissance de la maturité des comportements proposés est mis en place. La formule de décroissance d'un comportement proposé est la suivante :

$$m_{c,n+1} = \frac{m_{c,n}}{\alpha + m_{c,n}}$$

Où le α est le même que celui de la définition de maturité (cf. définition 3.1). Celle-ci est appliquée à partir de l'instant p où le comportement c est proposé par le système.

Avec ce mécanisme de décroissance de la maturité, il est possible de définir un seuil ρ en dessous duquel la maturité d'un comportement est considérée comme insuffisante. Le paramètre ρ a une influence sur deux aspects :

— il définit la maturité minimale qu'un comportement réalisable doit avoir pour être joué par le système ;

— il définit une limite à partir de laquelle un comportement réalisable est considéré comme obsolète et que, étant resté un certain temps sans réponse, il semble ignoré par l'utilisateur.

Dans le second cas, on considère alors que l'utilisateur refuse d'entrer dans le motif dialogique du comportement réalisable, c'est à dire qu'il joue un *ref.in* dans le jeu de communication de contextualisation du motif dialogique.

De cette manière, le système évite d'accumuler des comportements ouverts restés sans réponse. Cela lui permet de rester actif et de continuer à proposer des interactions même sans retour explicite de l'utilisateur sur les comportements proposés.

3.2.4 Gestion des échecs dialogiques et extra-dialogiques

Les motifs dialogiques permettent de différencier les échecs et succès *dialogiques* des échecs et succès *extra-dialogiques* [DD15].

Un succès dialogique est atteint quand le motif dialogique est réalisé en respectant les règles qui le définissent, c'est-à-dire qu'il se déroule « normalement », indépendamment du sujet sur lequel il porte. Un échec dialogique arrive quand la progression du dialogue prend un tour conventionnellement inattendu, comme le non-respect d'une règle ou l'abandon de l'activité. C'est notamment le cas lorsque l'un des interlocuteurs refuse de rentrer dans un motif dialogique proposé par son partenaire.

Le statut extra-dialogique du motif dépend quant à lui du sujet sur lequel il porte. Si l'activité conjointe sous-jacente au motif est complétée, alors celui-ci est un succès extra-dialogique. Dans le modèle DOGMA, l'état d'échec ou de réussite extra-dialogique est explicitement spécifié en termes d'engagements sociaux dans la description du motif dialogique.

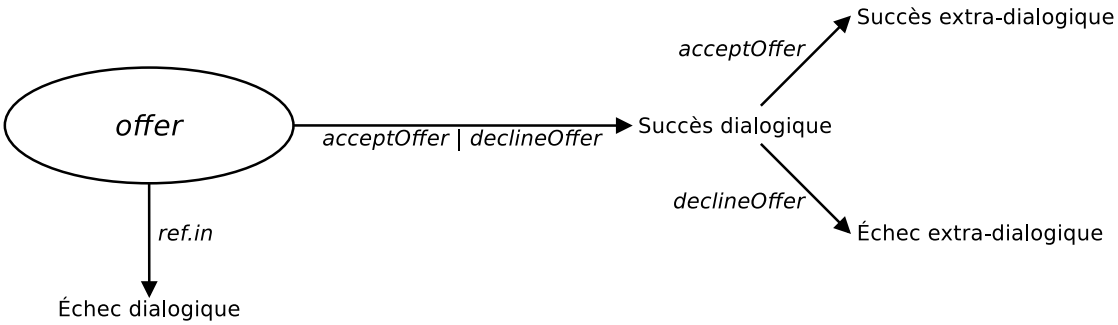

FIGURE 3.2 – Illustration du modèle d'activité dialogique à l'aide du jeu de dialogue de discussion d'action Offer.

Un statut extra-dialogique ne peut être accordé à un motif dialogique que si celui-ci est un succès dialogique. En effet, il n'est pas approprié d'évaluer la réussite de l'activité conjointe sous-jacente à un motif dialogique qui ne s'est pas ou mal déroulé. Cependant un motif dialogique peut se dérouler de manière tout à fait conventionnelle tout en n'atteignant pas ses conditions de succès extra-dialogiques.

La figure 3.2 illustre les différentes issues possibles en termes de réussite et d'échec dialogique et extra-dialogique à l'aide du jeu de dialogue d'offre. L'initiateur ouvre le jeu à l'aide de l'acte de dialogue *offer*. Son partenaire peut ignorer celui-ci (il joue alors un *ref.in* dans le jeu de communication de contextualisation associé), menant à un échec

dialogique. Si le partenaire rentre dans le jeu de dialogue (il joue un *acceptOffer* ou un *declineOffer*, ce qui entraine un *acc.in* dans le jeu de contextualisation), les conditions de succès dialogiques sont atteintes. Dans le cas où l'utilisateur joue un *acceptOffer*, les conditions de succès extra-dialogique sont atteintes, dans le cas où il joue un *declineOffer* les conditions d'échec extra-dialogique sont atteintes.

La distinction de ces échecs par notre système permet de différencier les causes du rejet de l'utilisateur : notre système considère comme un échec dialogique le fait que l'utilisateur refuse explicitement de rentrer dans un motif dialogique, ou qu'il n'y réagisse pas, c'est-à-dire qu'il le refuse implicitement avec le mécanisme d'obsolescence décrit dans la section 3.2.3. Il considère comme un échec extra-dialogique le fait que les conditions d'échec d'un motif dialogique s'étant déroulé correctement soient atteintes.

Si l'on s'en réfère au modèle en « 5-C » de Taylor-Powell [TPRG98] présenté en section 1.1.1, on peut voir un échec dialogique comme un échec de *coordination* de la part de l'agent, c'est-à-dire qu'il emploie un motif dialogique au mauvais moment. Un échec extra-dialogique se réfère à un échec de raisonnement sur la tâche sous-jacente à l'interaction, et non pas à un échec dans le processus collaboratif.

Cette différenciation pourrait être utilisée afin de mettre en place un mécanisme de pénalisation des comportements réalisables. En effet, un échec dialogique indique un mauvais choix de motif dialogique de la part du système, alors qu'un échec extra-dialogique indique un mauvais choix du contenu sémantique du motif dialogique. Une pénalité s'appliquerait alors lors du calcul de la maturité des comportements concernés afin de les désavantager par rapport à d'autres comportements n'ayant pas encore été testés. Les travaux sur l'utilisation des échecs dialogiques et extra-dialogiques sont laissés à des développements ultérieurs.

3.3 Gestion de l'interaction

Cette section présente l'organisation de l'interaction. Nous y précisons notamment les points clés de son déroulement et les mécanismes mis en place pour l'enrichir. Nous présentons dans un premier temps (section 3.3.1) le concept de comportement *proactif*, dont le but est d'augmenter les capacités d'interaction du système en lui permettant de créer de nouveaux comportements à partir de ceux attendus de la part de l'utilisateur. Dans un second temps (section 3.3.2), nous discutons de la possibilité, dans le contexte de notre modèle, de relaxer les contraintes des motifs dialogiques.

3.3.1 Comportements proactifs

Nous présentons ici un mécanisme supplémentaire du système dont le but est d'enrichir ses capacités d'interaction. Ce mécanisme se base sur les comportements *proactifs*, une catégorie spécifique de comportements réalisables par le système. Ils ont pour but d'inciter l'utilisateur à réaliser certains des comportements qui sont conventionnellement attendus de sa part dans le contexte de l'état actuel. Ils ne sont pas explicitement précisés dans les états mais sont déduits par le système à partir de certains comportements attendus,

dont le motif dialogique est soit un jeu de communication de transfert d'information (Inform), soit un jeu de dialogue de proposition d'action (Offer), soit un jeu de dialogue de demande d'action (Suggestion ou Request). À chacun de ces comportements attendus va correspondre un comportement proactif dont le motif dialogique est précisé en tableau 3.1.

Utilisateur		Système
Inform(x, p)	$\Rightarrow$	CheckQuestion(y, $?p$)
Inform(x, $p(a)$)	$\Rightarrow$	Inform(y, UserCanInformOn(p))
Offer(x, α)	$\Rightarrow$	Inform(y, UserCanDo(α))
Suggestion ou Request(x, α)	$\Rightarrow$	Inform(y, UserCanRequest(α))

TABLEAU 3.1 – Correspondance entre les motifs dialogiques des comportements attendus et ceux des comportements proactifs.

Ce tableau décrit quatre catégories de comportements proactifs. Le premier correspond à un scénario où il est conventionnellement attendu que l'utilisateur réalise un transfert d'information sur un prédicat d'arité 0 à l'aide d'un jeu d'Inform. Le motif dialogique du comportement proactif associé est alors un jeu de vérification (CheckQuestion), poussant l'utilisateur à s'engager sur le prédicat p ou sa négation. Le second diffère du premier sur le fait que le transfert d'information est réalisé sur un prédicat d'arité supérieure ou égale à 1. De ce fait, il n'est pas possible de demander à l'utilisateur de s'engager sur le prédicat lui-même étant donné que celui-ci est appliqué à une ou plusieurs variables (à a dans le tableau 3.1). Il n'est donc possible que de préciser à l'utilisateur le prédicat (ici p) sur lequel il est sensé s'engager. Cela est réalisé à l'aide d'un jeu de transfert d'information : Inform(x, UserCanInformOn(p)). Les deux dernières lignes correspondent à des motifs dialogiques de discussion d'action : Offer est un jeu de proposition d'action et Suggestion et Request sont des jeux de demande d'action. Les motifs des comportements proactifs associés ont donc pour but de signaler à l'utilisateur qu'il est conventionnellement attendu qu'il réalise les actions concernées. Pour cela, un jeu de transfert d'information est utilisé avec un Inform(y, UserCanDo(α)) pour signaler à l'utilisateur qu'il peut réaliser α et avec un Inform(y, UserCanRequest(α)) pour signaler à l'utilisateur qu'il peut demander au système de réaliser l'action α.

Les règles et les effets d'un comportement proactif sont exactement les mêmes que ceux du comportement attendu dont il est issu.

La table d'état 3.5 illustre le fonctionnement des comportements proactifs en exhibant celui associé au jeu de communication d'Inform.

L'objectif des comportements proactifs est double :

— d'un côté, ils permettent de relancer de manière pertinente un utilisateur n'ayant pas réalisé d'action depuis un certain temps (correspondant au prédicat UserPassive dans les préconditions) ;

— d'un autre côté, ils permettent au système de suggérer à l'utilisateur des actions localement pertinentes pour faire avancer la tâche (correspondant au prédicat UserNeedsHelp dans les préconditions).

Table avec comportement proactif	
Prérequis	E_0
Règles	$\varnothing$
Comportement attendu	
Motif dialogique	Inform(x, p)
Règles	$\varnothing$
Effets	$T_j \ni C(x, p, \mathbf{Crt})$
Comportement proactif associé	
Motif dialogique	CheckQuestion(y, ?p)
Préconditions	UserPassive $\vee$ UserNeedsHelp
Règles	$\varnothing$
Effets	$T_j \ni C(x, p, \mathbf{Crt})$

TABLE D'ÉTAT 3.5 – Table d'état exhibant le comportement proactif correspondant au jeu de communication d'Inform.

Ce second point est particulièrement intéressant dans le cas où l'utilisateur serait désemparé face à la tâche qu'il est sensé réaliser et aurait besoin de suggestions d'actions pour continuer d'avancer.

Les comportement proactifs se déclenchant dans des situations d'interaction bien spécifiques (inactivité de l'utilisateur ou demande d'aide de la part de celui-ci), ils n'ont pas de maturité et ne sont donc pas soumis aux règles des comportement réalisables.

3.3.2 Relaxation des contraintes d'un motif dialogique

Nous avons indiqué en section 3.1.3 qu'un comportement n'est jouable que si les préconditions de son motif dialogique sont respectées. Nous revenons ici sur cette règle et y ajoutons quelques exceptions.

Nous ne souhaitons pas que la rigidité des règles régissant notre modèle entrave la progression de l'utilisateur. Nous considérons qu'il faut laisser celui-ci s'exprimer librement même si cela signifie enfreindre certaines règles des motifs dialogiques et si cela n'est pas strictement cohérent dans le modèle dialogique.

Cela est dû à la difficulté de l'activité ayant motivé le dialogue de l'humain avec le système. Dans le cas d'un problème confus (cf. section 1.1.1), sa résolution peut nécessiter des changements d'opinion, des retours en arrière et une redéfinition du problème menant à une méthode de résolution différente. De ce fait, il est envisageable que l'opinion de l'utilisateur évolue au cours de l'interaction et qu'il se contredise.

Or, les règles des jeux de dialogue de DOGMA définissent des contraintes sur l'état du dialogue pour pouvoir rentrer dans un motif dialogique. Par exemple, pour jouer le jeu de dialogue Request(x, α), il faut que l'on ait aucune connaissance sur α ou $\neg\alpha$, c'est-à-dire $T_i \ni C(y, \alpha, \mathbf{Ina})$ et $T_i \ni C(y, \neg\alpha, \mathbf{Ina})$, x étant l'initiateur du jeu et y son partenaire.

Plaçons-nous maintenant dans une situation où le système (*sys*) et l'utilisateur (*util*) interagissent. Arrive un moment où le système propose à l'utilisateur de réaliser l'action

α. Cela est réalisé à l'aide d'un Offer(*sys*, α). L'utilisateur refuse l'offre du système (il joue un declineOffer(*util*, α)), ce qui mène à la création de l'engagement C(*sys*, α, **Fal**). Jusqu'ici, tout est cohérent. Si, dans la suite de l'interaction l'utilisateur change d'avis et souhaite que le système réalise l'action α, il va le faire à l'aide du jeu de dialogue de requête présenté ci-dessus. Cependant, les conditions d'entrée du jeu de dialogue ne sont pas vérifiées, puisque $T_i \ni C(sys, \alpha, \textbf{Fal})$. L'utilisateur n'a donc théoriquement pas le droit de faire cette requête au système, même s'il semble légitime qu'il change d'avis.

Ce type de situation nous pousse à considérer que les règles des motifs dialogiques de DOGMA ne sont pas toujours adaptées et que nous devons les contourner dans certaines situations.

S'il n'est pas possible de contextualiser un acte de dialogue de l'utilisateur de la manière décrite en section 3.2.2, nous permettons en dernier recours d'enfreindre les règles d'un motif dialogique associé à un comportement attendu si cela permet de contextualiser l'acte. L'idée ici est la même que pour une transition mature entre états (cf. section 3.2.2), à savoir donner la priorité à l'utilisateur et lui permettre de se comporter comme il le souhaite sans restriction de liberté. Le système doit s'adapter tant qu'il le peut aux actions de l'utilisateur, même si cela implique d'assouplir ses règles d'interaction.

De la même manière, le système peut enfreindre les mêmes règles dans le cadre des comportements proactifs. En effet, le mécanisme des comportements proactifs étant prévu pour faire avancer l'interaction dans des situations où celle-ci est bloquée, il est intéressant d'élargir au maximum les propositions faites par le système pour la relancer, même si cela implique de ne pas respecter strictement les règles dialogiques en place.

Nous ne permettons cependant pas la relaxation des contraintes sur les comportements réalisables par le système autrement que pour les comportements proactifs. En effet, nous considérons que, si l'utilisateur a le droit de prendre des libertés par rapport aux normes d'interaction, cela ne doit pas être le cas de la machine.

3.4 Discussion

Terveen [Ter95], ensuite repris par Fischer [Fis01] présente deux visions distinctes de la collaboration humain-machine. D'un côté, l'approche par *émulation* se base sur la métaphore de donner aux machines des aptitudes similaires à celles des humains. De l'autre, l'approche par *complémentarité* reconnaît la différence entre les humains et les machines et que cette asymétrie doit être exploitée pour développer des modes d'interaction et de collaboration [Suc07]. Nous défendons ici l'idée d'une approche mixte, tirant parti à la fois de l'utilisation d'une formalisation du dialogue pour se rendre accessible et interagir avec l'utilisateur humain et de comportements complémentaires tirant parti des capacités d'accès et de traitement de l'information de la machine.

Les capacités d'interaction de notre système dépassent le comportement simplement réactif, avec notamment la conception de capacités *proactives* dont le but est de solliciter l'utilisateur sur les actions qui sont attendues de sa part. Ce type de comportement est utile à la fois pour relancer un utilisateur qui aurait été inactif pendant un certain temps, et pour guider celui-ci dans le cas où il se trouverait dans une situation où il ne

sais pas quelles actions réaliser pour faire avancer la tâche. Ce mécanise se base sur les *comportements attendus* et leur motif dialogique pour guider l'utilisateur et lui suggérer des comportements pertinents dans l'état actuel de la tâche.

Un comportement possible enrichit le motif dialogique qui lui est associé à l'aide de règles localement et conventionnellement pertinentes. Ces règles permettent de gérer les conséquences indirectes (c'est-à-dire non décrites par un motif dialogique) d'un acte de dialogue. Cela est rendu possible par le fait qu'un état n'est que localement cohérent et correspond à un contexte particulier lié à une sous-tâche spécifique.

Contrairement à un système où le dialogue ferait l'objet d'une planification par le système, notre système prend le parti de ne respecter que les normes conventionnelles des interactions dialogiques sans chercher à planifier cette interaction. Cependant, cela ne signifie pas qu'il n'est pas possible de structurer le déroulement de l'interaction. L'utilisation de préconditions sur les états permet de gérer le séquencement de celles-ci en s'appuyant sur le déroulement de la tâche sous-jacente plutôt que sur celui du dialogue.

L'articulation entre conventionnel et intentionnel se fonde sur la notion de maturité d'un comportement. L'ouverture de comportements conventionnels par le système est déterminée par un mécanisme se servant de la maturité pour sélectionner le comportement le plus pertinent à jouer. Or la maturité d'un comportement est conditionnée par l'état d'activation de ses préconditions, celles-ci étant déterminées par la partie décisionnelle de l'agent.

Les états permettent une distinction claire entre les aspects conventionnels et les aspects intentionnels de la réalisation d'une tâche par le dialogue. Les aspects conventionnels sont capturés par l'utilisation de motifs dialogiques conventionnels auxquels sont associés des règles et qui sont regroupés en unités localement cohérentes au sein des tables d'état. D'un autre côté, les aspects intentionnels sont déterminés par l'utilisation de prédicats, qui capturent les raisonnements de l'agent. C'est la définition de ces prédicats qui définit le comportement de l'agent : des prédicats simples permettent de concevoir un agent réactif, alors que des prédicats plus développés permettent le développement d'un agent cognitif.

Dubuisson Duplessis [Dub14] présente dans ses travaux l'intérêt interprétatif des jeux de dialogue qui, en définissant un enchaînement attendu des actions dialogiques, aident à contextualiser un acte de dialogue joué au cours d'un dialogue. À une granularité plus élevée, les tables d'état ont aussi cet intérêt car les comportements attendus définis dans les tables d'état facilitent l'interprétation et la contextualisation des actions de l'utilisateur dans le cadre de la réalisation de la tâche. De même, les tables d'état ont un intérêt génératif car elles restreignent les possibilités d'interaction du système à celles considérées comme localement pertinentes et donnent des informations permettant au système de sélectionner le prochain coup dialogique à jouer.

Les tables d'état ainsi que les mécanismes reposant sur la maturité pour prendre des décisions permettent d'expliquer les décisions de l'agent, que ce soit en termes de contextualisation d'un coup dialogique de l'utilisateur, de transition entre les états ou d'émission d'un coup dialogique. Cette structuration de notre système ouvre la voie à un approfondissement de celui-ci dans la direction de *l'organisation explicable* [LMMC17]. Elle fournit assez d'informations pour permettre à l'agent de justifier les comportements qu'il

a eu au cours de l'interaction. En effet, le comportement du système est rendu explicable grâce aux préconditions des comportements réalisables et au mécanisme de sélection de ces derniers. Cependant les approfondissements liés à l'organisation explicable ne sont pas traités dans ce manuscrit.

Chapitre 4

Implémentation dans l'architecture CoCoA

« Il faut toujours plus de temps que prévu, même en tenant compte de la loi de Hofstadter. »

Douglas Hofstadter,
Gödel, Escher, Bach : Les Brins d'une Guirlande Éternelle

Sommaire

Dans ce chapitre, nous présentons l'architecture d'agent définie pour la mise en œuvre du modèle décrit dans le chapitre 3. Nous donnons notamment les contraintes liées à l'architecture d'un tel agent et les fonctionnalités attendues. Nous distinguons la *partie publique*, représentant le dialogue avec l'utilisateur, et la *partie privée*, permettant à l'agent de représenter l'état actuel de la tâche et de décider des actes de dialogue à émettre.

Nous introduisons CoCoA [1], notre modèle d'agent implémentant le modèle théorique donné dans le chapitre 3. Nous détaillons comment il s'articule avec le gestionnaire de dialogue choisi pour superviser sa partie publique et en donnons une représentation technique appuyée par des algorithmes.

La section 4.1 identifie les besoins fonctionnels d'un agent collaboratif et montre en quoi les systèmes existants ne sont pas satisfaisants. La section 4.2 présente l'architecture de CoCoA, une implémentation du modèle théorique décrit dans le chapitre 3, d'abord sous une perspective de haut niveau puis en détaillant ses composants principaux.

4.1 Caractérisation des besoins pour un agent collaboratif

Cette section décrit le cahier des charges nécessaire à l'implémentation de notre modèle. La partie 4.1.1 définit nos besoins fonctionnels et la partie 4.1.2 passe en revue les systèmes existants sous la perspective des besoins précédemment présentés.

4.1.1 Besoins fonctionnels

L'état de notre agent à un instant donné est représenté à l'aide d'un *état d'information*. La partie publique de cet état d'information est le *tableau de conversation*, qui représente l'état du dialogue. La cohérence de celui-ci tout au long de l'interaction est garantie par un *gestionnaire de dialogue*, qui met à jour la représentation courante du dialogue à chaque nouvelle action dialogique d'un interlocuteur.

Nous avons présenté en section 2.3.2 les différentes combinaisons possibles proposées par Ginzburg [Gin12] entre les parties publiques et privées de l'état d'information. Nous considérons dans notre modèle d'agent que la partie publique de l'état d'information est strictement partagée entre les interlocuteurs et que chacun dispose d'une partie privée qui lui est propre.

Le chapitre 3 décrit le modèle théorique de la *partie privée* de l'agent collaboratif que nous souhaitons implémenter. Cette partie privée doit permettre de représenter les prédicats calculés par l'agent, l'état courant de l'interaction et les maturités des comportements et des états.

Nous défendons l'idée selon laquelle il est profitable de séparer clairement les parties publiques et privés de notre agent, et ce pour trois raisons :

— du point de vue structurel, comme Grosz et Sidner [GS86] l'ont mis en avant, la structure intentionnelle du dialogue n'est *ni identique, ni isomorphe* à la tâche sous-jacente et donc tenter de représenter une tâche collaborative et le dialogue la

1. Pour « Commited Collaborative Agent. »

véhiculant au sein de la même structure n'est pas possible sans perdre des parties de l'un ou l'autre ;

— du point de vue de la conception, la réalisation d'un système de dialogue collaboratif dans un domaine particulier nécessite deux expertises très différentes : l'une sur le dialogue et l'autre sur le domaine de la tâche. Si la représentation du dialogue et de la tâche sont interdépendants, il est nécessaire pour toute personne souhaitant mettre en œuvre ce système de dialogue de maitriser ces deux expertises. À l'inverse, une distinction claire entre ce qui relève de la tâche et ce qui relève de l'interaction permet de déléguer chaque partie de la conception du système à un expert de chaque domaine ;

— du point de vue technique, le développement d'un système de dialogue collaboratif bénéficie d'une implémentation indépendante de chaque partie, une spécifiquement dédiée à la gestion du dialogue et l'autre à la réalisation de la tâche. Ce type d'architecture simplifie le cycle de développement de systèmes basés sur ce modèle en rendant certains de ses composants réutilisables d'un système à l'autre.

Nous souhaitons donc que l'architecture de notre agent fasse une distinction claire entre sa partie publique et sa partie privée. L'idée est d'isoler le problème de gestion du dialogue au sein du gestionnaire de dialogue (partie publique) et le problème de réalisation de la tâche au sein de la partie privée.

Nous détaillons maintenant les contraintes fonctionnelles qui s'appliquent pour l'implémentation de notre agent. Nous séparons nos besoins en deux parties : ceux qui s'appliquent à la partie publique et ceux qui s'appliquent à la partie privée de l'agent.

Partie publique Le tableau de conversation de l'agent doit posséder les caractéristiques suivantes :

Représentation de motifs dialogiques Ces motifs doivent pouvoir être utilisés pour une interprétation du comportement de l'utilisateur et une génération du comportement conventionnel de l'agent. De plus, il doit permettre de distinguer les issues possibles d'un motif dialogique, et notamment les possibilités de succès ou d'échec dialogique et extra-dialogique. Ces différentes issues sont nécessaires pour doter l'agent de caractéristiques collaboratives dès la définition de ses comportements réalisables, comme le fait de toujours accepter les requêtes de l'utilisateur en interdisant à l'agent de jouer un acte provoquant un échec extra-dialogique ;

Utilisation d'engagements sociaux Les motifs dialogiques doivent véhiculer une représentation des positions prises par les interlocuteurs sous la forme d'engagements sociaux (cf. section 2.3.3) ;

Indépendance vis-à-vis de la partie privée L'interprétation d'un coup dialogique par le gestionnaire de dialogue doit être réalisable en s'appuyant uniquement sur le contexte dialogique et sans dépendre de la tâche.

Partie privée La partie décisionnelle de l'agent doit posséder les caractéristiques suivantes :

Gestion de la tâche sans planification Notre agent ayant pour objectif d'aider à la résolution de problèmes confus et difficiles, une définition claire du problème et de sa résolution n'est pas possible à priori et ceux-ci peuvent évoluer au cours de l'interaction. De ce fait, nous avons pris le parti de ne pas nous fonder sur des approches par planification et de considérer que c'est l'interaction qui permet le déroulement de la tâche. La partie privée de notre agent doit donc pouvoir fonctionner sans définition explicite de buts ni de planification du dialogue ou de la tâche ;

Représentation de la tâche sous forme d'états Ces états doivent permettre de décrire le lien entre les motifs dialogiques utilisés pour l'interaction et la réalisation de la tâche. Ils doivent permettre l'usage d'engagements sociaux pour préciser les *effets contextuels* d'un motif dialogique. Les engagements que nous manipulons sont soit des engagements propositionnels (son créditeur s'engage au présent sur une proposition), soit des engagements en action (son créditeur s'engage sur l'occurrence d'une action dans le futur). Ces engagements peuvent être dans différents états, les engagements propositionnels pouvant être inactifs ou actifs, et les engagements en action pouvant être inactifs, actifs, satisfaits, échoués ou violés ;

De plus, l'interprétation d'un acte de dialogue de l'utilisateur doit tenir compte de l'état dans lequel se trouve la tâche au moment de l'occurrence de cet acte. Cela implique la capacité à lier de manière cohérente un acte de dialogue de l'utilisateur à un état de la tâche et à faire évoluer la représentation de l'état actuel de la tâche en conséquent.

Calcul de prédicats Les préconditions et prérequis présents dans les tables d'état sont formulés en utilisant des prédicats. Cet prédicats représentent les résultats des raisonnements de l'agent, réalisés en se basant sur l'état courant du dialogue et de la tâche, et sur des ressources extérieures propres au domaine ;

4.1.2 Systèmes de dialogue existants

La liaison étroite entre représentation, raisonnement et implémentation sur la plupart des systèmes actuels rend difficile leur comparaison ainsi que l'analyse de leurs propriétés [FA17]. De plus, de nombreux systèmes ne font pas de distinction claire entre la partie interactive (gestionnaire de dialogue) et la partie privée de leur agent, entremêlant ces deux entités [GTAP18], ce qui nous amène à nous en détourner.

Au vu des contraintes présentes sur la partie privée de l'état d'information de notre agent données en section précédente, nous rejetons à priori les approches utilisant de la planification pour la représentation du dialogue et de la tâche. Cela nous amène à nous détourner des systèmes comme COLLAGEN [RSL01] et Disco for Games [RS12] présentés en section 2.2.3 et RavenClaw [BR09]. Nous passons en revue les systèmes intéressants et mettons en avant leurs avantages et inconvénients au regard de nos besoins et contraintes de travail.

Approche par collaboration

Certains travaux [ABF02] se sont concentrés spécifiquement sur la conception d'un système de dialogue pour la résolution collaborative de problème entre un agent et un utilisateur. Cela a mené à l'élaboration du modèle de « Collaborative Problem Solving » [2] pour décrire le processus à suivre pour résoudre un problème. Celui-ci se compose de trois phases générales : déterminer les objectifs qui vont engendrer son comportement, déterminer les recettes permettant de progresser vers un objectif et finalement mettre en œuvre les recettes et évaluer leur succès [BA05]. Les recettes sont des plans pré-construits ou calculés à la volée afin d'atteindre un but. Ceux-ci sont composés *d'actes de plan* correspondant à des opérateurs sur la tâche qui sont véhiculés par des *actes d'interaction*, permettant de communiquer avec ses collaborateurs [ABF02]. De ce fait, l'utilisation des recettes constitue un bon candidat pour représenter des motifs dialogiques au sein de notre gestionnaire de dialogue, étant donné que les actes d'interaction peuvent être rapprochés de motifs dialogiques ayant un contenu sémantique (un acte de plan) correspondant à un opérateur sur la tâche. Ce modèle a récemment été implémenté sous le nom de Cogent à l'aide du système TRIPS [GTAP18], un système de planification collaborative (cf. section 1.3.1).

Approche QUD

Un gestionnaire de dialogue par état d'information a été réalisée par Larsson et Traum [LT00] selon l'approche QUD (les « questions under discussion », cf. section 2.3.2). Sa première implémentation a été faite en prolog sous le nom de TrindiKit, mais d'autres ont été implémentées par la suite. DIPPER est une réimplémentation du même modèle en Java [BKLO03]. `trindikit.py` a été codé en Python par Ljunglöf [Lju09].

L'approche par état d'information adoptée par ces systèmes en font de bons candidats pour servir de base au développement de notre agent. De plus, l'architecture de ces systèmes permet, comme nous le souhaitons, une gestion du dialogue indépendante de la tâche. Ce modèle a été utilisé avec succès pour l'implémentation de différents systèmes de dialogue [LLC+00, Lar02, LDC+12].

Approches par motifs d'interaction

L'approche du système Pamini [PW10] est orientée pour l'interaction humain-robot par le dialogue d'initiative mixte. Pamini fournit un ensemble de motifs d'interaction génériques basés sur des actes de dialogue permettant par exemple de réaliser des demandes d'action ou d'information. Ces motifs correspondent à ce que nous souhaitons employer comme base pour représenter les comportements possibles de notre agent.

Le gestionnaire de dialogue Dogma [Dub14, DPCK17] est basé sur une architecture avec un état d'information. Il intègre un tableau de conversation et les mécanismes nécessaires pour la mise en œuvre de motifs dialogiques sous la forme de jeux de dialogue de transfert d'information et de discussion d'action. Ces mécanismes font le lien entre

2. Traduisible par « résolution collaborative de problème ».

l'occurrence d'actes de dialogue et la création d'engagements sociaux propositionnels ou en action dans le tableau de conversation.

En plus des fonctionnalités interprétatives lui permettant de contextualiser un acte de dialogue dans la situation dialogique courante, DOGMA définit un ordre partiel sur les motifs dialogiques présents dans le tableau de conversation, permettant ainsi de leur attribuer un statut prioritaire ou non (les motifs dialogiques les plus prioritaires sont dits « saillants »). Cette structuration du dialogue donne au système des capacités prédictives, car elle permet de connaître les actes de dialogues attendus de la part de l'utilisateur dans l'état courant du tableau de conversation. Elle lui donne aussi des capacités génératives, le système ayant connaissance des actes de dialogue attendus de sa part au sein des motifs dialogiques saillants. DOGMA a de plus l'avantage d'être open-source et son implémentation est réalisée en scala [3].

Approche retenue

Le tableau 4.1 résume les réponses apportées par les systèmes existants à nos besoins fonctionnels. Ce tableau nous permet de choisir parmi les systèmes existants lequel est le plus adapté pour notre utilisation.

TABLEAU 4.1 – Synthèse des systèmes existants au regard des besoins fonctionnels de notre agent. ✓indique une compatibilité du système, ± une compatibilité nécessitant des adaptations et × une incompatibilité. ✓ ? indique une incertitude sur la compatibilité, les documents de référence étant trop succincts à ce sujet pour trancher.

	Cogent	TrindiKit	Pamini	DOGMA
Partie publique				
Motifs dialogiques	±	×	✓	✓
Engagements sociaux	±	×	×	✓
Indépendante de la partie privée	✓	✓	✓	✓
Partie privée				
Planification facultative	×	✓	✓ ?	×
Utilisation d'état	×	×	×	×
Calcul de prédicats	×	×	×	×

Cogent nécessite à priori la définition entre les interlocuteurs d'un *objectif* clair qui déterminera le comportement de l'agent. Il ne nous est donc pas possible de l'employer, une de nos contraintes étant que nous ne pouvons pas à priori établir un objectif.

TrindiKit et ses dérivés fonctionnant par une approche QUD et n'incluent donc pas le mécanisme de motifs dialogiques que nous souhaitons mettre en œuvre ni celui des engagements sociaux. De plus, leur développement est arrêté [4], ce qui laisse entrevoir des

3. Rendu disponible à l'adresse `https://gitlab.insa-rouen.fr/jlouvet/dogma`, visité le 29/11/2018.

4. Dernière contribution à trindikit.py le 16/04/2010, cf. `https://github.com/heatherleaf/py-trindikit`, visité le 26/11/2018.

problèmes de compatibilité, et le code n'est ni documenté ni complètement correct [5]. De ce fait, nous n'avons pas retenu ces systèmes pour l'implémentation de notre agent.

Cependant, les auteurs ne précisent pas s'ils font appel à une structure proche d'un état d'information et ne développent pas la manière dont est géré le dialogue, notamment si le système intègre une représentation structurée du dialogue, comme un tableau de conversation. De même, aucune structure formelle représentant des engagements réalisés par les interlocuteurs ne semble être présente au sein de ce système. En sus, l'orientation vers l'interaction avec des robots ajoute un degré de complexité important et non nécessaire dans notre cas. Nous avons donc préféré nous tourner vers une autre solution.

Aucun des systèmes présentés ci-dessus ne permettent l'implémentation de notre modèle que ce soit pour sa partie publique ou sa partie privée. DOGMA est un cas à part, celui-ci étant un gestionnaire de dialogue, il n'intègre aucune partie privée. Cependant, il répond à tous les besoins pour la partie publique de notre agent. Nous avons donc en définitive pris la décision d'utiliser DOGMA comme gestionnaire de dialogue et de le compléter par notre propre implémentation de la partie privée de l'agent développée dans le chapitre 3.

4.2 Le système CoCoA

Nous présentons maintenant l'architecture de CoCoA et notamment la structure de sa partie privée. CoCoA est le système que nous avons développé pour implémenter le modèle théorique décrit dans le chapitre 3. Celui-ci utilise DOGMA comme gestionnaire de dialogue pour sa partie publique et gère sa propre partie privée.

La section 4.2.1 donne l'architecture de CoCoA, en distinguant ce qui est délégué à DOGMA et donne la boucle principale de l'agent. La section 4.2.2 décrit les modules de gestion des prédicats et de l'état courant de CoCoA. Enfin, la section 4.2.3 présente le module chargé de décider du prochain coup joué par CoCoA.

4.2.1 Architecture de CoCoA

CoCoA étend la structure de DOGMA en intégrant différents modules conçus pour répondre aux besoins fonctionnels de sa partie privée identifiés dans la section 4.1.1.

Organisation de l'état d'information

L'architecture de notre agent s'organise autour de *l'état d'information* de Traum et Larsson [TL03] présenté en section 2.3.2, celle-ci est donnée en figure 4.1. La partie en gris clair représente le gestionnaire de dialogue DOGMA et la partie en noir représente la structure de CoCoA.

L'état d'information est divisé en deux : d'un côté il y a le *tableau de conversation*, qui représente l'état du dialogue à un instant donné. Il est supervisé par DOGMA et est

5. Cf. le `README` du projet trindikit.py : `https://github.com/heatherleaf/py-trindikit/blob/master/README.md`, visité le 26/11/2018.

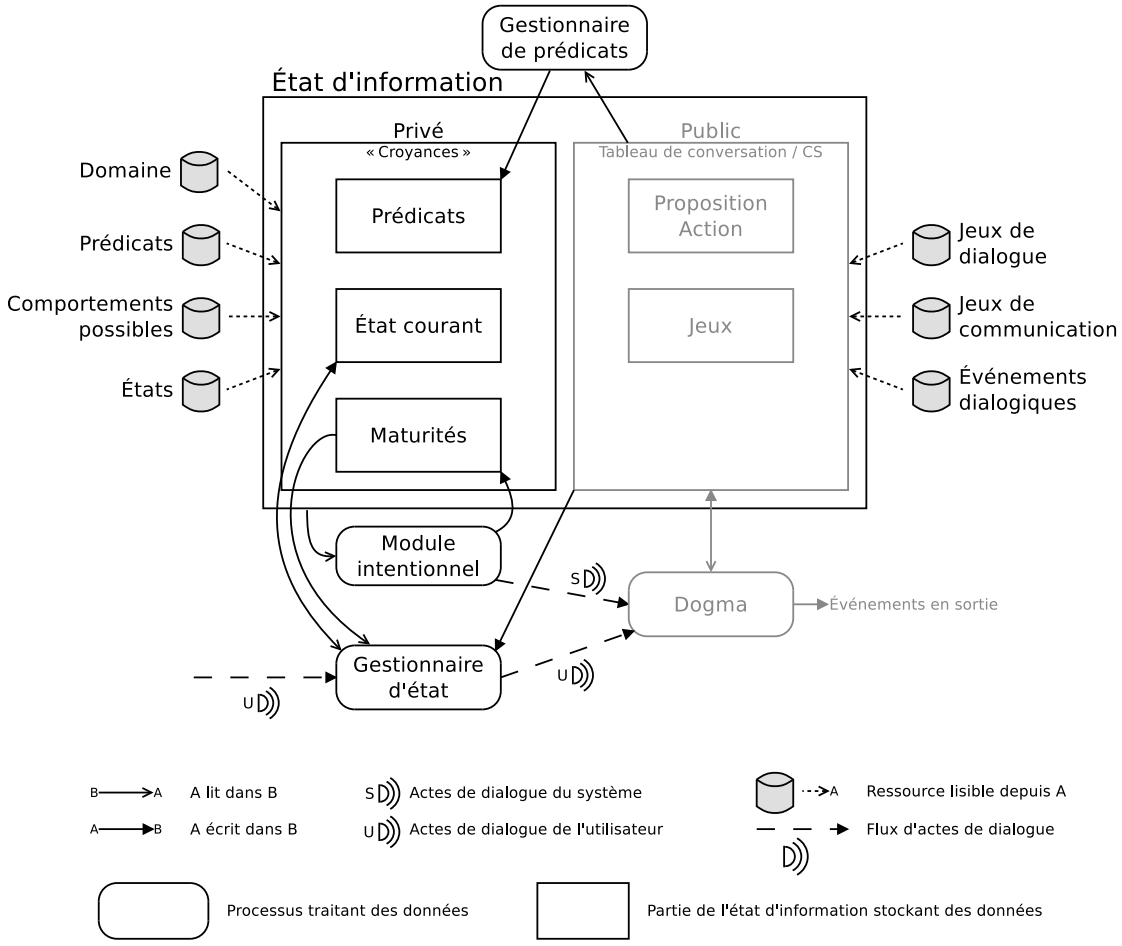

FIGURE 4.1 – Architecture technique de notre agent la partie grisée représente DOGMA et la partie en noir représente COCOA.

considérée comme strictement partagée entre l'agent et l'utilisateur. Il constitue la partie publique de l'état d'information. La partie privée de CoCoA n'y a qu'un accès en lecture.

D'un autre côté, il y a la *partie privée* de l'état d'information, qui contient les « croyances » de l'agent, c'est-à-dire toutes les informations qui lui sont propres et qu'il considère comme vraies pour représenter l'état actuel de la tâche. Celui-ci est supervisé par CoCoA, dont les modules y accèdent en lecture et en écriture. CoCoA est composé de trois modules : un gestionnaire de prédicats, un gestionnaire d'état et un module intentionnel. Ces modules traitent et mettent à jour les informations présentes dans son état d'information privé, c'est-à-dire les prédicats déduits, l'état courant de l'interaction et la valeur des maturités des états de la tâche.

La description des prédicats, des comportements possibles et des états de l'interaction est réalisée en dehors de la définition du fonctionnement de l'agent et est à la discrétion du concepteur de l'agent final. De la même manière, les jeux de dialogue, de communication et les événements dialogiques doivent être spécifiés dans les termes du gestionnaire de dialogue construit à partir de Dogma.

Nous décrivons maintenant comment s'organise CoCoA et comment sa partie privée se combine à Dogma.

Articulation entre la partie privée de CoCoA et Dogma

D'après Traum et Larsson [TL03], la gestion du dialogue est composée de quatre fonctions :

- mettre à jour le contexte dialogique à partir des communications des interlocuteurs ;
- avoir des capacités d'interprétation du comportement communicatif observé dépendantes du contexte ;
- s'interfacer avec le traitement de la tâche et le domaine d'application (base de données, gestionnaire de plans. . .) pour coordonner le comportement dialogique et non-dialogique ;
- décider du contenu sur lequel s'exprimer par la suite et quand l'exprimer.

Ces fonctions sont assumées par CoCoA et Dogma de façon partagée.

Mettre à jour le contexte dialogique C'est le rôle de Dogma en tant que gestionnaire de dialogue de veiller à la mise à jour de manière cohérente la structure du dialogue. Les actes de dialogue émis par CoCoA sont directement passés à Dogma pour être intégrés au dialogue, la cohérence dialogique de ceux-ci ayant été vérifiée avant leur émission. En ce qui concerne les actes de dialogue reçus de la part de l'utilisateur, ceux-ci sont pré-traités par la partie privée de CoCoA qui essaye de les rattacher de manière cohérente au contexte actuel de la tâche. C'est seulement si cette contextualisation est réussie qu'ils sont passés à Dogma pour être intégrés au dialogue. Si la contextualisation échoue, il sont simplement ignorés par l'agent. La partie privée de CoCoA a donc une influence indirecte sur le contexte dialogique en ayant la possibilité d'exclure des actes de dialogue de l'utilisateur si ceux-ci ne trouvent pas de place dans le contexte de la tâche.

L'algorithme 2 décrit plus loin présente la manière dont l'agent contextualise un acte de dialogue de l'utilisateur.

Contextualiser le comportement communicatif observé La contextualisation d'un coup dialogique par notre agent est réalisée à deux niveaux. Le premier niveau est celui de la tâche, que CoCoA va faire évoluer en contextualisant les coups dialogiques de l'utilisateur. Cette contextualisation est rendue possible par l'utilisation des prédicats et du concept de maturité dans la structuration de la tâche afin de décider de l'acceptabilité d'un coup dialogique de l'utilisateur dans le contexte actuel de la tâche. Si cette contextualisation du coup dialogique par rapport à la tâche réussit, elle rend possible sa contextualisation par rapport au dialogue. La structure des motifs dialogiques utilisés par Dogma lui donne des capacités *interprétatives* organisant la manière dont un acte de dialogue de l'utilisateur va être intégré à la représentation courante du dialogue.

S'interfacer avec le domaine La responsabilité des aspects propres au domaine est complètement dévolue à CoCoA. C'est au moment du calcul des prédicats que l'agent est capable d'activer ou non les prérequis des états et les préconditions des comportements réalisables. Le calcul de ces prédicats se base sur le contexte dialogique courant, les engagements contractés par les interlocuteurs, et les ressources du domaine d'application de la tâche auxquelles l'agent a accès. Une définition correcte de ces prédicats est donc primordiale pour permettre à l'agent de faire correctement le lien entre les actes de dialogue de l'utilisateur et la réalisation de la tâche.

Décider du prochain coup dialogique La décision du prochain coup à jouer se situe sur deux plans : le conventionnel, dépendant de Dogma, et l'intentionnel dépendant de CoCoA. Les capacités *génératives* de Dogma permettent à l'agent de savoir ce qui est conventionnellement attendu de sa part à n'importe quel moment du dialogue. La volonté de créer un agent collaboratif nous invite à donner la priorité aux actes conventionnellement attendus dans le contexte dialogique courant, afin de renforcer l'aspect coopératif du dialogue. C'est seulement si aucun acte n'est conventionnellement attendu que l'agent peut prendre l'initiative d'un comportement réalisable. Cette initiative est prise par CoCoA en fonction de l'état actuel de la tâche. L'algorithme 3 décrit plus loin illustre la manière dont l'agent articule le choix entre les coups attendus dans les contexte actuel et les initiatives qu'il peut prendre.

Boucle principale de l'agent

Afin de montrer l'organisation des concepts de maturité (section 3.2.1), de sélection d'un comportement (section 3.2.3) et de contextualisation d'un acte de dialogue de l'utilisateur (section 3.2.2), l'algorithme 1 décrit la boucle principale de l'agent. Nous verrons en section 4.2.3 l'algorithme de sélection du prochain coup à jouer par l'agent (algorithme 3) et l'algorithme de contextualisation d'un coup dialogique de l'utilisateur (algorithme 2).

Algorithme 1 : Boucle principale de l'agent

```
tant que vrai faire
    GestionnaireDePrédicats.MettreÀJourPrédicats();
    ModuleIntentionnel.MettreÀJourPrérequisEtPréconditions();
    ModuleIntentionnel.MettreÀJourMaturités() ; // Applique les définitions 3.1
     and 3.2
1   ProchainCoup ← ModuleIntentionnel.ChoisirProchainCoup() ; // Expliqué dans
     l'algorithme 3
2   si ProchainCoup ≠ NIL alors
3       JouerProchainCoup(ProchainCoup);
4   sinon
        // Attend que l'utilisateur fasse quelque chose
5       ad_u ← ObtenirActeDeDialogueUtilisateur() ;
        // Expliqué dans l'algorithme 2
6       GestionnaireDÉtat.ContextualiserActeDeDialogueUtilisateur(ad_u);
```

L'algorithme répète une boucle qui commence par faire appel au gestionnaire de prédicats pour mettre à jour ces derniers dans la partie privée de l'état d'information. C'est ensuite le module intentionnel qui est appelé pour mettre à jour les prérequis des états et les préconditions des comportements réalisables. Le calcul des prédicats se fait en utilisant l'état d'information actuel de l'agent ainsi que d'éventuelles ressources extérieures. La mise à jour des prérequis et préconditions est ensuite directe car ceux-ci ne dépendent que de la valeur des prédicats. Les nouvelles valeurs de maturité des comportements réalisables et des états (correspondant aux définitions 3.1 et 3.2) sont ensuite mises à jour par le module intentionnel. Le module intentionnel utilise ensuite ces valeurs pour décider du prochain coup à jouer. Si l'agent en trouve un, il le joue, incrémente le temps et répète la boucle. Sinon, il attend que l'utilisateur fasse un acte de dialogue, fait appel au gestionnaire d'états pour le contextualiser, incrémente le temps et répète la boucle.

La fonction `ObtenirActeDeDialogueUtilisateur()` (l.5) attend simplement que l'utilisateur émette un acte de dialogue. Un autre mécanisme, non présenté ici, décrit en section 3.3.1, est prévu pour solliciter l'utilisateur si celui-ci est inactif pendant un certain temps.

4.2.2 Gestionnaire de prédicats et gestionnaire d'état

Cette section détaille le fonctionnement du gestionnaire de prédicats et du gestionnaire d'état de CoCoA.

Gestionnaire de prédicats Le gestionnaire de prédicats est le module de CoCoA responsable du calcul des prédicats à partir de l'état courant du tableau de conversation et des ressources du domaine qu'il a à sa disposition. Ces prédicats sont calculés à partir des formules données par le concepteur du système final implémenté à partir de CoCoA. Comme montré dans l'algorithme 1, le gestionnaire de prédicats ré-évalue les prédicats à

chaque pas de temps de l'interaction et écrit l'ensemble des prédicats vrais dans la partie privée de l'état d'information.

Gestionnaire d'état Le gestionnaire d'état a pour but de réaliser la contextualisation des actes de dialogue de l'utilisateur dans le contexte actuel de la tâche et éventuellement d'ouvrir des comportements possibles et de mettre à jour l'état courant en conséquence. L'algorithme 2 décrit le processus de contextualisation d'un acte de dialogue de l'utilisateur.

Un acte de dialogue de l'utilisateur peut être attendu dans le contexte dialogique actuel. Dans cette situation (l. 5), l'agent récupère le motif dialogique associé attendant cet acte de dialogue (l. 6). Trois cas sont ensuite possibles :

— soit l'acte de dialogue de l'utilisateur correspond à une réponse à un comportement proposé par l'agent (il a joué l'acte *prop.in* du jeu de communication de contextualisation associé, cf. définition 3.6). En d'autres termes, l'agent a pris l'initiative d'un comportement possible et l'utilisateur a répondu à cette initiative. Dans ce cas, le comportement associé au motif dialogique passe dans l'état ouvert (cf. diagramme 2.3) et on change d'état dans le cas où l'état associé au motif dialogique est différent de l'état courant (l. 10).

— soit l'acte de dialogue de l'utilisateur ferme un motif dialogique ouvert (il correspond à un *prop.out* ou *acc.out* du jeu de communication de contextualisation associé). Dans ce cas, le comportement associé au motif dialogique est fermé (l. 12).

— soit l'acte de dialogue correspond à un acte interne à un motif dialogique ouvert, sans fermer celui-ci. Dans ce cas, aucune modification n'est apportée à l'état d'information privé.

Une fois ce traitement réalisé, l'acte de dialogue est envoyé à DOGMA pour être intégré au tableau de conversation (l. 13).

Si l'acte de dialogue n'est pas attendu dans le contexte dialogique actuel (l. 14 et suivantes), l'agent cherche dans l'état courant s'il est possible de trouver un comportement attendu ouvert par l'acte de dialogue de l'utilisateur (l. 14). S'il en trouve un, il l'ouvre et envoie l'acte de dialogue à DOGMA.

Si l'état courant ne permet pas de contextualiser l'acte de dialogue de l'utilisateur, l'agent se tourne vers les états candidats (l. 18). S'il existe au moins un état candidat permettant de contextualiser l'acte de dialogue de l'utilisateur, l'agent sélectionne celui dont la maturité est maximale et le désigne comme nouvel état courant (l. 19). Il récupère ensuite le motif dialogique de cet état attendant le coup dialogique de l'utilisateur (l. 20) et l'ouvre (l. 21), puis envoie l'acte de dialogue à DOGMA.

Si les états candidats ne permettent pas non plus de contextualiser le coup dialogique de l'utilisateur, l'agent applique la même stratégie mais en utilisant les états matures (l. 23 à 27).

Dans le cas où toutes les tentatives précédentes ont été infructueuses, le coup de l'utilisateur est ignoré (l. 29).

Il est important de noter qu'un changement d'état se fait toujours suite à une action de l'utilisateur. Celle-ci peut être en réaction d'une sollicitation de l'agent (cas où l'utilisateur

Algorithme 2 : Contextualisation d'un acte de dialogue de l'utilisateur

Entrées : ad_u : acte de dialogue de l'utilisateur.
Sorties : ∅

```
1  ÉtatCourant ← GestionnaireDÉtat.ÉtatCourant ;
2  ComportementsOuverts ← GestionnaireDÉtat.ComportementsOuverts ;
3  ÉtatsCandidats ← GestionnaireDÉtat.ÉtatsCandidats ;
4  ÉtatsMatures ← GestionnaireDÉtat.ÉtatsMatures ;
   // Si l'acte de dialogue est attendu dans un comportement possible ouvert ou
      proposé par l'agent
5  si ComportementsOuverts.Attend(ad_u) alors
6  |  CP ← ComportementsOuverts.ObtenirComportementPossibleDestination(ad_u);
   |  // Si l'utilisateur répond à un comportement proposé par l'agent
7  |  si CP.EstOuvertPar(ad_u) alors
8  |  |  GestionnaireDÉtat.Ouvrir(CP);
9  |  |  si CP.Comportement.État ≠ ÉtatCourant alors
10 |  |  |  GestionnaireDÉtat.NouvelÉtatCourant(CP.État);
11 |  si CP.EstFerméPar(ad_u) alors
12 |  |  GestionnaireDÉtat.Fermer(CP);
13 |  Dogma.NouvelActeDeDialogue(ad_u);
   // Si l'acte de dialogue ouvre un comportement attendu de l'état courant
14 sinon si ÉtatCourant.Attend(ad_u) alors
15 |  CP ← ÉtatCourant.ComportementAttenduOuvertPar(ad_u);
16 |  GestionnaireDÉtat.Ouvrir(CP);
17 |  Dogma.NouvelActeDeDialogue(ad_u);
   // Si l'acte de dialogue ouvre un comportement attendu dans un état candidat
18 sinon si ÉtatsCandidats.Attend(ad_u) alors
19 |  ÉtatCourant ← ÉtatsCandidats.ÉtatsAttendant(ad_u).PlusGrandParMaturité();
20 |  CP ← ÉtatCourant.ComportementAttenduOuvertPar(ad_u);
21 |  GestionnaireDÉtat.Ouvrir(CP);
22 |  Dogma.NouvelActeDeDialogue(ad_u);
   // Si l'acte de dialogue ouvre une comportement attendu dans un état mature
      dont les prérequis ne sont pas satisfaits
23 sinon si ÉtatsMatures.Attend(ad_u) alors
24 |  ÉtatCourant ← ÉtatsMatures.ÉtatsAttendant(ad_u).PlusGrandParMaturité();
25 |  CP ← ÉtatCourant.ComportementAttenduOuvertPar(ad_u);
26 |  GestionnaireDÉtat.Ouvrir(CP);
27 |  Dogma.NouvelActeDeDialogue(ad_u);
   // En dernier ressort, l'agent ignore l'acte de dialogue de l'utilisateur
28 sinon
29 |  IgnorerActeDeDialogue(ad_u);
```

répond à un comportement proposé), mais c'est toujours suite à un acte de dialogue de l'utilisateur qu'une transition entre état est réalisée. Ce mécanisme a été délibérément conçu de cette manière pour laisser à l'utilisateur la main sur l'interaction et le déroulement

de la tâche. De cette manière l'agent peut proposer des évolutions mais ce sont, en définitive, les décisions de l'utilisateur qui priment.

Notons ici que c'est le gestionnaire d'état qui réalise l'accommodation (cf. section 3.2.2) de l'agent à une action de l'utilisateur. En effet, celui-ci peut faire évoluer la partie privée de l'état d'information en ouvrant un comportement attendu ou en changeant l'état courant afin de permettre la contextualisation correcte d'un acte de dialogue de l'utilisateur.

4.2.3 Module intentionnel

Ce module incarne la partie décisionnelle de l'agent. C'est à lui que revient la tâche de choisir, selon les méthodes spécifiées dans la section 3.2, le prochain coup dialogique à jouer en fonction du contexte dialogique, de l'état courant et des valeurs de maturité présentes dans la partie privée de l'état d'information. L'algorithme 3 décrit le processus décisionnel mis en œuvre par l'agent pour choisir son prochain coup dialogique.

Algorithme 3 : Choix de son prochain coup par l'agent

Entrées : ∅
Données : TdC : tableau de conversation.
Sorties : CoupChoisi : prochain coup choisi par l'agent.

```
// Récupération des coups attendus de la part de l'agent dans les
   comportements ouverts
CoupsAttendus ← TdC.ObtenirCoupsAttendusSystème();
si CoupsAttendus.NonVide() alors
    CoupChoisi ← SélectionnerMeilleurCoup(CoupsAttendus);
    Dogma.NouvelActeDeDialogue(CoupChoisi);
sinon
    si SystèmePeutPrendreInitiative() alors
        ÉtatCourant ← GestionnaireDÉtat.ÉtatCourant ;
        ÉtatsCandidats ← GestionnaireDÉtat.ÉtatsCandidats ;
        CompCandidats ← (ÉtatsCandidats ∪ ÉtatCourant).ComportementsMatures();
        si CompCandidats.NonVide() alors
            CoupChoisi ← CoupChoisi.PlusGrandParMaturité().CoupDInitiative ;
            Dogma.NouvelActeDeDialogue(CoupChoisi);
        sinon
            CoupChoisi ← NIL ;
    sinon
        CoupChoisi ← NIL ;
Retourner(CoupChoisi);
```

Le choix du prochain coup dialogique se fait selon deux critères. Tout d'abord, l'agent vérifie si des coups dialogiques sont attendus de sa part dans le contexte dialogique actuel, la fonction `ObtenirCoupsAttendusSystème()` faisant appel à la capacité générative de DOGMA sur le tableau de conversation. Si c'est le cas, l'agent choisit le meilleur de ces coups à l'aide de la fonction `SélectionnerMeilleurCoup()`, l'envoie à DOGMA pour qu'il

soit intégré au tableau de conversation et le retourne. L'écriture de cette fonction est laissée à la discrétion du concepteur de l'agent final. En effet différentes stratégies de sélection peuvent être mises en œuvre et peuvent modifier le comportement de l'agent (choisir le coup s'intégrant dans le motif dialogique le plus ancien ou le plus récent, par exemple).

Dans le cas où rien n'est conventionnellement attendu de la part de l'agent, celui-ci vérifie dans un premier temps si il est conventionnellement acceptable qu'il prenne une initiative (c'est-à-dire qu'il n'a pas déjà un nombre trop important de comportements ouverts ou proposés, cf. section 3.2.3), avec la fonction `SystèmePeutPrendreInitiative()`. Si il est acceptable que l'agent prenne une initiative, il sélectionne le comportement de plus grande maturité parmi ceux de l'état courant et des états candidats (l. 9 et 11), l'envoie à DOGMA pour qu'il soit intégré au tableau de conversation et le retourne.

S'il n'est pas conventionnellement acceptable que l'agent prenne une initiative ou qu'il ne trouve aucun comportement mature à proposer au sein de l'état courant et des états matures, cet algorithme renvoie un coup vide, ce qui revient à ne rien faire.

4.3 Conclusion

Nous avons présenté dans ce chapitre l'implémentation du modèle théorique du chapitre 3 dans COCOA. Dans un premier temps nous avons caractérisé les besoins pour cette implémentation et décrit les systèmes existants pouvant prétendre à être utilisé pour cette implémentation. Nous avons retenu le gestionnaire de dialogue DOGMA pour l'implémentation de la partie publique de notre modèle et, aucun système existant ne remplissant le cahier des charges, nous avons implémenté sa partie privée.

La partie privée de COCOA ajoute trois modules en complément de DOGMA pour mettre en œuvre le modèle : un gestionnaire de prédicats, un gestionnaire d'état et un module intentionnel permettant à l'agent de décider du prochain coup dialogique qu'il doit jouer. Nous avons notamment vu que c'est le gestionnaire d'état qui réalise l'accommodation de l'état d'information de l'agent lors de l'occurrence d'un coup dialogique de l'utilisateur. La description de COCOA donnée dans ce chapitre laisse délibérément de côté toute application sur une tâche collaborative humain-machine, un application à une tâche de recherche d'information collaborative étant développée dans les chapitres suivants.

COCOA a été implémenté en Scala (version 2.11) [OSV08], afin de permettre une compatibilité optimale avec DOGMA, lui aussi implémenté en Scala. L'implémentation de COCOA est sous licence GNU GPLv3 et est disponible sur la plate-forme d'hébergement de sources de l'INSA Rouen Normandie, à l'adresse `https://gitlab.insa-rouen.fr/jlouvet/cocoa`.

Troisième partie

Mise en application

Chapitre 5

Application à une recherche d'information assistée

« Bien qu'il connaisse son objectif, le chemin pour y parvenir n'est pas toujours celui qu'il imaginait. »

Paulo Coelho, Le manuel du guerrier de la lumière

Sommaire

Ce chapitre illustre l'utilisation du modèle d'interaction collaborative développé dans le chapitre 3 pour une tâche de recherche d'information collaborative dans le domaine médical. Le but est de montrer comment ce modèle s'applique à un problème réel et en quoi les différents outils disponibles s'adaptent à des besoins observés dans des cas concrets.

Nous nous basons sur l'étude d'un corpus humain-humain afin de modéliser le processus de résolution de problème pour ensuite extraire les propriétés importantes de ce processus et le transposer sous la forme de comportements possibles regroupés au sein de tables d'état.

En section 5.1, nous décrivons les spécificités de la recherche d'information collaborative médicale. Nous y présentons aussi le corpus Cogni-CISMeF, sur lequel se base notre scénario d'interaction humain-humain de recherche d'information collaborative médicale. Les sections suivantes présentent les différentes phases de ce scénario ré-écrites en utilisant le formalisme du chapitre 3 : la section 5.2 pour les phases d'ouverture, de verbalisation et de construction d'une première requête, la section 5.4 pour la phase d'évaluation des résultats et la section 5.5 pour la phase de réparation de la requête. La section 5.6 détaille les niveaux de modularité apportés par notre modèle. Nous terminons en section 5.7 en présentant les propriétés de la RICM conservées par notre modèle et les limites que cette application a mises en lumière.

5.1 Exemple humain-humain de recherche d'information collaborative médicale

Cette section donne les caractéristiques de la recherche d'information collaborative médicale (RICM). Notre modélisation de la RICM humain-humain s'appuie sur les travaux ayant été réalisés lors du projet Cogni-CISMeF (PI CNRS TCAN 2004-2006).

Dans un premier temps (section 5.1.1), nous présentons la collecte d'un corpus de RICM humain-humain. Nous donnons ensuite (section 5.1.2) le scénario de RICM issu de l'étude de ce corpus. Enfin (section 5.1.3) nous discutons des modifications qui seront apportées à ce scénario par le passage à une interaction humain-machine.

5.1.1 Le corpus Cogni-CISMeF

Le projet CISMeF

Le projet CISMeF (Catalogue et Index des Sites Médicaux de langue Française [CIS]), a été initié par le Centre Hospitalier Universitaire de Rouen en 1995. Il indexe les principaux sites et documents francophones d'information de santé, avec plus de 123 000 sites et documents référencés au 13/09/2018. La terminologie CISMeF permettant l'indexation des ressources est structurée en quatre niveaux hiérarchiques : les *méta-termes*, correspondant à des spécialités biologiques ou médicales, les *mots-clés*, basés sur une terminologie dérivée du MeSH (« Medical Subject Headings », un système de méta-données médicales), les *qualificatifs*, une hiérarchie associée à chaque mot-clé et les *types de ressources*, décrivant la nature des informations véhiculées. Ce projet a permis le lancement en 2000 de

Doc'CISMeF, un moteur de recherche associé à ce catalogue de ressources [DLD+01].

Le projet Cogni-CISMeF

L'utilisation de CISMeF par un utilisateur quelconque est entravée par le fait que le langage de requête est complexe et qu'une maîtrise de la terminologie CISMeF est nécessaire pour faire des recherches efficaces [Loi08]. Pour pallier cette difficulté, le projet Cogni-CISMeF a vu le jour dans le but de concevoir un agent assistant dont la fonction est de faciliter l'accès à ce service [CDH+10]. Un corpus d'interaction humain-humain a été collecté pour servir de base au développement de cet agent. Il est constitué de dialogues d'assistance sur une tâche de RICM utilisant le moteur de recherche de CISMeF. Si l'on se rapporte à la taxonomie de Golovchinsky [GPB09, GQPG09], présentée en section 1.4.2, le type de RIC du corpus Cogni-CISMeF est *d'intention explicite*, *synchrone* et *co-localisé*.

Au cours de ces dialogues, deux rôles se distinguent : un *utilisateur* ayant un besoin d'information médicale, et un *expert* CISMeF aidant l'utilisateur à trouver des ressources pertinentes. L'expert dispose d'un accès à CISMeF et mène la recherche en collaboration avec l'utilisateur. Afin d'avoir le plus d'informations sur le déroulement de l'interaction et l'utilisation de CISMeF, l'expert avait pour instruction de verbaliser le plus possible ses actions. Deux membres du projet ont joué le rôle d'experts et 21 membres du laboratoire ont joué celui d'utilisateurs. Ces derniers ne présentaient aucune expertise du domaine médical. 21 dialogues (9 pour un expert et 12 pour l'autre) ont été enregistrés et retranscrits au sein de ce corpus pour un total de 1 800 échanges représentant 37 000 mots. Un exemple de dialogue extrait du corpus est donné dans le tableau 5.1.

A1	[...] bah peut-être qu'on peut essayer d'élargir la recherche dans ce cas là si on regarde un petit peu les mots qu'on a mis /
B2	on n'a quand même pas mis grand chose
A3	bah non alors
B4	pourquoi enlever / on peut enlever analyse
A5	alors enlevons analyse
B6	et diagnostic
A7	oui
[...]	
A8	[...] j'aurais presque envie de mettre diagnostic quand même parce que / parce que on va voir ce que ça donne
B9	oui normalement c'est un diagnostic / ok / essayons comme ça
A10	on va essayer comme ça sinon on enlèvera encore des choses pour arriver à avoir des / donc je relance la recherche avec l'accès thématique cancéro le mot clé cismef colon et puis le qualificatif diagnostic sans précision du type de ressource qu'on recherche

Tableau 5.1 – Extrait d'un dialogue du corpus (VD06). A est l'expert et B est le demandeur.

Les travaux de Loisel [Loi08, LKC08] à partir de ce corpus ont donné lieu à un premier modèle d'agent assistant pour une tâche de RI sur CISMeF, basé sur GoDiS [LLC+00]. Ce prototype a souffert des limitations de GoDis sur l'extension à une tâche complexe et

l'ajout de coups dialogiques et n'a pas été développé plus en profondeur [Dub14]. Ce corpus a par la suite été annoté à l'aide du schéma multifonctionnel DIT++ (cf. section 2.1.4) puis utilisé par Dubuisson Duplessis pour le modèle de jeu de dialogue de DOGMA présenté en section 2.4.2 [Dub14].

5.1.2 Scénario de RICM

L'annotation réalisée à l'aide de DIT++ sur le corpus COGNI-CISMEF montre que la dimension principale du dialogue est *Task* (actes contribuant à l'avancée de l'activité sous-jacente au dialogue), qui représente 68.30% des fonctions annotées. Les quatre dimensions suivantes sont *Time Management* (9.93%), *Auto-Feedback* (9.44%), *Own Communication Management* (5.31%) et *Turn Management* (2.76%). Ces quatre dimensions correspondent respectivement aux actes de dialogue signalant que le locuteur a besoin de temps pour formuler sa contribution, aux actes de dialogue apportant des informations sur le traitement par le locuteur de l'énoncé précédent, aux actes de dialogue indiquant que le locuteur modifie sa contribution courante et aux actes de dialogue permettant de s'accorder sur le rôle de locuteur.

Nous nous focalisons sur la dimension *Task* et laissons de côté les autres. Cette décision est motivée par le fait que notre objectif est de structurer l'enchaînement des motifs dialogiques contribuant directement à la réalisation de la tâche sous-jacente à l'interaction. Les autres dimensions se rapportent à des phénomènes principalement propres à l'interaction humain-humain orale et leur intégration dans notre modèle ne nous semble pas primordiale.

L'analyse du corpus a permis d'identifier et de caractériser les différentes phases de la tâche de RICM, qui jouent chacune un rôle spécifique dans l'avancement de la tâche [LDC+16, LDC+17]. Nous avons distingué cinq phases :

La verbalisation – établissement du sujet de la recherche entre les deux interlocuteurs (utilisateur et expert). L'utilisateur commence par exprimer son besoin d'information à l'expert. Après une première formulation, il peut éventuellement y ajouter des précisions spontanées. En réaction, l'expert peut demander des précisions s'il juge la formulation initiale pas assez précise ou s'il ne l'a pas comprise. Il peut aussi tenter de la reformuler afin de faire valider sa compréhension par l'utilisateur. S'il considère que la formulation est suffisante, ou que l'utilisateur n'a rien à y ajouter, l'expert commence la construction de la requête ;

La construction de la requête – recherche de termes dans la terminologie CISMEF correspondant à la verbalisation de l'utilisateur. Il s'agit pour les interlocuteurs de trouver de façon collaborative un alignement entre la terminologie métier et le vocabulaire usuel de l'utilisateur pour ensuite remplir le formulaire de recherche ;

Le lancement de la requête – exécution de la requête courante par l'expert. Cette phase est souvent réalisée de manière implicite, c'est-à-dire sans être verbalisée ;

L'évaluation des résultats – cette phase commence par l'évaluation explicite par l'expert des résultats retournés par la requête. Si ceux-ci ne lui semblent pas satisfaisants, il décide de réparer la requête sans attendre les retours de pertinence de l'utilisateur.

Si les résultats lui semblent satisfaisants, il les présente à l'utilisateur. Si celui-ci les juge aussi satisfaisants, le but est atteint et la recherche se termine, le besoin d'information étant comblé. Si l'utilisateur les juge partiellement satisfaisants, c'est-à-dire non adaptés à son profil ou n'abordant pas tous les thèmes de son besoin d'information, ou insatisfaisants, il faut réparer la requête. L'utilisateur peut aussi abandonner la recherche suite à une évaluation négative ;

La réparation de la requête – l'expert et l'utilisateur tentent de mettre en place des tactiques pour modifier la requête tout en respectant le besoin d'information de l'utilisateur. Trois tactiques ont été observées : la *précision*, affinant la requête en y ajoutant des mots-clés ou en les rendant plus spécifiques, *la généralisation*, simplifiant la requête en supprimant certains de ses mots-clés ou en les remplaçant par d'autres plus génériques, et *la reformulation*, pour une requête ni trop précise ni trop générale mais qui ne renvoie pas de documents satisfaisants pour l'utilisateur. Cependant, ces tactiques ne sont pas mutuellement exclusives : il est possible de combiner une précision ou une généralisation avec une reformulation. L'extrait de dialogue présenté dans le tableau 5.1 est un exemple de réparation de requête.

En plus de ces cinq phases, une phase d'ouverture et de clôture ont été observées. La phase d'ouverture est facultative et consiste en de simples salutations. Enfin, la phase de clôture peut faire apparaître des propositions d'une nouvelle recherche.

Le scénario représenté par la Figure 5.1 synthétise les phases ainsi que leurs enchaînements possibles. Les états en pointillés correspondent aux actions qui peuvent être réalisées de manière implicite, c'est-à-dire sans verbalisation des participants à l'interaction. La boucle lancement/évaluation/réparation est bien présente.

Différentes propriétés de la RI et de la RIC ont été mises en lumière au sein de ce corpus. Notamment, le processus de résolution de problème mis en œuvre par les participants est un processus itératif, opportuniste, stratégique et interactif [Dub14], selon la description que nous avons donné des processus RI et de RIC en section 1.4.2 :

Itératif – nécessite de répéter le même processus pour combler le besoin d'information de l'utilisateur par raffinage successif [SE98, MW07]. Le besoin d'information de l'utilisateur n'est pas fixe mais évolue lorsqu'il découvre de nouvelles ressources. Cet aspect est illustré par la répétition du motif lancement/évaluation/réparation de la requête, représenté par une boucle dans le graphe du scénario présenté en figure 5.1 ;

Opportuniste – la découverte de certaines ressources peut modifier la perception de l'utilisateur de son besoin d'information et mener la recherche dans une direction *inattendue*. C'est ce que Bates décrit dans son modèle de la *cueillette de baies*, qui décrit la recherche d'information comme un processus dynamique évoluant selon les ressources découvertes au cours de son déroulement [Bat89]. Cet aspect opportuniste rend l'interaction difficilement prédictible et donc difficilement planifiable a priori ;

Stratégique – en accord avec les observations de Bates [Bat79, Bat90] (cf. section 1.4.1), différents niveaux de stratégies ont été observés dans le corpus, correspondant aux *coups*, *tactiques*, *stratagèmes* et *stratégies* permettant de faire avancer la tâche de RI ;

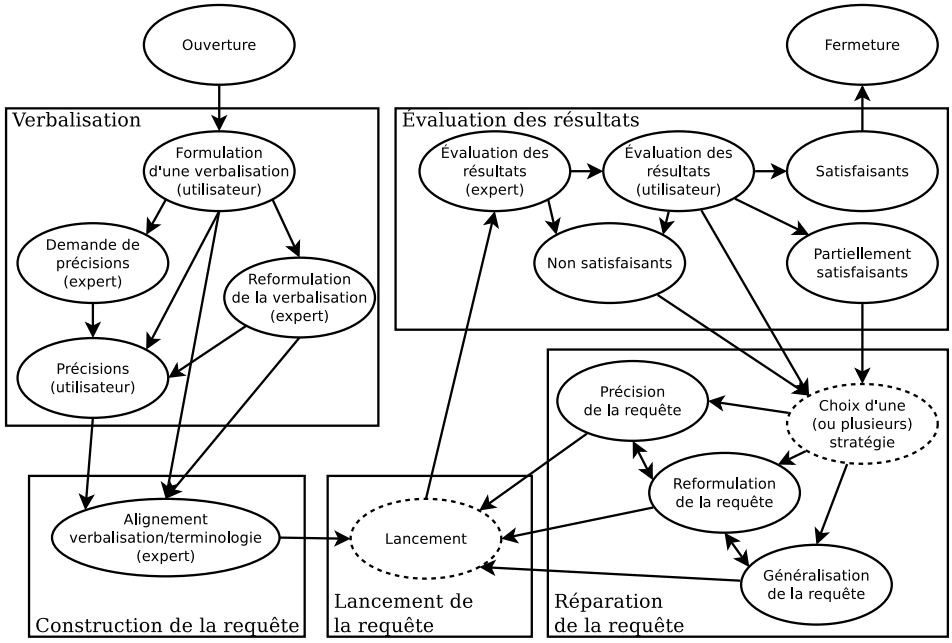

FIGURE 5.1 – Scénario issu de l'analyse du corpus représentant les enchaînements des phases de la tâche de RICM.

Interactif – la recherche d'une réponse au besoin d'information de l'utilisateur nécessite de nombreux échanges entre celui-ci et l'expert qui vient l'aider. Au cours de ces échanges, les participants à la recherche s'influencent mutuellement pour faire avancer la tâche, correspondant à notre définition d'une interaction (cf. définition 1.1, en section 1.1).

5.1.3 Comparaison entre la RICM humain-humain et humain-machine

Nous discutons maintenant des modifications que le passage d'une interaction humain-humain à une interaction humain-machine va apporter. Dans la conception des états correspondant au scénario de RICM, nous ne cherchons pas à avoir une approche *par émulation*, mais une approche *par complémentarité*, qui reconnaît l'asymétrie entre les humains et les machines et qui en tire partie pour développer des modes d'interaction et de collaboration [Ter95, Fis01, Suc07]. La structure de la tâche humain-machine s'en retrouve donc modifiée par rapport à celle humain-humain présentée en section 5.1.2.

Nous discutons dans la suite de cette section des divergences structurelles du modèle humain-machine vis-à-vis du modèle humain-humain. Nous justifions ensuite le changement de proportion des fonctions de discussion d'action par rapport aux fonctions de transfert d'information.

Divergences structurelles

Notre objectif lors du passage à une représentation humain-machine n'est pas de copier l'interaction humain-humain mais d'en tirer les propriétés essentielles pour aider l'utilisateur à réaliser sa tâche. L'interaction se trouve *de facto* modifiée sur trois aspects :

— le passage d'un interlocuteur humain à un agent collaboratif amène un changement des compétences partagées par les interlocuteurs. Par exemple, l'agent est capable de lancer des requêtes « en privé », c'est-à-dire sans le faire savoir à l'utilisateur, afin de les évaluer et de prendre des décisions avant d'interagir avec l'utilisateur. De ce fait, l'évaluation des résultats par l'expert, qui était réalisée explicitement lors de l'interaction humain-humain, sera réalisée implicitement lors de l'interaction humain-machine, et ne fera donc pas l'objet d'un état (cf. section 5.4.1) ;

— les différentes stratégies de réparations sont fusionnées en une seule étape. En effet, s'il est pertinent de distinguer spécifiquement les trois stratégies qui peuvent être employées pour réparer la requête lors de l'étude de la tâche de recherche d'information, cette distinction n'est plus nécessaire lorsque l'on se positionne au niveau de l'interaction et un seul état regroupera ces trois stratégies ;

— la situation de collaboration n'est pas la même. Dans le corpus Cogni-CISMeF, c'est l'expert CISMeF qui a l'ascendant sur l'interaction et qui mène la recherche, l'utilisateur observant ses actions. Dans le cas humain-machine, nous préférons laisser la priorité à l'utilisateur, en considérant que ses décisions prévalent sur celles de la machine. Cela se retrouve à la fois dans le modèle décisionnel de l'agent présenté en section 3.2 et dans la définition des états représentant le scénario de RICM humain-machine donné dans les sections 5.2, 5.4 et 5.5. Nous appliquons ici la règle donnée par Lieberman qui suggère, dans le cadre de la conception d'un agent autonome d'interface, de *proposer plutôt que d'agir* [Lie97].

De manière à laisser l'ascendant à l'utilisateur sur l'interaction et la réalisation de la tâche, le système ne prendra pas de décision à la place de celui-ci et ne sera capable que de lui faire des propositions. C'est dans cette optique que les règles des états sont construites, pour tenir compte de la part d'implicite dans le comportement de l'utilisateur et faire en sorte que ce soit le système qui suive les actions de l'utilisateur et non l'inverse.

Proportion transfert d'information/discussion d'action

Le passage à une interaction humain-machine provoque un important remaniement de la structure de la tâche et de l'enchainement de ses états. En effet, le fonctionnement intrinsèque de la machine entraîne la suppression d'états comme celui d'évaluation des résultats d'une requête par l'expert, qui est maintenant réalisé implicitement et avant le lancement de la requête.

Cela nous amène à prendre nos distances avec certains aspects du corpus, notamment au niveau de l'utilisation des fonctions de transfert d'information de la taxonomie DIT++. Dans le corpus, celles-ci sont largement majoritaires et représentent 68.4% des fonctions dialogiques employées par les interlocuteurs dans la dimension *Task*. À l'inverse, nos tables

d'état présentent principalement des motifs de discussion d'action (Request ou Offer par exemple), qui ne comptent que pour 31.6% des fonctions dialogiques de la dimension *Task* du corpus. Cette sous-représentation des fonctions associées à des discussions d'action est justifiée sur plusieurs aspects :

— premièrement, dans le corpus, les interlocuteurs étant co-localisés, ils peuvent utiliser d'autres canaux que la communication verbale pour interagir et notamment pour signifier leur approbation à une proposition d'action. Ces phénomènes n'étant pas verbalisés, il n'ont pas été pris en compte dans l'étude qui a été faite du corpus.

— deuxièmement, dans le corpus, l'expert a l'ascendant sur l'interaction, ce qui lui permet de supposer un accord implicite de son interlocuteur lorsqu'il suggère une action.

Cela se traduit dans le corpus par un bien plus grand nombre de fonctions initiatives (*suggestion*, *offer* ou *request* par exemple), que de fonctions réactives (*acceptSuggestion*, *declineSuggestion*, *acceptOffer*, *declineOffer*, *acceptRequest* ou *declineRequest* par exemple) [Dub14].

Le nombre important de fonctions fournissant de l'information [1] a plusieurs causes [Dub14] :

— lors de la collecte du corpus, les interlocuteurs étaient incités à verbaliser toutes leurs actions, alourdissant la proportion de fonctions fournissant de l'information ;

— la méthode de segmentation des dialogues du corpus a mené à un découpage qui peut associer plusieurs fois à un énoncé la même fonction fournissant de l'information ;

— l'évaluation des résultats par l'expert contenait beaucoup d'interventions informatives. Cependant, comme expliqué plus haut, cette partie du scénario ne sera pas retranscrite sous forme d'état ;

— l'utilisation de fonctions de transfert d'information est courant dans le dialogue oral pour maintenir la cohérence de l'environnement partagé de la tâche entre les interlocuteurs.

5.2 Ouverture, verbalisation et construction

Cette section ainsi que les trois suivantes (5.3, 5.4 et 5.5) présentent l'application de notre modèle formel de la collaboration (cf. section 3.1.2) à la recherche d'information collaborative médicale présenté en section 5.1.2. La traduction de chaque état du scénario humain-humain en une table d'état est discutée. Chaque table construite à partir du scénario est accompagnée des explications nécessaires à sa compréhension et de la description des prédicats qu'elle contient. Comme discuté en section 5.1.3, un état présent dans le scénario humain-humain n'est pas systématiquement transformé en table d'état et peut donc ne pas apparaître dans le scénario humain-machine.

Nous décrivons dans un premier temps les phases d'ouverture, de verbalisation et de construction de la requête à l'aide des tables d'état d'ouverture (table 5.1), de formulation

1. Une spécialisation des fonctions de transfert d'information, comptant pour un peu plus de 60% de la dimension *Task* et très largement représenté par la fonction *inform*.

d'une verbalisation (table 5.2), de reformulation de la verbalisation (table 5.3), de demande de précision (table 5.4), de précision de la verbalisation (table 5.5) et d'alignement verbalisation/terminologie (table 5.6).

Conventions d'écriture Nous utilisons x pour représenter l'utilisateur et y pour le système. Nous considérons i comme le temps courant (T_i représente la version actuelle du tableau de conversation) et j le temps auquel le comportement concerné se ferme (utilisé principalement pour l'écriture de la partie « effets » des comportements).

Les préconditions et prérequis sont définis à l'aide de prédicats, vus comme des déductions faites par l'agent sur l'état de l'interaction. De cette manière, on ne se préoccupe pas de leur définition lors de l'écriture des états, mais simplement de leur sémantique. Une séparation claire est faite entre leur utilisation dans les états et la manière dont ils sont calculés. Ils font le lien entre l'état de l'interaction et le déclenchement d'un comportement dialogique. Leur définition formelle est donnée dans l'annexe A.

Nous faisons la distinction entre les prédicats sur lesquels portent les engagements propositionnels et les prédicats dont la valeur est calculée par le système. Les prédicats déduits commencent par une majuscule et ceux sur lesquels portent les engagements par une minuscule. Par exemple le prédicat verbComplete est utilisé pour l'engagement de l'utilisateur sur le fait qu'il a terminé sa verbalisation, on le retrouve donc dans l'engagement C(x, verbComplete, **Crt**). Le système va se servir de cet engagement pour déduire le fait que la verbalisation de l'utilisateur est complète, représenté par le prédicat VerbalisationComplete dont la définition est donnée en prédicat 4. Les prédicats déduits sur le tableau de conversation sont à prendre du point de vue du système, car ils vont intervenir uniquement dans ses déductions.

5.2.1 Ouverture

L'état d'ouverture est très simple, c'est le premier dans lequel se trouve l'interaction. Sa table est donnée en table 5.1. Il ne contient qu'un comportement réalisable, dont le rôle est d'informer l'utilisateur (à l'aide du prédicat function(*helpForIR*), *helpForIR* étant une constante) que la fonction du système est de l'aider à réaliser sa tâche de recherche d'information.

Ouverture	
Prérequis	InteractionNotStarted
Règles	$\varnothing$
Comportement réalisable	
Motif dialogique	Inform(y, function(*helpForIR*))
Préconditions	$\neg$ SystemFunctionShared
Règles	$\varnothing$
Effets	$T_j \ni$ C(y, function(*helpForIR*), **Crt**)

TABLE D'ÉTAT 5.1 – État d'ouverture.

Avec comme prédicats déduits :

InteractionNotStarted vrai si l'interaction n'a pas encore commencé (on se trouve au tout début de l'interaction) ;

SystemFunctionShared vrai si le système a partagé avec l'utilisateur le fait que sa fonction est de l'aider sur une tâche de RI.

5.2.2 Formulation d'une verbalisation

Le scénario donné en section 5.1.2 indique que l'état de formulation d'une verbalisation suit l'état d'ouverture de l'interaction. La table de cet état est donnée en table 5.2. Ses prérequis correspondent au fait que le système a partagé sa fonction avec l'utilisateur (SystemFunctionShared définit en prédicat 2) et que celui-ci n'a pas encore formulé de verbalisation (VerbalisationMissing définit en prédicat 3). Il correspond au moment où l'utilisateur décrit son besoin d'information en langue naturelle et ne contient donc que des comportements attendus. Le premier comportement correspond au fait que l'utilisateur exprime une partie de son besoin d'information, en utilisant le jeu de communication de transfert d'information Inform sur le prédicat verbExpression(*expr*). Le second comportement correspond au fait que l'utilisateur affirme qu'il a terminé d'exprimer son besoin d'information, en utilisant le jeu de communication de transfert d'information Inform sur le prédicat verbComplete.

Formulation d'une verbalisation		
Prérequis	$\wedge$	SystemFunctionShared VerbalisationMissing
Règles	$\varnothing$	
Comportement attendu		
Motif dialogique	Inform(x, verbExpression(*expr*))	
Règles	$\varnothing$	
Effets	$\mathrm{T}_j \ni \mathrm{C}(x, \text{verbExpression}(expr), \mathbf{Crt})$	
Comportement attendu		
Motif dialogique	Inform(x, verbComplete)	
Règles	$\varnothing$	
Effets	$\mathrm{T}_j \ni \mathrm{C}(x, \text{verbComplete}, \mathbf{Crt})$	

TABLE D'ÉTAT 5.2 – État de formulation d'une verbalisation.

Cet état utilise un nouveau prédicat déduit :

VerbalisationMissing vrai si l'utilisateur n'a pas encore formulé de verbalisation.

5.2.3 Reformulation de la verbalisation

Le principe de l'état de reformulation est que le système tente d'interpréter la verbalisation de l'utilisateur et de l'exprimer en d'autres termes afin de vérifier que cette

interprétation est correcte du point de vue de l'utilisateur. Cet état est décrit dans la table d'état 5.3. Ses prérequis décrivent le fait que la verbalisation de l'utilisateur doit être terminée (VerbalisationComplete défini en prédicat 4) et qu'elle doit être suffisamment complète (VerbalisationSufficient défini en prédicat 6).

L'état ne contient qu'un seul comportement, réalisé par le système, où celui-ci utilise un jeu de vérification afin de savoir si sa reformulation est correcte ou non. Ce jeu de vérification porte sur le prédicat isReformulation(*expr*, *ref*), qui est vrai si *ref* est une reformulation de *expr*. La précondition de ce comportement est que le système a trouvé une reformulation de la verbalisation de l'utilisateur, correspondant au prédicat Reformulation(*expr*, *ref*), défini en prédicat 7.

Reformulation de la verbalisation		
Prérequis	$\wedge$	VerbalisationComplete VerbalisationSufficient
Règles	$\varnothing$	
Comportement réalisable		
Motif dialogique	CheckQuestion(y, ?isReformulation(*expr*, *ref*))	
Préconditions	Reformulation(*expr*, *ref*) $\wedge$ VerbExpression(*expr*)	
Règles	$\varnothing$	
Effets	$\vee$	$T_j \ni C(x, \text{isReformulation}(expr, ref), \mathbf{Crt})$ $T_j \ni C(x, \neg\, \text{isReformulation}(expr, ref), \mathbf{Crt})$

Table d'état 5.3 – État de reformulation de la verbalisation.

Avec comme prédicats déduits :

VerbalisationComplete vrai si l'utilisateur a affirmé que sa verbalisation est complète ;

VerbalisationSufficient vrai si la verbalisation de l'utilisateur est suffisamment riche ;

Reformulation(*expr*, *ref*) vrai si le système considère que *ref* est une reformulation de *expr* ;

VerbExpression(*expr*) vrai si *expr* est une expression de la verbalisation.

5.2.4 Demande de précisions de la verbalisation

L'état de demande de précision intervient lorsque l'utilisateur a terminé sa verbalisation mais que le système considère que celle-ci n'est pas assez complète. L'état a donc pour prérequis le prédicat VerbalisationComplete. Le prédicat PrecisionNotAsked est ajouté pour éviter de demander à plusieurs reprises la précision de sa verbalisation à l'utilisateur. Le prédicat QueryEmpty permet d'éviter de demander à l'utilisateur de préciser sa verbalisation s'il est déjà en train de formuler sa requête. Le seul comportement possible de l'état est émis par le système, qui demande à l'utilisateur de réaliser l'action de préciser sa verbalisation (représentée par preciseVerb) à l'aide du jeu de dialogue Request. Les préconditions de ce comportement correspondent au fait que la verbalisation de l'utilisateur n'est pas suffisante. Les règles décrivent que si l'utilisateur accepte la requête du système,

alors son engagement sur la complétion de sa verbalisation est *annulé* c'est-à-dire passé de l'état **Crt** à l'état **Ina**.

Demande de précision de la verbalisation		
Prérequis	$\wedge$	VerbalisationComplete PrecisionNotAsked QueryEmpty
Règles	$\varnothing$	
Comportement réalisable		
Motif dialogique	Request(y, preciseVerb)	
Préconditions	$\neg$ VerbalisationSufficient	
Règles	Succès $\Rightarrow$ C(x, verbComplete, **Ina**)	
Effets	$\vee$	$\wedge$: $T_j \ni$ C(x, preciseVerb, **Crt**) $T_j \ni$ C(x, verbComplete, **Ina**) $T_j \ni$ C(x, preciseVerb, **Fal**)

TABLE D'ÉTAT 5.4 – État de demande de précisions.

Cet état utilise les nouveaux prédicats déduits suivants :

PrecisionNotAsked vrai si le système n'a pas encore demandé à l'utilisateur de préciser sa verbalisation ;

QueryEmpty vrai si la requête ne comporte aucun mot-clé.

Remarque

Les règles du comportement réalisable de cet état illustrent les capacités de gestion du cycle de vie d'un engagement de notre système. En effet, les règles associées aux comportements possibles permettent, comme on le voit ici, de conserver une cohérence des engagements les uns par rapport aux autres sans pour autant avoir à exprimer sous la forme d'acte de dialogue toutes les opérations réalisées sur le tableau de conversation. Cette structure permet la gestion extra-dialogique des conséquences d'un acte de dialogue dans un contexte donné.

5.2.5 Précision de la verbalisation

L'état de précision de la verbalisation intervient au moment où l'utilisateur souhaite apporter un complément à sa verbalisation. Sa table d'état est donnée en table 5.5. Dans l'interaction humain-machine sur la tâche de RICM, nous considérons que le complément de la verbalisation de l'utilisateur ne sera réalisé que sur demande du système, faite dans l'état de demande de précision (contrairement à l'interaction humain-humain où l'utilisateur peut donner des précisions spontanément). Les prérequis de l'état de précision sont donc le fait que l'utilisateur se soit engagé sur la précision de sa verbalisation, avec le prédicat UserWillPreciseVerbalisation. Dans cet état, deux comportements sont attendus. Le premier correspond au fait que l'utilisateur ajoute des informations à sa verbalisation,

à l'aide du jeu de communication Inform. Les règles de ce comportement précisent que l'ouverture du motif dialogique d'Inform sur une expression de la verbalisation entrainent la satisfaction de l'engagement en action extra-dialogique contracté dans l'état de demande de précision. Celui-ci passe alors dans l'état **Ful**. Le second comportement attendu correspond au fait que l'utilisateur indique qu'il a terminé sa verbalisation à l'aide d'un jeu de communication Inform. Ses règles précisent que si l'utilisateur ouvre un motif dialogique d'Inform sur le fait qu'il a terminé sa verbalisation, mais qu'il est toujours engagé sur l'action de la compléter (il n'a donc rien ajouté à sa verbalisation), alors il *viole* cet engagement en action, qui passe donc dans l'état **Vio**.

Précision de la verbalisation		
Prérequis	UserWillPreciseVerbalisation	
Règles	$\varnothing$	
Comportement attendu		
Motif dialogique	Inform(x, verbExpression($expr$))	
Règles	Entrée $\Rightarrow$ C(x, preciseVerb, **Ful**) $\lfloor$ $\mathrm{T}_i \ni$ C(x, preciseVerb, **Crt**)	
Effets	$\vee$	$\mathrm{T}_j \ni$ C(x, verbExpression($expr$), **Crt**) $\mathrm{T}_j \ni$ C(x, preciseVerb, **Ful**)
Comportement attendu		
Motif dialogique	Inform(x, verbComplete)	
Règles	Entrée $\Rightarrow$ C(x, preciseVerb, **Vio**) $\lfloor$ $\mathrm{T}_i \ni$ C(x, preciseVerb, **Crt**)	
Effets	$\vee$	$\mathrm{T}_j \ni$ C(x, verbComplete, **Crt**) $\mathrm{T}_j \ni$ C(x, preciseVerb, **Vio**)

TABLE D'ÉTAT 5.5 – État de précision de la verbalisation.

Cet état utilise prédicat déduit suivant :

UserWillPreciseVerbalisation vrai si l'utilisateur s'est engagé à préciser sa verbalisation.

Remarque

De la même manière que pour l'état de demande de précision, nous avons ici un exemple de l'utilité des règles pour maintenir la cohérence extra-dialogique du tableau de conversation tout au long de la réalisation de la tâche. En effet, ces règles lient l'occurrence de certains actes de dialogue relatifs à la verbalisation à la mise à jour en conséquence de l'engagement en action de l'utilisateur de préciser sa verbalisation. Ces exemples illustrent la gestion du cycle de vie d'un engagement en action sans avoir besoin de recourir à un jeu de dialogue de décharge actant explicitement la réalisation de l'action.

5.2.6 Alignement verbalisation/terminologie

La phase de construction de la requête ne comporte qu'un seul état, celui d'alignement verbalisation/terminologie, présenté en table 5.6. Ses prérequis couvrent trois situations : l'utilisateur a verbalisé son besoin d'information (VerbalisationComplete, prédicat 4) et celui-ci est suffisant (VerbalisationSufficient, prédicat 6), ou il a donné une verbalisation et l'agent n'a pas demandé de précisions (VerbalisationComplete et PrecisionNotAsked, prédicat 8) ou la demande de précision par le système à échoué (VerbalisationPrecisionFailed, prédicat 12). Dans tous les cas, pour que les prérequis soient activés, il est nécessaire que la requête n'ait pas encore été lancée. Cet état contient un comportement attendu, correspondant à l'ajout d'un mot-clé à la requête, réalisé à l'aide du jeu de dialogue Request portant sur l'action d'ajouter le mot-clé souhaité (addKeyWord(*kw*)). Les règles de ce comportement indiquent que le système n'a pas le droit de refuser cette demande et qu'il exécute l'action instantanément, entraînant la satisfaction de l'engagement en action correspondant et la création de l'engagement sur le fait que le mot-clé fait partie de la requête. Les règles décrivant l'exécution instantanée de cette action illustrent la simplification de la réalisation d'une action sur laquelle s'est engagée le système, discutée en section 3.1.3.

L'autre comportement possible de l'état correspondent à la même action mais initiée par le système. Le système peut proposer à l'utilisateur d'ajouter un mot-clé à la requête, à l'aide du jeu de dialogue Offer. Les préconditions indiquent que le mot-clé *kw* ne doit pas faire partie de la requête courante (¬QueryKeyWord(*kw*), prédicat 13) et que l'ajout du mot-clé à la requête est pertinent (RelevantForAddition(*kw*), prédicat 14). Les règles de ce comportement correspondent à l'exécution directe de l'action proposée par le système.

Avec comme nouveaux prédicats déduits :

QueryNeverLaunched vrai si aucun lancement de requête n'a encore été réalisé ;

VerbalisationPrecisionFailed vrai si la précision de la verbalisation a échoué, c'est-à-dire que l'utilisateur a refusé de préciser sa verbalisation ou s'y est engagé mais ne l'a pas fait ;

QueryKeyWord(*kw*) vrai si le mot-clé *kw* est un mot-clé de la requête courante ;

RelevantForAddition(*kw*) vrai si l'ajout du mot-clé *kw* à la requête est constructif pour préciser la requête.

Alignement verbalisation/terminologie		
Prérequis	$\wedge$	QueryNeverLaunched $\vee$ $\wedge$ VerbalisationComplete $\wedge$ VerbalisationSufficient $\vee$ PrecisionNotAsked $\vee$ VerbalisationPrecisionFailed
Règles	$\varnothing$	
Comportement attendu		
Motif dialogique	Request(x, addKeyWord(kw))	
Règles	$\neg$ Échec Succès $\Rightarrow$ C(x, queryKeyWord(kw), **Crt**) Succès $\Rightarrow$ C(y, addKeyWord(kw), **Ful**)	
Effets	$\wedge$	$T_j \ni$ C(x, queryKeyWord(kw), **Crt**) $T_j \ni$ C(y, addKeyWord(kw), **Ful**)
Comportement réalisable		
Motif dialogique	Offer(y, addKeyWord(kw))	
Préconditions	$\neg$ QueryKeyWord(kw) $\wedge$ RelevantForAddition(kw)	
Règles	Succès $\Rightarrow$ C(x, queryKeyWord(kw), **Crt**) Succès $\Rightarrow$ C(y, addKeyWord(kw), **Ful**)	
Effets	$\vee$	$\wedge$ $T_j \ni$ C(x, queryKeyWord(kw), **Crt**) $\wedge$ $T_j \ni$ C(y, addKeyWord(kw), **Ful**) $\vee$ $T_j \ni$ C(y, addKeyWord(kw), **Fal**)

TABLE D'ÉTAT 5.6 – État d'alignement verbalisation/terminologie.

5.3 Lancement de la requête

La phase de lancement de la requête ne contient qu'un seul état, qui correspond à l'action de lancement de la requête courante par l'un ou l'autre des interlocuteurs. Cet état est présenté en table 5.7. Les préconditions de cet état indiquent simplement que la requête doit être lançable (QueryLaunchable, prédicat 15). Le comportement attendu permet à l'utilisateur de demander au système de lancer la requête courante à l'aide du jeu de dialogue Request. De la même manière que pour l'ajout de mot-clé à la requête, le système ne peut pas refuser cette demande et l'exécute instantanément, menant à la satisfaction de l'engagement en action correspondant et à la création de l'engagement sur le fait que la requête a été lancée.

Le système peut aussi proposer à l'utilisateur de lancer la requête courante à l'aide du jeu de dialogue Offer. Les préconditions indiquent que le lancement de la requête courante doit être pertinent (QueryLaunchRelevant, prédicat 16). De la même manière que pour le comportement précédent, les règles correspondent à l'exécution directe de l'action proposée par le système.

Avec comme nouveaux prédicats déduits :

QueryLaunchable vrai si la requête courante respecte les critères minimaux pour être

Lancement de la requête		
Prérequis	QueryLaunchable	
Règles	$\varnothing$	
Comportement attendu		
Motif dialogique	Request(x, launchCurrentQuery)	
Règles	$\neg$ Échec Succès $\Rightarrow$ C(y, currentQueryLaunched, **Crt**) Succès $\Rightarrow$ C(y, launchCurrentQuery, **Ful**)	
Effets	$\wedge$	$T_j \ni$ C(y, currentQueryLaunched, **Crt**) $T_j \ni$ C(y, launchCurrentQuery, **Ful**)
Comportement réalisable		
Motif dialogique	Offer(y, launchCurrentQuery)	
Préconditions	QueryLaunchRelevant	
Règles	Succès $\Rightarrow$ C(y, currentQueryLaunched, **Crt**) Succès $\Rightarrow$ C(y, launchCurrentQuery, **Ful**)	
Effets	$\vee$	$\wedge$ $T_j \ni$ C(y, currentQueryLaunched, **Crt**) $\wedge$ $T_j \ni$ C(y, launchCurrentQuery, **Ful**) $T_j \ni$ C(y, launchCurrentQuery, **Fal**)

TABLE D'ÉTAT 5.7 – État de lancement de la requête.

lancée ;

QueryLaunchRelevant vrai si le système juge qu'il est pertinent de proposer à l'utilisateur de lancer la requête courante.

5.4 Évaluation des résultats

La phase d'évaluation des résultats est celle dont la structure va être le plus impactée par le passage du scénario humain-humain à une interaction humain-machine. En effet certaines parties du scénario sont réalisées explicitement lors de l'interaction humain-humain mais le seront implicitement lors de l'interaction humain-machine et ne seront donc pas représentées sous la forme d'états. C'est notamment le cas de l'état d'évaluation des résultats par l'expert, discuté plus longuement en section 5.4.1. C'est aussi le cas des états où les résultats sont jugés partiellement satisfaisants ou non satisfaisants, discutés en section 5.4.4. Seulement deux états sont finalement préservés dans cette phase. Nous les présentons dans la suite de cette section : l'évaluation des résultats par l'utilisateur, en section 5.4.2 (table 5.8) et la sortie dans le cas où l'évaluation serait positive, en section 5.4.3 (table 5.9).

5.4.1 Évaluation des résultats par l'expert

Dans le scénario humain-humain, l'évaluation des résultats renvoyés par le moteur de recherche est réalisée explicitement par l'expert, qui les commente pour faire part de son jugement à son interlocuteur. Cette situation sera complètement modifiée lors du passage à une interaction humain-machine étant donné que le système aura accès aux résultats de la requête avant qu'ils ne soient présentés à l'utilisateur. Il est donc capable de les évaluer de manière « cachée », c'est-à-dire lors du calcul de prédicats, dans les états en amont du lancement de la requête. Cette situation lui permet de prendre de l'avance sur les conséquences que peut avoir la proposition d'ajouter un mot-clé à la requête ou de lancer celle-ci. C'est par exemple ce qui est fait dans les prédicats QueryLaunchable (prédicat 15) et QueryLaunchRelevant (prédicat 16), qui tiennent compte des résultats renvoyés par la requête courante dans leur évaluation.

La possibilité d'anticiper l'évaluation des résultats d'une requête donne un avantage au système qui a alors la possibilité de connaître les conséquences d'une action qu'il envisage de proposer à l'utilisateur. Le prédicat QueryLaunchable (prédicat 15) illustre cet intérêt car il évite de proposer à l'utilisateur de lancer une requête qui ne renverrait aucun résultat (c'est ce qui justifie son utilisation en précondition de la table de lancement de la requête 5.7).

Cette évaluation implicite réalisée à priori par le système fait partie de son fonctionnement privé et n'est pas visible pour l'utilisateur. Il n'existe donc pas d'état d'évaluation des résultats de la requête par le système à proprement parler. Les seules informations sur l'évaluation d'une requête par le système qui sont rendues publiques sont celles que le système juge pertinentes au moment de l'écriture de la requête. Plutôt que d'avoir accès à l'intégralité de l'évaluation réalisée par le système, l'utilisateur n'aura accès qu'à la partie que le système considère comme utile. C'est ensuite à l'utilisateur de décider si ce que le système propose l'intéresse et d'en tenir compte ou non.

5.4.2 Évaluation des résultats par l'utilisateur

La phase d'évaluation des résultats est grandement simplifiée dans notre vision de l'interaction humain-machine et est principalement composée de l'état d'évaluation des résultats par l'utilisateur, décrit en table 5.8. Le prérequis de cet état est le fait que la requête actuelle a été lancée (CurrentQueryLaunched, prédicat 17). Les trois comportements attendus de cet état correspondent au fait que l'utilisateur consulte des ressources (seenResource(r)), les évalue (relevantResource(r) ou $\neg$ relevantResource(r)) et se prononce sur la satisfaction de son besoin d'information par les résultats de la requête courante (currentQueryResultsSatisfying ou $\neg$ currentQueryResultsSatisfying). Ces comportements sont réalisés à l'aide du jeu de communication Inform. Le comportement réalisable de cet état correspond au fait que le système demande à l'utilisateur s'il a terminé sa recherche, à l'aide d'un jeu de CheckQuestion portant sur le prédicat researchOver. Les préconditions de ce comportement spécifient que l'utilisateur doit avoir évalué les résultats de la requête courante comme satisfaisants (CurrentQueryResultsSatisfying, prédicat 18), qu'il n'a pas exprimé le fait que sa recherche est terminée (ResearchNotOver, prédicat 19) et que le

système ne lui a pas déjà demandé s'il a terminé sa recherche (ResearchOverNotAsked, prédicat 20) depuis la dernière fois qu'il a exprimé sa satisfaction sur les résultats de la requête. Dans le cas où l'utilisateur infirme le fait qu'il a terminé sa recherche (il joue un acte *disconfirm*), menant à un échec du motif dialogique, une règle s'applique et désactive l'engagement de l'utilisateur sur le fait que les résultats de la requête courante sont satisfaisants, dans le cas où cet engagement est activé.

Évaluation des résultats (utilisateur)	
Prérequis	CurrentQueryLaunched
Règles	$\varnothing$
Comportement attendu	
Motif dialogique	Inform(x, seenResource(r))
Règles	$\varnothing$
Effets	$\mathrm{T}_j \ni \mathrm{C}(x, \text{seenResource}(r), \textbf{Crt})$
Comportement attendu	
Motif dialogique	Inform(x, relevantResource(r) | $\neg$ relevantResource(r))
Règles	$\varnothing$
Effets	$\mathrm{T}_j \ni \mathrm{C}(x, \text{relevantResource}(r) \mid \neg \text{relevantResource}(r), \textbf{Crt})$
Comportement attendu	
Motif dialogique	Inform(x, currentQueryResultsSatisfying | $\neg$ currentQueryResultsSatisfying)
Règles	$\varnothing$
Effets	$\mathrm{T}_j \ni \mathrm{C}(x, \text{currentQueryResultsSatisfying} \mid \neg \text{currentQueryResultsSatisfying}, \textbf{Crt})$
Comportement réalisable	
Motif dialogique	CheckQuestion(y, ?researchOver)
Préconditions	CurrentQueryResultsSatisfying $\wedge$ ResearchNotOver $\wedge$ ResearchOverNotAsked
Règles	Échec $\Rightarrow \mathrm{C}(x, \text{currentQueryResultsSatisfying}, \textbf{Ina})$ $\lfloor\ \mathrm{T}_i \ni \mathrm{C}(x, \text{currentQueryResultsSatisfying}, \textbf{Crt})$
Effets	$\mathrm{T}_j \ni \mathrm{C}(x, \text{researchOver}, \textbf{Crt})$ $\vee$ $\wedge$ $\mathrm{T}_j \ni \mathrm{C}(x, \neg\ \text{researchOver}, \textbf{Crt})$ $\mathrm{T}_j \ni \mathrm{C}(x, \text{currentQueryResultsSatisfying}, \textbf{Ina})$

TABLE D'ÉTAT 5.8 – État d'évaluation des résultats par l'utilisateur.

Avec comme prédicats déduits :

CurrentQueryLaunched vrai si la requête courante a été lancée ;

CurrentQueryResultsSatisfying vrai si les résultats de la requête courante ont été jugés satisfaisants par l'utilisateur ;

ResearchNotOver vrai si l'utilisateur n'a pas exprimé le fait que sa recherche est terminée ;

ResearchOverNotAsked vrai si le système n'a pas demandé à l'utilisateur si sa recherche est terminée depuis la dernière fois qu'il a exprimé que les résultats de la requête sont satisfaisants.

5.4.3 Sortie

La sortie de l'interaction correspond au moment où la tâche est complétée et où la session de recherche se termine. L'état de sortie correspond donc simplement au moment

où le système informe l'utilisateur que leur interaction est terminée. Cette action est capturée dans le comportement possible de l'état, dont le motif dialogique est le jeu de communication Inform. Les prérequis de l'état, de même que les préconditions du comportement, sont le fait que la recherche soit terminée (ResearchOver, prédicat 21).

Sortie	
Prérequis	ResearchOver
Règles	$\varnothing$
Comportement réalisable	
Motif dialogique	Inform(y, interactionOver)
Préconditions	ResearchOver
Règles	$\varnothing$
Effets	$T_j \ni C(y, \text{interactionOver}, \textbf{Crt})$

TABLE D'ÉTAT 5.9 – État de sortie.

ResearchOver vrai si l'utilisateur a dit avoir terminé sa recherche.

5.4.4 Résultats partiellement satisfaisants et non satisfaisants

De même que pour l'évaluation des résultats par le système, cet état n'est pas retranscrit dans la version humain-machine de l'interaction sur une tâche de RICM. En effet, comme l'évaluation des résultats par le système est réalisée implicitement (cf. section 5.4.1), le rejet des résultats de la requête ne sera pas à son initiative dans la phase d'évaluation des résultats. De plus, le niveau de satisfaction de l'utilisateur par rapport aux résultats renvoyés par la requête a déjà été exprimé dans son état d'évaluation des résultats. Le fait que les résultats de la requête sont satisfaisants ou partiellement satisfaisants sera donc déterminé uniquement à partir des actions de l'utilisateur lors de son évaluation des résultats.

5.5 Réparation de la requête

On retrouve dans la phase de réparation de la requête les trois stratégies mises en place dans le scénario humain-humain : précision de la requête, généralisation de la requête et reformulation. Comme discuté en section 5.1.3, nous ne définissons pas une étape par stratégie. En effet, chaque stratégie nécessite de simples opérations sur la requête (ajout, suppression ou substitution de mots-clés) et c'est les préconditions de chaque comportement associé qui permettent au système d'utiliser certaines de ces opérations pour appliquer ces stratégies. La distinction entre les différentes stratégies se fait donc au niveau du calcul des prédicats plutôt qu'au niveau de la définition de l'étape de réparation en elle-même.

L'état de réparation de la requête comportant de nombreux comportements, nous les avons séparés en deux tables distinctes : une pour les comportement attendus de la

Comportements de réparation (attendus)		
Prérequis	$\wedge$	CurrentQueryLaunched $\neg$ CurrentQueryResultsSatisfying
Règles		_ $\Rightarrow$ C(x, $\neg$ currentQueryResultsSatisfying, **Crt**) $\lfloor$ $T_i \ni$ C(x, $\neg$ currentQueryResultsSatisfying, **Ina**)
Comportement attendu		
Motif dialogique		Request(x, addKeyWord(kw))
Règles		$\neg$ Échec Succès $\Rightarrow$ C(x, queryKeyWord(kw), **Crt**) Succès $\Rightarrow$ C(y, addKeyWord(kw), **Ful**) Succès $\Rightarrow$ C(y, currentQueryLaunched, **Ina**) $\lfloor$ $T_i \ni$ C(y, currentQueryLaunched, **Crt**)
Effets	$\wedge$	$T_j \ni$ C(x, queryKeyWord(kw), **Crt**) $T_j \ni$ C(y, addKeyWord(kw), **Ful**) $T_j \ni$ C(y, currentQueryLaunched, **Ina**)
Comportement attendu		
Motif dialogique		Request(x, removeKeyWord(kw))
Règles		$\neg$ Échec Succès $\Rightarrow$ C(x, queryKeyWord(kw), **Ina**) Succès $\Rightarrow$ C(y, removeKeyWord(kw), **Ful**) Succès $\Rightarrow$ C(y, currentQueryLaunched, **Ina**) $\lfloor$ $T_i \ni$ C(y, currentQueryLaunched, **Crt**)
Effets	$\wedge$	$T_j \ni$ C(x, queryKeyWord(kw), **Ina**) $T_j \ni$ C(y, removeKeyWord(kw), **Ful**) $T_j \ni$ C(y, currentQueryLaunched, **Ina**)

TABLE D'ÉTAT 5.10 – Comportements attendus de l'état de réparation de la requête.

part de l'utilisateur (table 5.10) et une pour les comportements réalisables par le système (table 5.11).

La règle générale stipule que si l'on rentre dans un état de la phase de réparation, cela signifie que les résultats de la requête courante ne sont pas satisfaisants. Le premier comportement attendu correspond à la demande de la part de l'utilisateur d'ajouter un mot-clé à la requête, réalisé à l'aide du jeu de dialogue Request. Ce comportement attendu est similaire à celui de l'état d'alignement verbalisation/terminologie présenté en table 5.6. Ses règles stipulent que le système n'a pas le droit de refuser la demande de l'utilisateur et que celle-ci est exécutée immédiatement. Le second comportement attendu de l'utilisateur est la demande par celui-ci de supprimer un mot-clé de la requête, à l'aide d'un jeu de dialogue Request. Les règles de ce comportement indiquent que le système ne peut pas refuser et qu'il exécute directement la demande de l'utilisateur. Pour les deux comportements attendus, par rapport à l'état d'alignement verbalisation/terminologie, une règle s'ajoute indiquant que l'acceptation par le système de cette demande désactive

les engagements sur le lancement de la requête.

Du côté du système, trois comportements sont réalisables. Les deux premiers sont similaires à ceux de l'utilisateur : le premier consiste à proposer à l'utilisateur d'ajouter un mot-clé à la requête, comme dans l'état d'alignement verbalisation/terminologie (table 5.6), en respectant les mêmes préconditions. Le second consiste à proposer à l'utilisateur de supprimer un mot-clé de la requête. Les préconditions pour permettre la proposition de ce comportement vérifient que le mot-clé proposé (*kw*) appartient bien à la requête courante et que sa suppression est pertinente. Les règles précisent que si l'utilisateur accepte cette proposition, elle est mise en œuvre immédiatement par le système. Le troisième comportement permet au système de proposer à l'utilisateur de substituer un mot-clé par un autre. Ce comportement est utile notamment pour la mise en œuvre de stratégies de réparation de la requête impliquant la simplification ou la spécification de mots-clés. Les préconditions vérifient que le mot-clé à substituer (*kw*) appartient bien à la requête courante et que le mot-clé substituant (*skw*) est pertinent.

Avec comme prédicats déduits :

RelevantForRemoval(*kw*) vrai si la suppression du mot-clé *kw* de la requête est pertinente ;

RelevantForSubstitution(*kw*, *skw*) si le remplacement du mot-clé *kw* par *skw* est pertinent pour modifier la requête, sans forcément la préciser ou la simplifier.

Comportements de réparation (réalisables)		
Prérequis	$\wedge$	CurrentQueryLaunched $\neg$ CurrentQueryResultsSatisfying
Règles		$\_ \Rightarrow$ C(x, $\neg$ currentQueryResultsSatisfying, **Crt**) $\lfloor$ T$_i \ni$ C(x, $\neg$ currentQueryResultsSatisfying, **Ina**)
Comportement réalisable		
Motif dialogique		Offer(y, addKeyWord(kw))
Préconditions		$\neg$ QueryKeyWord(kw) $\wedge$ RelevantForAddition(kw)
Règles		Succès $\Rightarrow$ C(x, queryKeyWord(kw), **Crt**) Succès $\Rightarrow$ C(y, addKeyWord(kw), **Ful**) Succès $\Rightarrow$ C(y, currentQueryLaunched, **Ina**) $\lfloor$ T$_i \ni$ C(y, currentQueryLaunched, **Crt**)
Effets	$\vee$	$\wedge$ T$_j \ni$ C(x, queryKeyWord(kw), **Crt**) $\wedge$ T$_j \ni$ C(y, addKeyWord(kw), **Ful**) $\wedge$ T$_j \ni$ C(y, currentQueryLaunched, **Ina**) T$_j \ni$ C(y, addKeyWord(kw), **Fal**)
Comportement réalisable		
Motif dialogique		Offer(y, removeKeyWord(kw))
Préconditions		QueryKeyWord(kw) $\wedge$ RelevantForRemoval(kw)
Règles		Succès $\Rightarrow$ C(x, queryKeyWord(kw), **Ina**) Succès $\Rightarrow$ C(y, removeKeyWord(kw), **Ful**) Succès $\Rightarrow$ C(y, currentQueryLaunched, **Ina**) $\lfloor$ T$_i \ni$ C(y, currentQueryLaunched, **Crt**)
Effets	$\vee$	$\wedge$ T$_j \ni$ C(x, queryKeyWord(kw), **Ina**) $\wedge$ T$_j \ni$ C(y, removeKeyWord(kw), **Ful**) $\wedge$ T$_j \ni$ C(y, currentQueryLaunched, **Ina**) T$_j \ni$ C(y, removeKeyWord(kw), **Fal**)
Comportement réalisable		
Motif dialogique		Offer(y, substituteKeyWord(kw, skw))
Préconditions		QueryKeyWord(kw) $\wedge$ RelevantForSubstitution(kw, skw)
Règles		Succès $\Rightarrow$ C(x, queryKeyWord(kw), **Ina**) Succès $\Rightarrow$ C(x, queryKeyWord(skw), **Crt**) Succès $\Rightarrow$ C(y, substituteKeyWord(kw, skw), **Ful**) Succès $\Rightarrow$ C(y, currentQueryLaunched, **Ina**) $\lfloor$ T$_i \ni$ C(y, currentQueryLaunched, **Crt**)
Effets	$\vee$	$\wedge$ T$_j \ni$ C(x, queryKeyWord(kw), **Ina**) $\wedge$ T$_j \ni$ C(x, queryKeyWord(skw), **Crt**) $\wedge$ T$_j \ni$ C(y, substituteKeyWord(kw, skw), **Ful**) $\wedge$ T$_j \ni$ C(y, currentQueryLaunched, **Ina**) T$_j \ni$ C(y, substituteKeyWord(kw, skw), **Fal**)

TABLE D'ÉTAT 5.11 – Comportements réalisables de l'état de réparation de la requête.

5.6 Modularité du modèle

Le fonctionnement de notre système a trois grands niveaux de modularité. Le premier niveau (section 5.6.1) permet de généraliser l'écriture des comportements possibles en se focalisant uniquement sur le contenu sémantique du motif dialogique associé et son impact sur la tâche. Le second niveau (section 5.6.2) montre l'utilité des règles pour spécifier les conséquences d'un acte de dialogue dans un contexte particulier. Le dernier niveau (section 5.6.3) montre une distinction franche entre la partie conventionnelle de l'interaction et les intentions sous-jacentes qui permettent à l'agent de décider des comportements à jouer.

5.6.1 Les comportements comme opérations atomiques

L'élément central d'un comportement est le motif dialogique associé. Ce motif véhicule un transfert d'information ou une discussion d'action. Il correspond à une *opération atomique* pouvant être réalisée pour faire progresser la tâche. L'organisation de ces opérations les unes par rapport aux autres explique la manière dont se déroule la tâche. Un comportement possible est une brique à positionner aux endroits appropriés de la réalisation de la tâche. C'est en ce sens que sont définies les tables d'état, qui regroupent ces opérations en unités cohérentes. En effet, certains comportements ne sont pas propres à un seul état et peuvent être réutilisés ailleurs. Une même opération peut se retrouver à différents moments de la réalisation de la tâche, c'est-à-dire dans différents états. Cela amène à avoir un même comportement possible présent dans plusieurs états. L'écriture d'un comportement peut donc être vue non pas comme une définition spécifique à un moment restreint de l'interaction, mais comme la conception d'une brique que l'on choisira de positionner aux endroits appropriés de la réalisation de la tâche.

C'est par exemple le cas du recoupement des comportements possibles des états d'alignement verbalisation/terminologie et de réparation de la requête.

Ce point de vue permet d'envisager la modélisation d'une interaction séparant distinctement les actions attendues des interlocuteurs pour faire avancer la tâche des interactions nécessaires à la réalisation de ces actions. Pour modéliser la réalisation de la tâche, il est uniquement nécessaire d'organiser les transferts d'information et les actions permettant de faire avancer la tâche. Il semble envisageable, dès lors qu'on a construit un ensemble de comportements encapsulant ces phénomènes de transfert d'information et de discussion d'action, de ne plus se soucier des interactions qui les véhiculent et de se focaliser sur leur organisation au sein des états.

Cela simplifie grandement l'écriture et la lecture des tables, celles-ci se résumant pour chaque ligne à une discussion d'action ou un transfert d'information, cette représentation synthétisant les informations importantes et laissant de côté celles spécifiques à l'interaction.

5.6.2 Les règles comme spécifications contextuelles

Cette modularité « par comportement » est cependant à relativiser en fonction du contexte de chaque état. Comme nous l'avons vu dans la description d'un état (section 3.1.2), les comportements possibles ne correspondent pas uniquement à un motif dialogique donné, et comportent aussi des règles. Or, pour un même motif dialogique, les règles du comportement auquel il appartient peuvent varier en fonction de l'état dans lequel il se trouve. Par exemple, quand l'utilisateur formule une verbalisation, les règles varient selon le contexte. Dans le cas où on se trouve dans l'état de formulation d'une verbalisation (section 5.2.2), aucune règle ne s'applique, alors que si on est dans l'état de précision de la verbalisation (section 5.2.5), une règle existe précisant que l'ouverture du motif par l'utilisateur entraîne la complétion de l'engagement en action extra-dialogique associé (création de C(x, preciseVerb, **Ful**)). Ces deux comportements sont reportés dans les tables 5.12 pour l'état de formulation d'une verbalisation et 5.13 pour l'état de précision de la verbalisation.

Comportement de formulation d'une verbalisation	
Comportement attendu	
Motif dialogique	Inform(x, verbExpression($expr$))
Règles	$\varnothing$
Effets	$T_j \ni C(x, \text{verbExpression}(expr), \textbf{Crt})$

TABLE D'ÉTAT 5.12 – Comportement de formulation d'une expression de verbalisation tiré de l'état de formulation d'une verbalisation (table 5.2).

Comportement de précision de la verbalisation		
Comportement attendu		
Motif dialogique	Inform(x, verbExpression($expr$))	
Règles	Entrée $\Rightarrow$ C(x, preciseVerb, **Ful**) $\lfloor$ $T_i \ni$ C(x, preciseVerb, **Crt**)	
Effets	$\vee$	$T_j \ni C(x, \text{verbExpression}(expr), \textbf{Crt})$ $T_j \ni C(x, \text{preciseVerb}, \textbf{Ful})$

TABLE D'ÉTAT 5.13 – Comportement de formulation d'une expression de verbalisation tiré de l'état de précision de la verbalisation (table 5.5).

Cependant, nous défendons l'idée selon laquelle il est possible de donner une définition « minimale » d'un comportement possible et de ses règles, et de les spécifier en fonction de la situation à laquelle on souhaite les appliquer. De nombreuses règles sont propres non pas au contexte courant du déroulement de la tâche mais au contexte général régissant l'interaction.

Notre agent étant collaboratif, nous avons décidé qu'il doit accepter toutes les demandes de l'utilisateur. Cela implique par exemple que lorsque l'utilisateur joue un jeu de dialogue Request, le système est contraint de faire en sorte que celui-ci soit un succès, peu importe

l'action sur laquelle porte le jeu de dialogue et peu importe le contexte dans lequel se trouve l'interaction. De même, lorsque le système s'engage sur la réalisation d'une action, il réalise celle-ci immédiatement (discuté en section 3.1.3, page 91). Ce phénomène est capturé par l'écriture de règles dans le comportement attendu associé au motif dialogique engageant le système sur la réalisation de l'action (souvent un jeu de dialogue Offer ou Request). Le tableau 5.14 illustre cet aspect sur l'ajout d'un mot-clé à la requête : les règles décrivent le fait que le système ne peut pas décliner la requête de l'utilisateur (¬ Échec) et qu'il ajoute le mot-clé directement (création des engagements C(x, queryKeyWord(kw), **Crt**) et C(y, addKeyWord(kw), **Ful**)).

Comportement de requête d'ajout de mot-clé		
Comportement attendu		
Motif dialogique	Request(x, addKeyWord(kw))	
Règles	¬ Échec	
	Succès ⇒ C(x, queryKeyWord(kw), **Crt**)	
	Succès ⇒ C(y, addKeyWord(kw), **Ful**)	
Effets	$\wedge$	$T_j \ni$ C(x, queryKeyWord(kw), **Crt**)
		$T_j \ni$ C(y, addKeyWord(kw), **Ful**)

TABLE D'ÉTAT 5.14 – Exemple de comportement avec des règles correspondant à des contraintes correspondant au contexte général de l'interaction.

Les règles des comportements possibles sont donc en partie régies par des principes généraux propres au contexte de l'interaction. Dans le cas où des règles s'appliquent à un état spécifique, il est possible de les ajouter au comportement dans cet état sans perdre la généricité de celui-ci. À cela s'ajoute le mécanisme de règles générales à un état, appliquées lors de l'entrée dans cet état et qui permettent de modifier l'état du dialogue afin de le faire correspondre au contexte propre de l'état. Cette utilisation de règles spécifiques constitue un deuxième niveau de modularité.

5.6.3 Séparation des interactions et des décisions

Un troisième niveau de modularité est l'indépendance entre la définition des prédicats et l'utilisation qui en est faite. La manière dont ils sont calculés n'est pas prise en compte lors de l'écriture des états. La définition des prédicats donnée en annexe A pour l'interaction humain-machine sur une tâche de RICM n'est qu'une proposition. Nous avons fait le choix de les définir à l'aide de formules logiques mais il est tout à fait envisageable de le faire autrement sans pour autant avoir à modifier la structure conventionnelle de la tâche donnée dans les états des sections 5.2, 5.4 et 5.5. La manière dont sont définis les prédicats est totalement indépendante de la structure de l'interaction et c'est à la discrétion du concepteur de choisir leur fonctionnement. Cela permet de faire évoluer leur définition sans affecter la définition des états, permettant une séparation claire entre la structure de données propre au raisonnement de l'agent et la structure de la tâche.

En résumé on peut dire que notre modèle de comportement possible sépare le *pourquoi*,

le *comment* et le *quoi* :

Le pourquoi est la précondition d'un comportement réalisable. Elle explique pourquoi le système peu réaliser ce comportement et elle est activée à l'issue de déductions réalisées par le système ;

Le comment est le motif dialogique du comportement réalisable. Il décrit comment le comportement doit être conventionnellement réalisé par les participants au dialogue ;

Le quoi est le contenu sémantique du motif dialogique du comportement réalisable. Il correspond à ce qui est transmis par l'initiateur du motif sous la forme d'une action ou d'une proposition.

5.7 Synthèse

Nous revenons ici sur la tâche de RICM et décrivons en quoi les états donnés dans les sections 5.2, 5.3, 5.4 et 5.5 correspondent à ce modèle. Nous donnons aussi les limites actuelles du modèle et comment les dépasser.

5.7.1 Propriétés conservées

Nous avons vu en section 5.1.2 que le modèle de RICM est interactif, opportuniste, itératif et stratégique. Ces propriétés sont conservées lors du passage au modèle humain-machine.

Interactif – Par construction de notre modèle, la tâche de RICM ne peut être réalisée que par l'interaction. Toute action réalisée par l'humain ou la machine s'intègre dans les motifs interactifs que sont les jeux de dialogue, eux-même organisés dans les structures cohérentes décrites par les états ;

Opportuniste – Le fait de ne pas planifier le déroulement de la tâche offre la possibilité de faire évoluer son déroulement de manière imprévisible. La prise de décision de la machine se base sur l'état courant de l'interaction et peut donc, en fonction de celui-ci, proposer spontanément des comportements impossibles à prévoir. De même, les mécanismes de contextualisation d'un coup de l'utilisateur permettent de s'adapter à de nombreuses actions de sa part, lui donnant un large champ d'action et ne le restreignant pas à un petit ensemble de coups dialogiques ;

Itératif – La boucle lancement/évaluation/réparation se retrouve dans les tables d'état représentant le scénario de RICM. Le lancement est réalisé à l'aide des comportements portant sur les motifs Request ou Offer sur l'action launchCurrentQuery. À la suite de ce lancement, le prédicat CurrentQueryLaunched est vrai, ce qui permet de rentrer dans l'état d'évaluation des résultats par l'utilisateur (table 5.8). Si l'évaluation de l'utilisateur est négative, alors l'engagement C(x, $\neg$ currentQueryResultsSatisfying, **Crt**) est créé. La création de cet engagement fait en sorte que les prérequis de l'état de réparation soient satisfaits. Cet enchaînement d'états peut être répété autant de fois que nécessaire jusqu'à ce que l'utilisateur trouve une réponse satisfaisante à son besoin d'information ;

Stratégique – Le fait que la tâche ne soit pas explicitement planifiée n'empêche pas la mise en place de stratégies. C'est notamment le cas pour la réparation de la requête, où le système décide selon la situation actuelle des opérations les plus pertinentes à entreprendre pour faire évoluer la requête, correspondant au niveau des *tactiques* définies par Bates [Bat90]. De plus, les comportements attendus de la part de l'utilisateur dans les états de réparation de la requête lui permettent de mettre en place tous les coups nécessaires à l'élaboration de tactiques, stratagèmes ou stratégies au sens de Bates.

Dans le corpus COGNI-CISMEF, quand les résultats de la requête ne sont pas satisfaisants pour l'utilisateur, celui-ci entre tout de suite dans la phase de réparation de la requête sans même le verbaliser. Les seules actions explicites retranscrites dans le corpus portent sur le fait que la requête est lancée, que l'expert propose les résultats à l'utilisateur et que celui-ci propose directement une modification de la requête. Cela implique que l'utilisateur court-circuite l'état d'évaluation des résultats et n'émet donc pas d'évaluation explicite sur les résultats de la requête. La transition vers la réparation est donc faite de manière implicite et est déduite par l'expert à la demande de l'utilisateur. Cependant, cette déduction n'est pas triviale pour le système car elle correspond à une situation où l'utilisateur ne respecte pas l'ordre dans lequel il doit a priori réaliser ses actions.

Il est néanmoins primordial de permettre ce cas de figure et de donner au système la capacité d'intégrer ces raccourcis dans son interprétation du comportement de l'utilisateur. C'est dans cette optique que la règle de l'état de réparation de requête a été mise en place (cf. tables 5.10 et 5.11). Celle-ci indique que dès l'entrée dans cet état, l'utilisateur rejette les résultats de la dernière requête lancée. L'utilisateur est libre de faire ou non des retours sur son appréciation des résultats de la requête, et le système est capable d'interpréter ses actions à ce sujet, même implicites.

5.7.2 Limites

Il n'est pas possible, dans l'état actuel de notre modèle, de réaliser certains coups dialogiques, souvent implicites dans l'interaction humain-humain. C'est notamment le cas de l'acceptation implicite de la part de l'utilisateur d'une proposition ou d'une demande du système. Par exemple, dans l'état de demande de précision (tableau 5.4), si le système demande à l'utilisateur de préciser sa verbalisation à l'aide du jeu de dialogue Request(y, preciseVerb) et que l'utilisateur, plutôt que d'accepter explicitement en jouant un acceptRequest(x, preciseVerb) commence directement à compléter sa verbalisation en faisant un inform(x, verbExpression(*expr*)), rien n'est prévu pour jouer implicitement l'acte d'acceptation. Le jeu de requête se trouvera donc sans réponse alors qu'il s'est implicitement terminé.

On se heurte ici à une limite de notre modèle, qui ne permet pas en l'état de décrire tous les comportements implicites del'utilisateur. Cependant, une extension est envisageable tout en préservant la structure actuelle du formalisme. La prise en compte de ce type de comportements implicites est possible grâce aux règles du comportement par lequel l'utilisateur accepte implicitement la demande du système. L'écriture de ces règles implique

de manipuler directement les actions dialogiques du motif dialogique du comportement implicitement manipulé, ce qui actuellement n'est pas prévu. Cela nécessite la prise en compte d'engagements en actions dialogiques, dont nous n'avons que peu discuté dans ces travaux cas ils se rattachent plus au maniement de structures spécialisées de gestion du dialogue qu'à la gestion de la tâche en elle-même par l'interaction. De ce fait, nous laissons la prise en compte des phénomènes d'acceptation implicites de discussion d'action à des développements futurs.

Chapitre 6

Mise en œuvre expérimentale

« J'éprouvais ce sentiment poignant de ne pas être compris, tout au plus décodé, d'être le seul détenteur du sens dans un monde insensible où la seule chose qui passait, c'était un peu d'électricité dans des milliers de circuits. Je pouvais poser n'importe quelle question, on m'opposait toujours une réponse dans cet univers plein, et quelle que fût son ineptie, c'était à moi d'y insuffler un sens. »

Alain Damasio, La zone du dehors

Sommaire

Ce chapitre donne une évaluation de notre modèle (chapitre 3) et de son application à une tâche de RICM (chapitre 5). Hillman identifie trois approches pour une évaluation d'utilisabilité (« usability evaluation » en anglais) d'un système de dialogue [Hil18] :

L'analyse experte – elle a pour principe de confier à des experts humains l'évaluation de certains aspects du système. Deux types d'expertises permettent d'évaluer un système de dialogue, d'une part *l'expertise ergonomique*, qui se concentre sur la facilité d'utilisation du système. D'autre part, *l'expertise de domaine*, qui se concentre sur l'utilité du système de dialogue vis-à-vis du domaine auquel il s'applique ;

L'analyse avec utilisateur – elle fait appel à des futurs utilisateurs du système pour tester un prototype de celui-ci. Le prototype n'a pas besoin d'être le système final et peut être que partiellement implémenté avec un magicien d'Oz complétant les parties manquantes ;

L'analyse par simulation – cette approche utilise un modèle d'utilisateur humain afin d'analyser les interactions possibles entre celui-ci et le système. Une architecture possible de modèle utilisateur est celle *basée sur des règles*. Cette architecture décrit quelles actions l'utilisateur doit entreprendre en fonction du contexte dans lequel il se trouve.

La première approche ne nous est pas possible car elle nécessite d'avoir à notre disposition des experts capables d'évaluer le système. L'accès à la plate-forme CISMeF ne nous étant pas possible actuellement, la seconde approche a été écartée. Nous nous tournons donc vers l'approche par modèle utilisateur. Cette méthode présente l'avantage de ne pas nécessiter d'intervention humaine pour tester la cohérence du comportement interactif de notre modèle.

La section 6.1 définit le modèle d'utilisateur nous permettant de simuler des interactions humain-machine sur la tâche de recherche d'information collaborative dans différents contextes. La section 6.2 présente le protocole expérimental suivi pour le déroulement des expériences et donne les métriques d'évaluation utilisées. La section 6.3 présente les résultats des différentes expériences réalisées.

6.1 Comportement conventionnel des participants à l'interaction

Nous appliquons une approche *par simulation* en établissant un modèle de l'utilisateur que l'on fait varier pour tester différents scénarios possibles, afin de générer des suites de motifs dialogiques simulant une interaction entre un expert et un utilisateur sur une tâche de RICM. Nous décrivons dans un premier temps en section 6.1.1 une structure permettant de donner des enchaînements conventionnellement attendus de motifs dialogiques pour la réalisation de la tâche de RICM. Nous expliquons ensuite en section 6.1.2 la variabilité possible des comportements des interlocuteurs au sein de cette structure. Enfin, nous introduisons en section 6.1.3 des altérations de ces comportements conventionnels en nous basant sur des observations faites sur le corpus Cogni-CISMeF.

6.1.1 Enchaînement conventionnel de l'interaction de RICM

Nous nous basons sur la description de la tâche de RICM humain-humain faite dans la section 5.1.2. La transposition de cette tâche pour une interaction humain-machine a amené quelques modifications sur la manière dont s'enchaînent les actions au cours de l'interaction. Notamment, comme discuté dans la section 5.4, l'évaluation des résultats de la requête par l'expert (dans notre cas le système [1]), ne sera pas faite explicitement et n'apparaîtra pas dans cette description. Nous proposons donc ici une version modifiée de la figure 5.1, décrivant la structure conventionnelle de la tâche de RICM. L'automate présenté en figure 6.1 représente les enchaînements conventionnellement attendus entre l'utilisateur et le système. La transition entre les sommets se fait par la réalisation de motifs dialogiques, les sommets représentant différents états de la tâche. Afin de retranscrire les enchaînement conventionnellement attendus, certains états de la tâche peuvent être représentés par plusieurs sommets sur l'automate. Nous considérons que la transition entre deux sommets n'est réalisée que si un des motifs dialogiques qui les relie est réalisé avec succès.

Afin de simplifier la lecture de l'automate, certains motifs dialogiques ont été regroupés sous des appellations génériques. C'est le cas des comportements de modification de la requête (ajout et suppression de mots-clés à la requête par l'utilisateur humain et le système, proposition de substitution de mots-clés par le système), regroupés sous l'étiquette ModifRequête et pour les comportements d'évaluation des résultats (consultation de ressource, marquage d'une ressource comme pertinente ou non par l'utilisateur), regroupés sous l'étiquette ÉvalRésultats.

Deux hypothèses simplificatrices ont été appliquées sur la structure de la tâche par rapport à la description qui en a été donné en section 5.1.2. D'abord, notre prototype n'étant pas doté d'un outil de RI médicale ni d'un outil de traitement automatique des langues, la reformulation de la verbalisation de l'utilisateur par le système a été laissée de côté sur cette figure et ne sera pas évaluée dans la suite de ce chapitre. Ensuite, pour rester cohérent avec les états de la tâche donnés dans la section 5.5, les trois états de la phase de réparation (précision, généralisation, reformulation) ont été regroupés en un seul état de réparation de la requête.

1. Afin d'éviter toute confusion avec l'utilisateur, nous utilisons dans ce chapitre le terme « système » plutôt que « agent » pour désigner l'entité implémentant notre modèle.

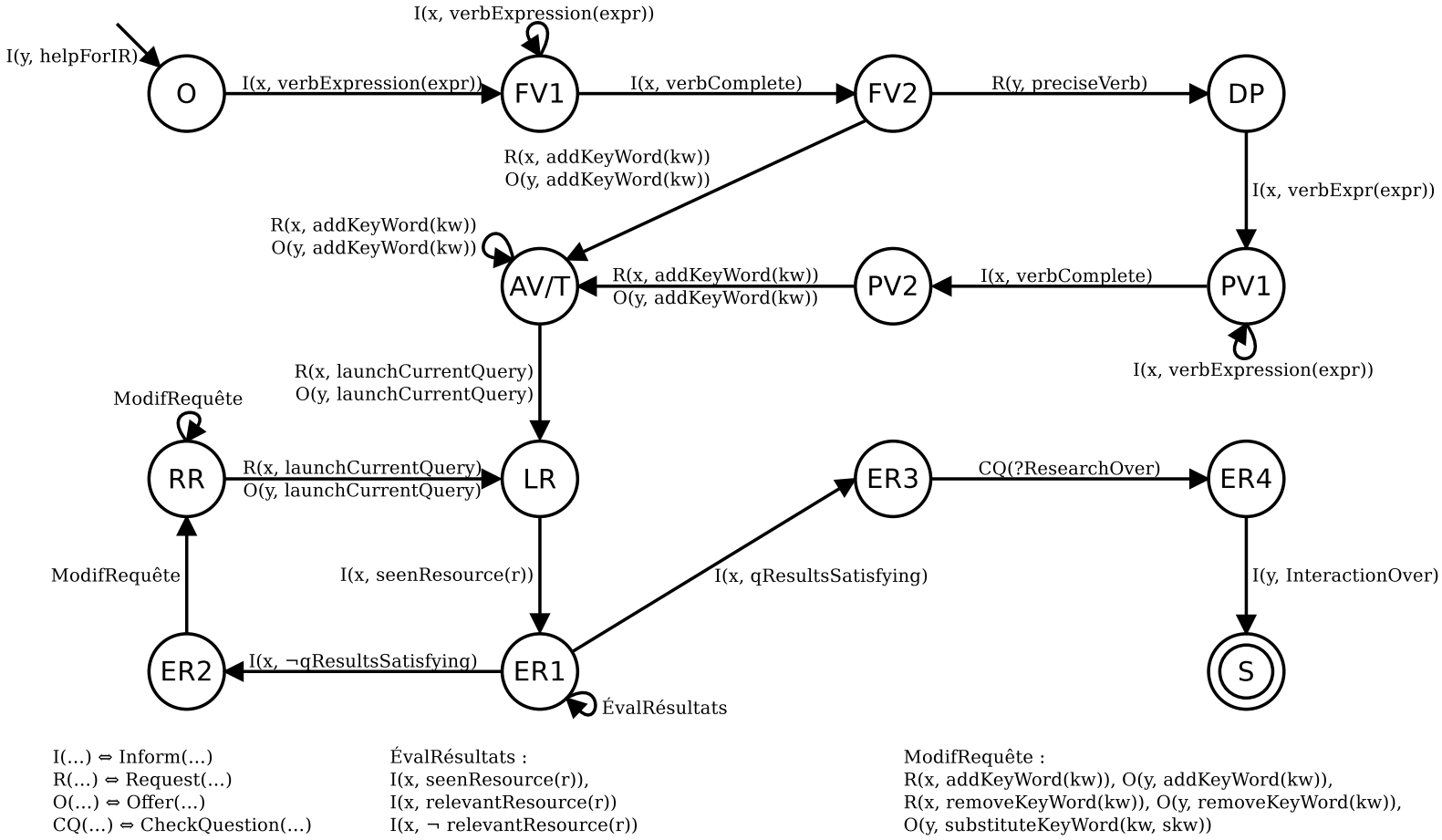

FIGURE 6.1 – Automate permettant de simuler les enchaînements conventionnellement attendus de motifs dialogiques lors d'une interaction humain-machine sur une tâche de RICM. **O** représente l'état d'ouverture, **FV1** et **FV2** représentent l'état de formulation de la verbalisation, **DP** représente l'état de demande de précision, **PV1** et **PV2** représentent l'état de précision de la verbalisation, **AV/T** représente l'état d'alignement verbalisation/terminologie, **LR** représente l'état de lancement de la requête, **ER1**, **ER2**, **ER3** et **ER4** représentent l'état d'évaluation des résultats, **RR** représente l'état de réparation de la requête et **S** représente l'état de sortie. x représente l'utilisateur et y le système.

Ce scénario est une vision idéalisée et synthétique de l'interaction permettant à un utilisateur humain et à notre système de réaliser une tâche de recherche d'information collaborative. Nous discutons dans la suite de cette section de l'utilisation de ce scénario pour simuler différents types d'initiatives et des modifications que l'on peut y apporter pour retranscrire certains phénomènes observés dans le corpus COGNI-CISMEF.

6.1.2 Variation des initiatives prises par les participants

Le scénario présenté en section 6.1.1 donne une vision d'ensemble des enchaînements possibles de coups dialogiques pour la réalisation d'une tâche de RICM. Cependant, celui-ci peut-être décliné selon le type d'initiative que vont prendre les participants. En effet, un certain nombre d'actions sur la tâche (comme le lancement de la requête ou l'ajout d'un mot-clé) peuvent être réalisées à l'initiative de l'utilisateur ou du système. Trois cas de figure sont possibles : l'utilisateur prend la totalité des initiatives, le système prend la totalité des initiatives ou l'initiative est mixte et les deux participants contribuent à l'avancée de la tâche.

Initiative 100% utilisateur

C'est un cas de figure où l'utilisateur est capable de réaliser sa recherche sans assistance. Celui-ci prend la totalité des initiatives, amenant à une situation proche d'une recherche individuelle où l'utilisateur verbaliserait son besoin d'information au début de sa recherche, ses actions et ses évaluations des résultats en cours de recherche.

Dans cette situation, il est intéressant d'étudier la capacité du système à contextualiser correctement les différentes actions de l'utilisateur. Son rôle est purement *réactif* et se limite à répondre collaborativement aux motifs dialogiques ouverts par l'utilisateur, en répondant systématiquement par des actes de dialogue menant au succès du motif, comme des *acceptRequest* ou *agree*. Le système ne sera pas à l'initiative de motifs dialogiques de requête ni d'offre. La seule initiative que le système sera amené à prendre, est de demander à l'utilisateur quand celui-ci aura jugé les résultats de la requête satisfaisants, s'il a terminé sa recherche d'information (le seul comportement réalisable de l'état d'évaluation des résultats, correspondant à un motif de *CheckQuestion*).

Initiative 100% système

Cette configuration est à l'opposée de la précédente : l'utilisateur n'est pas capable de réaliser sa recherche par lui-même et dépend du système. L'utilisateur va donc simplement formuler son besoin d'information au début de l'interaction et ne sera pas capable de proposer de modifications à la requête. Il ne sera pas à l'initiative de motifs dialogiques de requête d'action (par exemple, il ne pourra pas ouvrir le motif Request(x, removeKeyWord(kw))).

Ce sera donc au système d'être *proactif* et de prendre l'initiative d'actions permettant de formuler la requête, de la lancer et de faire d'éventuelles réparations. Il pourra utiliser les motifs d'offre pour proposer à l'utilisateur les modifications lui permettant de faire avancer la recherche. Les seules décisions que l'utilisateur aura à prendre sont d'accepter

ou non les propositions du système, d'évaluer les résultats de la requête et de dire s'ils répondent à son besoin d'information ou non.

Initiative mixte

Le principe de l'initiative mixte est que les participants co-réalisent la tâche sous-jacente au dialogue et que chacun intervient au moment qui lui est le plus opportun, en contribuant par ce qu'il sait le mieux faire à cet instant précis. À la fois l'utilisateur et le système sont capables de contribuer activement à l'avancement de la tâche.

Ces interventions opportunistes au cours de l'interaction peuvent mener à un entrecroisement des participations au dialogue. Ce phénomène permet de se rapprocher d'une structure plus naturelle du dialogue, dont le flot serait moins linéaire et qui verrait s'entrelacer les participations au dialogue.

La volonté de re-créer une interaction *collaborative* implique cependant que chaque participant au dialogue donne toujours la priorité aux réponses attendues par son interlocuteur avant de prendre l'initiative de jouer un de ses propres motifs.

6.1.3 Altération du comportement conventionnel de l'utilisateur

L'étude du corpus Cogni-CISMeF nous a permis d'identifier un certain nombre de comportements de la part des interlocuteurs altérant celui donné dans la section 6.1.1. En effet, celui-ci décrit à l'aide de motifs dialogiques une interaction idéalisée entre un utilisateur ayant un besoin d'information et un expert venant l'aider à répondre à ce besoin d'information. Dans le corpus, certains phénomènes identifiés ne correspondent pas à cette vision idéalisée et nécessitent de la modifier pour les faire apparaître.

Violation d'un engagement en action

Les engagements en action ont un cycle de vie contenant cinq états. Ils peuvent être inactifs (**Ina**), créés (**Crt**), complétés (**Ful**), échoués (**Fal**) ou violés (**Vio**). La création d'un engagement en action se fait de manière explicite à l'aide d'un motif dialogique portant sur une action. La complétion d'un engagement en action a lieu lorsque son contenu en action est réalisé par son débiteur. Elle peut être faite explicitement avec l'utilisation d'un *jeu de décharge* afin de permettre au débiteur de signaler l'exécution de l'action. Ce comportement est cependant très « verbeux », dans le sens où il implique l'exécution d'un jeu pour chaque action réalisée.

Lors de sa conception, nous avons fait l'hypothèse que le système était capable de réaliser les actions instantanément après s'y être engagé, et donc que les engagements en action qu'il contracte sont directement complétés. Ce mécanisme n'est pas applicable pour l'utilisateur, qui peut s'engager à réaliser une action puis ne pas la réaliser. Dans le cadre du scénario de RICM, le seul engagement en action que peut contracter l'utilisateur est le fait de préciser sa verbalisation. La satisfaction de cet engagement en action est réalisée automatiquement lorsque l'utilisateur suit le scénario présenté en figure 6.1. La violation de cet engagement a lieu lorsque l'utilisateur s'engage à préciser sa verbalisation puis qu'il

ré-affirme que celle-ci est terminée sans l'avoir modifiée. Le système doit être capable de détecter cette violation et de continuer à contextualiser correctement ses actions et à y répondre de manière pertinente.

Le diagramme 6.2 présente l'arête que ce type de comportement introduit dans l'automate 6.1, amenant l'utilisateur à ne pas réaliser une partie des comportements attendus de sa part dans l'état de précision de la verbalisation (tous ceux relatifs au sommet PV1 dans l'automate).

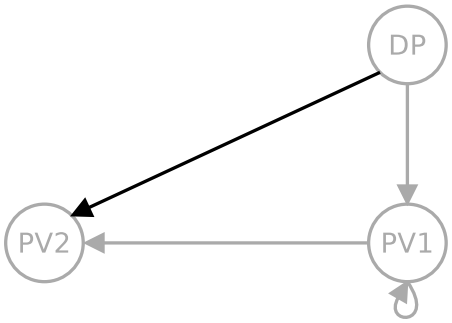

FIGURE 6.2 – Illustration de la modification apportée sur l'automate par la violation de l'engagement sur la précision de sa verbalisation par l'utilisateur. En gris figure la partie concernée de l'automate et en noir la nouvelle arrête introduite par cette modification.

Modification du besoin d'information

Dans le corpus COGNI-CISMEF, l'utilisateur cherchant des informations médicales pouvait être amené à spécifier, modifier ou reformuler son besoin d'information au cours du dialogue. Le contexte principal dans lequel ce type de phénomène avait lieu le fait sortir de la boucle lancement/évaluation/réparation de la requête suite à un lancement ou à une évaluation peu concluante pour ajouter des informations à la formulation de son besoin d'information initial. Cela traduit le fait que le besoin d'information de l'utilisateur évolue pendant sa recherche, et qu'il peut être nécessaire de le mettre à jour au cours de l'interaction. Cependant, cela n'entraîne pas une reprise de la tâche à partir de cet état passé du scénario, mais plutôt fait faire un « détour » par l'état de précision de la verbalisation du besoin d'information afin d'actualiser les informations qui y avaient été données en premier lieu. Cela implique que cette modification du parcours de l'utilisateur n'est que temporaire et qu'il ne faut pas parcourir à nouveau l'intégralité du scénario pour en revenir à l'état de l'interaction précédant cette discontinuité. Dans le cas d'une mise à jour de la verbalisation par l'utilisateur, une fois celle-ci terminée, l'état suivant dans lequel l'interaction doit se trouver est l'évaluation des résultats.

Le diagramme 6.3 présente les modifications que ce type de comportement introduit dans l'automate 6.1, nécessitant l'introduction d'un état supplémentaire à l'automate pour permettre à l'utilisateur de temporairement retourner dans l'état de précision de la verbalisation.

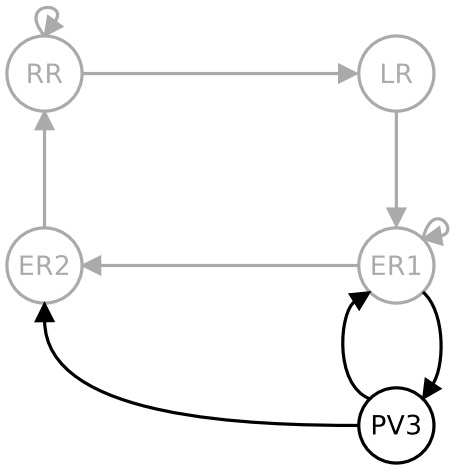

FIGURE 6.3 – Illustration de la modification apportée sur l'automate par la précision de son besoin d'information par l'utilisateur lors de l'évaluation des résultats d'une requête. En gris figure la partie concernée de l'automate et en noir la nouvelle arrête introduite par cette modification.

Rejet implicite des résultats de la requête

Les participants aux dialogues du corpus COGNI-CISMEF étant co-localisés, ils étaient parfois amenés à ne pas verbaliser l'intégralité de leurs actions, plus particulièrement lorsque ces actions pouvaient être comprises de manière implicite. C'est le cas de l'évaluation des résultats de la requête. Certains dialogues du corpus montrent un enchaînement direct entre le lancement de la requête et sa réparation, sans qu'un acte de dialogue d'évaluation ne soit joué, ni sur les résultats, ni sur la requête elle-même. Cependant, l'utilisateur sait que le fait de modifier la requête rejette implicitement les résultats renvoyés.

L'introduction de ce phénomène dans la simulation du comportement de l'utilisateur signifie qu'il est possible, suite au lancement d'une requête, que celui-ci ne formule aucun des motifs d'Inform attendus dans l'état d'évaluation des résultats et qu'il passe directement à la modification de celle-ci.

Le diagramme 6.4 présente l'arête que ce type de comportement introduit dans l'automate 6.1, permettant à l'utilisateur de passer directement de l'état de lancement de la requête à l'état de réparation de la requête sans passer par les états intermédiaires ER2 et ER3.

Saut en avant

En généralisant l'idée de l'évaluation implicite des résultats, nous introduisons la possibilité que l'utilisateur fasse un « saut en avant » n'importe où dans l'automate. Lorsqu'il lui est possible de prendre une initiative, celui-ci pourra réaliser implicitement son prochain coup et jouer explicitement le coup d'après.

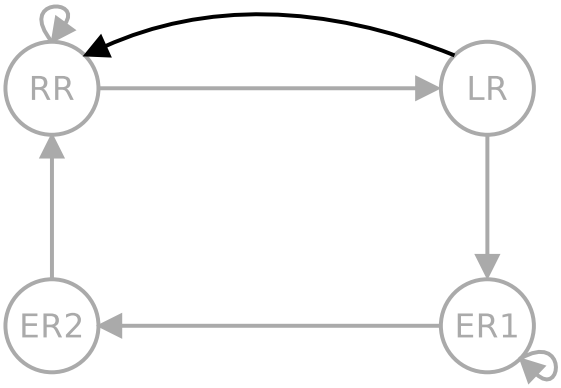

FIGURE 6.4 – Illustration de la modification apportée sur l'automate par le rejet implicite des résultats de la requête par l'utilisateur. En gris figure la partie concernée de l'automate et en noir la nouvelle arrête introduite par cette modification.

6.2 Protocole expérimental

Nous décrivons maintenant le cadre dans lequel nous allons réaliser les différentes simulations des interactions entre notre système et le modèle d'utilisateur.

Dans un premier temps, en section 6.2.1, nous expliquons comment le système va être capable de calculer ses prédicats relatifs au domaine de la tâche. Ensuite, en section 6.2.2, nous décrivons comment l'automate présenté en figure 6.1 va être exploité pour simuler un utilisateur humain. Enfin, en section 6.2.3, nous présentons les métriques qui seront utilisées pour évaluer le système.

6.2.1 Calcul des prédicats

Les prédicats sont un élément clé des processus décisionnels du système. Il ont une influence lors de l'interprétation des actions de l'utilisateur et lors de la prise d'initiative de comportements par le système.

Un certain nombre de prédicats utilisés par le système portent uniquement sur l'état de l'interaction et de la tâche, c'est par exemple le cas de VerbalisationComplete et de CurrentQueryLaunched. Ces prédicats sont purement descriptifs et leur calcul est simple et ne dépend que d'engagements pris au cours du dialogue. Nous considérons donc que le calcul de ces prédicats sera toujours effectué correctement.

En revanche, les prédicats permettant de modifier la requête comme RelevantForAddition(kw) ou RelevantForSubstitution(kw, skw) nécessitent de la part du système une utilisation d'outils propres au domaine d'application de la tâche comme des dictionnaires de synonymes ou d'outils de traitement automatique des langues. Pour éviter le biais que pourrait amener l'utilisation d'une ressource extérieure au cours de nos expériences, nous simulons ces outils du domaine à l'aide de deux dictionnaires enrichis au fur et à mesure du dialogue pour permettre au système de calculer les prédicats de ce type. Le premier permet de lier une expression de la verbalisation de l'utilisateur à des mots-clés du moteur

de recherche, et l'autre étant un dictionnaire de synonymes. Par exemple, l'occurrence d'un acte de dialogue inform(x, verbExpression(*expr*)), introduira une ou plusieurs entrées associées à la clé *expr* dans le dictionnaire faisant correspondre des mots-clés de la terminologie à des expressions de la verbalisation. De même si l'utilisateur demande un ajout de mot-clé à la requête avec un request(x, addKeyWord(kw)), au moins un synonyme de kw sera ajouté dans le dictionnaire des synonymes.

6.2.2 Simulations des dialogues

Nous simulons un environnement dialogique pour la réalisation de nos expériences. Cet environnement permet à notre système et à l'utilisateur d'échanger des coups dialogiques afin de réaliser un dialogue. L'algorithme 4 présente l'algorithme d'un tour de parole dans l'environnement dans lequel le dialogue a lieu. Au cours de l'interaction, chaque interlocuteur a sa propre représentation de l'état courant de la tâche dans lequel il se trouve. Cet état change tout au long de la réalisation de la tâche en fonction des coups dialogiques échangés. En effet, l'utilisateur peut changer d'état courant lors de l'appel de `ProchainCoup()` (l. 5 de l'algorithme) et le système peut changer d'état courant lorsqu'il contextualise un coup dialogique de l'utilisateur au cours de l'appel de `NouveauCoupUtilisateur()` (l. 6 de l'algorithme). Nous considérons que cette fonction renvoie l'issue de la contextualisation réalisée à ce moment-là, qui peut être réussie ou échouée.

Lors de son appel à `JouerProchainCoup()` (l. 8 de l'algorithme), le système doit choisir le prochain coup à jouer. Celui-ci correspond à l'algorithme 3. Lors de la description de cet algorithme, nous avons précisé que c'est au concepteur du système final d'implémenter la fonction `SélectionnerMeilleurCoup()` qui sélectionne le meilleur coup parmi les coups conventionnellement attendus de la part du système. Nous avons implémenté cette fonction très simplement en faisant en sorte que celui-ci sélectionne en premier les coups qui ferment un motif dialogique, puis ceux qui correspondent au déroulement d'un motif dialogique et enfin ceux qui répondent à une proposition d'ouverture d'un motif dialogique par l'utilisateur.

L'algorithme illustre ligne 10 le principe de *décrochage*, qui se produit lorsque, suite à une mauvaise contextualisation d'une action de l'utilisateur, l'état du système est différent de l'état de l'utilisateur. De plus, celui-ci illustre aussi le principe *d'échec de contextualisation*, qui peut avoir lieu lorsque le système n'arrive pas à contextualiser un coup de l'utilisateur (cf. algorithme 2).

Nous utilisons une approche procédurale pour générer l'acte de dialogue que l'utilisateur peut émettre à chaque tour de parole selon l'automate présenté en figure 6.1. À chaque instant de l'interaction, l'utilisateur se trouve sur un état de l'automate, son état initial étant celui d'ouverture. Lorsque l'utilisateur peut prendre l'initiative d'un comportement, il se sert des arêtes sortantes de l'état sur lequel il se trouve. Dans les cas où il y a plusieurs arêtes sortantes, l'utilisateur simulé tire au hasard parmi celles-ci en tenant compte des probabilités qui leur sont attribuées. Les chances de tirage pour les différentes transitions possibles entre états sont décrites dans le tableau 6.1. Elles peuvent être modifiées dans le cas des altérations du comportement de l'utilisateur.

Algorithme 4 : Simulation d'un tour de parole dans l'environnement

Entrées : Ordonnanceur : ordonnanceur des tours de parole, Utilisateur : utilisateur humain simulé, Système : système assistant l'utilisateur à la réalisation de sa tâche.

Sorties : Décrochage : booléen représentant le fait que le système est en décrochage ou non, RésultatContextualisation : issue de la contextualisation si celle-ci a été réalisée par le système.

```
1  Locuteur ← Ordonnanceur.ProchainLocuteur();
2  Décrochage ← Faux;
3  RésultatContextualisation ← Succès;
4  si Locuteur = Utilisateur alors
       // Cet appel peut faire changer l'état courant de l'utilisateur
5      CoupUtilisateur ← Utilisateur.ProchainCoup();
       // Cet appel peut faire changer l'état courant du système
6      RésultatContextualisation ← Système.NouveauCoupUtilisateur(CoupUtilisateur);
7  sinon
8      Système.JouerProchainCoup();
9  si Utilisateur.ÉtatCourant ≠ Système.ÉtatCourant alors
10     Décrochage ← Vrai;
11 Retourner Décrochage, RésultatContextualisation
```

Si c'est le système qui prend l'initiative d'un comportement, l'utilisateur se déplacera sur l'automate en suivant l'arrête correspondant au contenu de l'initiative prise par le système. S'il n'existe pas d'arête correspondante, l'utilisateur restera sur l'état où il se trouve.

Les probabilités ont été choisies à partir des observations faites dans le corpus et dans le but de faire apparaître des phénomènes que l'on souhaite observer au cours des simulations. Nous avons fait en sorte que la verbalisation de l'utilisateur se termine plutôt rapidement de manière à ce que le système puisse faire une demande de précision de la verbalisation. Nous avons aussi fait en sorte que la première formulation de la requête amène les interlocuteurs à écrire une requête plutôt riche, alors que pour la réparation de la requête le nombre de modifications apportées à celle-ci avant qu'elle soit à nouveau lancée est souvent plus faible. De la même manière, dans le corpus, l'évaluation des résultats était plutôt longue et l'expression de la satisfaction ou non-satisfaction de l'utilisateur n'arrivait qu'après un certain nombre de tours de parole, ce qui explique la faible probabilité associée à chacune de ces options.

L'ordonnancement des coups dialogiques se fait en alternant les deux interlocuteurs, ce qui modifie légèrement la boucle principale de l'agent décrite dans l'algorithme 1 en faisant en sorte que celui-ci puisse émettre au plus un coup dialogique entre deux coups de l'utilisateur. Le premier tour de l'interaction est donné au système car c'est lui qui ouvre l'interaction. De manière à éviter que ce soit toujours le même interlocuteur qui prenne les initiatives au cours du dialogue, chaque interlocuteur a une chance sur quatre de passer son tour. Dans ce cas, il joue un coup vide et c'est son interlocuteur qui peut prendre l'initiative. Cette proportion a été choisie comme un compromis entre le suivi du

Départ	Contenu sémantique	Arrivée	Probabilité
O	verbExpression(*expr*)	FV1	1
FV1	verbExpression(*expr*)	FV1	1/2
FV1	verbComplete	FV2	1/2
DP	verbExpression(*expr*)	PV1	1
PV1	verbExpression(*expr*)	PV1	1/2
PV1	verbComplete	PV2	1/2
FV2	addKeyWord(*kw*)	AV/T	1
AV/T	addKeyWord(*kw*)	AV/T	2/3
AV/T	launchCurrentQuery	LR	1/3
LR	seenResource(*res*)	ER1	1
ER1	seenResource(*res*)	ER1	1/3
ER1	relevantResource(*res*)	ER1	1/3
ER1	currentQueryResultsSatisfying	ER3	1/6
ER1	¬ currentQueryResultsSatisfying	ER2	1/6
ER2	addKeyWord(*kw*)	RR	1/2
ER2	removeKeyWord(*kw*)	RR	1/2
RR	addKeyWord(*kw*)	RR	1/4
RR	removeKeyWord(*kw*)	RR	1/4
RR	launchCurrentQuery	LR	1/2

TABLEAU 6.1 – Probabilités de tirage des contenus sémantique et sommets correspondants en fonction du sommet de départ.

déroulement de la tâche par un interlocuteur et l'aspect interactif du dialogue permettant aux deux interlocuteurs de prendre des initiatives à tour de rôle.

Nous considérons que l'utilisateur répondra toujours au système de manière à ce que les motifs dialogiques ouverts atteignent leurs conditions de succès.

L'aléatoire permettra de générer une grande variété d'actions de la part de l'utilisateur tout en respectant les règles définissant son comportement. Les évaluations seront réalisées en répétant un grand nombre de fois l'exécution d'un dialogue entre le système et un utilisateur simulé pour représenter un cas de figure présenté dans la section 6.1.

6.2.3 Métriques d'évaluation

Nous avons réalisé une étude quantitative du système. Pour ce faire, nous introduisons plusieurs métriques sur lesquelles nous nous appuierons lors de l'évaluation des différents attributs de notre système.

Proportion d'initiatives prises dans chaque état

Chaque comportement initié par un interlocuteur est comptabilisé au cours des expériences. Afin de donner une perspective globale sur le déroulement de l'interaction, le rôle de chaque interlocuteur et le temps passé dans les différents états, les initiatives sont différenciées en fonction des états auxquels les comportements qui leurs sont associés correspondent.

L'état de référence pour comptabiliser une initiative est celui dans lequel se trouve l'utilisateur simulé suite à la contextualisation par celui-ci de l'initiative en question. Les initiatives prises par les interlocuteurs sont donc vues du point de vue de l'utilisateur.

Nous introduisons les métriques suivantes :

I_u : proportion d'initiatives prises par l'utilisateur ;

I_s : proportion d'initiatives prises par le système.

Contextualisation et décrochages

Pour chaque coup dialogique joué par l'utilisateur lorsqu'il prend l'initiative d'un comportement, trois cas de figure sont possibles :

— celui-ci est bien contextualisé par le système. Le système a correctement interprété l'acte de dialogue de l'utilisateur ;

— celui-ci est mal contextualisé par le système. Le système a interprété l'acte de dialogue de l'utilisateur, mais ne l'a pas contextualisé dans l'état approprié ;

— celui-ci n'est pas contextualisé par le système. Le système n'a pas été capable de trouver un comportement possible où contextualiser l'acte de dialogue de l'utilisateur.

Ces deux derniers cas de figure mènent à un *décrochage*, c'est-à-dire une divergence entre les états dans lesquels se trouvent l'utilisateur et le système.

Les échecs de contextualisation sont des événements graves car ils correspondent à un échec sur la tâche et sur l'interaction. Lorsqu'il échoue à contextualiser une initiative de l'utilisateur, le système est incapable d'y répondre et ne respecte donc pas les conventions d'interaction attendues par l'utilisateur. Nous associons donc aux échecs de contextualisation les métriques suivantes :

$C_{\varnothing d}$: proportion de dialogues contenant des échecs de contextualisation parmi les dialogues étudiés ;

$C_{\varnothing}$: proportion d'initiatives de l'utilisateur dont la contextualisation a échoué.

Ces métriques calculent les proportions d'initiatives dont la contextualisation a échoué par rapport au nombre d'initiatives prises par l'utilisateur dans la totalité des dialogues étudiés. Nous introduisons une métrique supplémentaire se focalisant sur les dialogues contenant des échecs de contextualisation :

$C_{\varnothing/\varnothing}$: proportion d'initiatives de l'utilisateur dont la contextualisation a échoué parmi les dialogues contenant des échecs de contextualisation.

Une métrique mesurant les proportions d'initiatives bien et mal contextualisées n'est pas pertinent dans le contexte de ces expériences. En effet, une initiative de l'utilisateur mal contextualisée est un phénomène instantané et il est plus intéressant d'étudier les effets qu'une mauvaise contextualisation peut avoir, notamment les caractéristiques des décrochages qui s'en suivent. Un décrochage est temporaire et la suite des actions de l'utilisateur peut permettre au système de rattraper son écart et de compenser la divergence qui a eu lieu en changeant à nouveau d'état pour retrouver celui où se trouve l'utilisateur. Il est alors intéressant de mesurer la capacité du système à se rattraper après avoir mal contextualisé une action de l'utilisateur.

Nous introduisons donc des métriques supplémentaires afin de caractériser les décrochages réalisés par le système :

$D_{\#}$: proportion de dialogue contenant des décrochages parmi les dialogues étudiés ;

D_r : rapport entre le nombre de décrochages et le nombre d'altérations du comportement conventionnel de l'utilisateur ;

D_l : longueur moyenne des décrochages ;

D_p : proportion moyenne des décrochages par rapport à la durée totale des dialogues.

Type de transition réalisée

Le système implémente deux types de transitions possibles entre les états de la tâche :

— les transitions vers des états candidats : les prérequis de l'état destination sont satisfaits. C'est le type de transition privilégié par le système ;

— les transitions vers des états matures : si aucun état candidat ne permet de réaliser une transition, alors le système se tourne vers les états dont la maturité est non nulle ou les états ne comportant pas de comportements possibles instanciés.

Nous introduisons les métriques suivantes :

T_c : proportion de transitions réalisées vers des états candidats ;

T_m : proportion de transitions réalisées vers des états matures.

Nous raffinons T_m en deux métriques distinguant si la transition réalisée vers un état mature a mené à un décrochage ou non :

T_{ms} : proportion de transitions vers des états matures réalisées avec succès (c'est-à-dire n'ayant pas mené à un décrochage) ;

T_{md} : proportion de transitions vers des états matures ayant mené à un décrochage.

Inconsistance logique

Lorsqu'il calcule les prédicats déduits sur le tableau de conversation, le système vérifie qu'il n'y a pas d'inconsistance logique dans les engagements pris par les différents participants. Il s'assure notamment que l'utilisateur n'est pas engagé sur deux propositions opposées (formellement, C(x, p, **Crt**) et C(x, ¬p, **Crt**) en même temps), comme le fait qu'il est à la fois satisfait et non satisfait par les résultats de la requête courante. Dans le cas où une telle contradiction est détectée, le système n'est pas capable de continuer à raisonner de manière correcte sur la tâche et de calculer les prédicats nécessaires à sa prise de décision. De ce fait, il s'arrête et ne mène pas l'interaction à son terme.

Nous introduisons la métriques suivante :

$L_\perp$: proportion de dialogues abandonnées par le système suite à une inconsistance logique.

6.3 Analyse des résultats des expérimentations

Cette section présente l'étude faite du déroulement de dialogues simulés selon le protocole décrit en section 6.2 entre notre système et un utilisateur suivant le comportement décrit en section 6.1. La section 6.3.1, présente l'analyse des dialogues simulés en variant le type d'initiative que les interlocuteurs peuvent prendre sans que l'utilisateur ne puisse altérer son comportement, ainsi que les dialogues simulant la violation d'un engagement en action par l'utilisateur. En section 6.3.2, nous analysons les dialogues simulés où les altérations apportées par l'utilisateur à son comportement conventionnel peuvent avoir mené à des erreurs de la part du système.

6.3.1 Initiative utilisateur/système/mixte et violation d'un engagement en action

Nous présentons dans cette section l'étude du déroulement des dialogues pour les différentes modalités d'initiatives que peuvent prendre les participants. L'objectif est simplement de montrer que le système fonctionne dans le cas où l'utilisateur suit un comportement conventionnel sans altération, qu'il soit ou non à l'initiative de comportements faisant avancer la tâche.

Les résultats donnés dans cette section sont donc des résultats préliminaires, illustrant trois cas de figure du déroulement conventionnel de la tâche que nous souhaitons

simuler : l'initiative système, l'initiative utilisateur et l'initiative mixte. Les métriques principalement utilisées sont celles mesurant les initiatives prises par le système (I_s) et par l'utilisateur (I_u) dans chaque état.

Initiative utilisateur

16000 dialogues ont été simulés en supprimant au système la capacité de réaliser des jeux d'offre et de requête. La taille moyenne d'un dialogue simulé dans ce contexte est de 73,34 coup dialogiques. La totalité des dialogues simulés s'est terminée correctement dans l'état de fermeture suite à une satisfaction du besoin d'information de l'utilisateur. L'intégralité des 143565 transitions entre états ayant eu lieu lors de ces simulations ont été faites vers des états candidats ($T_c = 100\%$). Le tableau 6.2 présente une synthèse des initiatives d'ouverture de comportements possibles prises par chaque interlocuteur dans les différents états de la tâche. Un dialogue illustrant le type de dialogue simulé est présent en annexe B.1.1 tableau B.1.

État	I_s	I_u	Total
Ouverture	16000 (100,00%)	0 (0,00%)	16000
Formulation de la verbalisation	0 (0,00%)	48067 (100,00%)	48067
Alignement V/T	0 (0,00%)	47651 (100,00%)	47651
Lancement de la requête	0 (0,00%)	31855 (100,00%)	31855
Évaluation des résultats	16000 (11,10%)	128209 (88,90%)	144209
Réparation de la requête	0 (0,00%)	30639 (100,00%)	30639
Sortie	16000 (100,00%)	0 (0,00%)	16000
Total	48000 (14,35%)	286421 (85,65%)	334421

TABLEAU 6.2 – Synthèse des initiatives prises par les interlocuteurs pour chaque état dans le cas 100% initiative utilisateur.

Cette table ne présente pas les états de demande de précisions ni de précision de la verbalisation. Le premier ne contient qu'un comportement réalisable par le système (faisant la requête à l'utilisateur de préciser son besoin d'information) et ne sera donc jamais joué. De ce fait, étant donné que l'utilisateur respecte la structure de l'automate de la figure 6.1, celui-ci ne précisera jamais sa verbalisation.

Nous remarquons ici que les initiatives prises par le système se restreignent à une occurrence d'un comportement réalisable dans trois états. Le premier état est celui d'ouverture, qui a pour seule fonction de permettre au système de débuter la conversation. Le deuxième état est celui d'évaluation des résultats, où le système se contente d'initier le comportement vérifiant que l'utilisateur a terminé sa recherche une fois que celui-ci a exprimé la satisfaction de son besoin d'information par les résultats de la requête courante. Le troisième état est celui de fermeture de l'interaction qui permet au système de terminer la conversation une fois que l'utilisateur juge son besoin d'information satisfait. Aucun de ces trois états n'a d'impact sur le déroulement de la tâche de recherche d'information en elle-même et c'est uniquement l'utilisateur qui réalise celle-ci.

Initiative système

16000 dialogues ont été simulés en supprimant à l'utilisateur la capacité de proposer des modifications à la requête. La taille moyenne d'un dialogue simulé dans ce contexte est de 96,04 coup dialogiques. La totalité des dialogues simulés s'est terminée correctement dans l'état de fermeture suite à une satisfaction du besoin d'information de l'utilisateur. L'intégralité de 168556 transitions entre états ayant eu lieu lors de ces simulations ont été faites vers des états candidats ($T_c = 100\%$), c'est-à-dire des états dont les prérequis étaient satisfaits. Le tableau 6.3 présente une synthèse des initiatives d'ouverture de comportements possibles prises par chaque interlocuteur dans les différents états de la tâche. Un dialogue illustrant le type de dialogue simulé est présent en annexe B.1.2 dans les tableaux B.2 et B.3.

État	I_s	I_u	Total
Ouverture	16000 (100,00%)	0 (0,00%)	16000
Formulation de la verbalisation	0 (0,00%)	47648 (100,00%)	47648
Demande de précisions	12053 (100,00%)	0 (0,00%)	12053
Précision de la verbalisation	0 (0,00%)	36650 (100,00%)	36650
Alignement V/T	76020 (100,00%)	0 (0,00%)	76020
Lancement de la requête	32150 (100,00%)	0 (0,00%)	32150
Évaluation des résultats	16000 (11,07%)	128592 (88,93%)	144592
Réparation de la requête	38262 (100,00%)	0 (0,00%)	38262
Sortie	16000 (100,00%)	0 (0,00%)	16000
Total	206485 (49,24%)	212890 (50,76%)	419375

TABLEAU 6.3 – Synthèse des initiatives prises par les interlocuteurs pour chaque état dans le cas 100% initiative système.

L'utilisateur ne prend d'initiatives que dans les états de formulation de la verbalisation, de précision de la verbalisation et d'évaluation des résultats. Les deux premiers sont des états où seul l'utilisateur peut initier des comportements et le troisième est celui où il exprime ses opinions sur les résultats de la requête.

La possibilité de passer par les états de demande de précision et de précision de la requête explique que les dialogues sont plus longs que ceux de l'expérience précédente, ainsi que le fait que le système prends plus de temps que l'utilisateur à faire l'alignement verbalisation/terminologie et un peu plus de temps à réparer la requête.

Ces dialogues simulés montrent la capacité du système à guider un utilisateur qui ne serait pas capable de chercher une réponse à un besoin d'information par lui-même.

Initiative mixte

16000 dialogues ont été simulés en permettant à la fois à l'utilisateur et au système de prendre l'initiative d'ouvrir des comportements. La taille moyenne d'un dialogue simulé selon cette modalité est de 81,73 coup dialogiques. La totalité des dialogues simulés

s'est terminée correctement dans l'état de fermeture suite à une satisfaction du besoin d'information de l'utilisateur. L'intégralité des 148906 transitions entre états ayant eu lieu lors de ces simulations ont été faites vers des états candidats ($T_c = 100\%$). Le tableau 6.4 présente une synthèse des initiatives d'ouverture de comportements possibles prises par chaque interlocuteur dans les différents états de la tâche. Un dialogue illustrant le type de dialogue simulé est présent en annexe B.1.3 dans les tableaux B.4 et B.5.

État	I_s	I_u	Total
Ouverture	16000 (100,00%)	0 (0,00%)	16000
Formulation de la verbalisation	0 (0,00%)	47794 (100,00%)	47794
Demande de précisions	2441 (100,00%)	0 (0,00%)	2441
Précision de la verbalisation	0 (0,00%)	7273 (100,00%)	7273
Alignement V/T	17167 (31,92%)	36621 (68,08%)	53788
Lancement de la requête	9671 (30,21%)	22337 (69,79%)	32008
Évaluation des résultats	16000 (11,08%)	128440 (88,92%)	144440
Réparation de la requête	18099 (43,23%)	23766 (56,77%)	41865
Sortie	16000 (100,00%)	0 (0,00%)	16000
Total	95378 (26,38%)	266231 (73,62%)	361609

TABLEAU 6.4 – Synthèse des initiatives prises par les interlocuteurs pour chaque état dans le cas de l'initiative mixte.

Les seuls états où l'un ou l'autre des interlocuteurs peut avoir 100% des initiatives sont les états ne comportant que des comportements réalisables ou que des comportements attendus.

Au cours des trois expériences précédentes, aucune inconsistance logique n'a eu lieu ($L_\perp = 0\%$), toutes les transitions ont été réalisées vers des états candidats ($T_c = 100\%$) et donc aucun décrochage n'a eu lieu ($D_\# = 0\%$). Nous avons donc un fonctionnement idéal du système montrant sa capacité à suivre un utilisateur réalisant une tâche de recherche d'information et à prendre des initiatives pour l'aider à avancer, dans le cas où celui-ci aurait un comportement respectant la structure définie dans l'automate de la figure 6.1.

Dans la suite des expériences présentées dans ce chapitre, afin de se rapprocher au plus d'une situation réelle, les dialogues sont simulés dans des conditions d'initiative mixte.

Violation d'un engagement en action

Nous présentons maintenant l'analyse des dialogues simulés dans le cas où l'utilisateur peut violer un engagement sur le fait qu'il va préciser sa verbalisation. Afin de rendre visible le phénomène de violation d'un engagement en action par l'utilisateur, nous avons fixé à 50% la probabilité que cela ait lieu lorsqu'il est engagé sur une action. Au sein des dialogues simulés, nous conservons ceux contenant au moins une violation d'un engagement en action de l'utilisateur. Sur les 20000 dialogues réalisés au cours de cette expérience, 1495 (7.475%) contiennent des altérations réalisées par l'utilisateur. Cette faible proportion

s'explique par le fait qu'il y a plusieurs conditions à réunir pour que l'utilisateur fasse cette altération. Il faut d'abord que l'utilisateur formule une verbalisation que le système considère comme insuffisante. Ensuite, il faut que l'utilisateur cède la parole au système afin que celui-ci puisse faire la demande de préciser sa verbalisation. Enfin, il faut que l'utilisateur décide de ne pas respecter cet engagement en action.

Un dialogue illustrant le type de dialogue simulé est présent en annexe B.1.4 dans les tableaux B.6 et B.7. Plus spécifiquement, le tableau 6.5 montre le passage où l'utilisateur viole son engagement. Les actes 8 à 13 correspondent au motif dialogique de requête de la part du système sur l'action de préciser sa verbalisation, que l'utilisateur accepte (acte 11). La violation à lieu à l'acte 14, où l'utilisateur, plutôt que de préciser sa verbalisation, ré-affirme que celle-ci est terminée.

	Acte	E_s	E_h
8	prop.in(Agent-XorG, Request(preciseVerbalization.))	FV	FV
9	acc.in(Human-John, Request(preciseVerbalization.))	RP	RP
10	request(Agent-XorG, preciseVerbalization.)	RP	RP
11	acceptRequest(Human-John, preciseVerbalization.)	RP	RP
12	prop.out(Agent-XorG, Request(preciseVerbalization.))	RP	RP
13	acc.out(Human-John, Request(preciseVerbalization.))	RP	RP
14	inform(Human-John, verbalizationComplete.)	PV	PV

TABLEAU 6.5 – Extrait du dialogue simulé numéro 908880938 illustrant la violation d'un engagement par l'utilisateur. E_s est l'état du système à chaque tour de parole et E_h est l'état de l'utilisateur à chaque tour de parole.

La taille moyenne des 1495 dialogues présentant une violation de son engagement par l'utilisateur est de 86,54 coups, soit environ cinq coups de plus que pour l'initiative mixte sans violation. Cette différence s'explique par le fait que l'interaction passe systématiquement par l'état de précision de la verbalisation, ce qui ajoute quelques tours de parole. Le tableau 6.6 présente la répartition des initiatives prises par les différents interlocuteurs au cours des 1495 dialogues simulés contenant une altération. Nous remarquons que ces proportions sont très proches de celles de l'expérience sans violation.

Aucune inconsistance logique n'a eu lieu ($L_\perp = 0\%$), toutes les transitions on été réalisées vers des états candidats ($T_c = 100\%$) et donc aucun décrochage n'a eu lieu ($D_\# = 0\%$). Cet exemple illustre la capacité du système à continuer de fonctionner correctement malgré une altération du comportement conventionnel de l'utilisateur.

6.3.2 Mise à jour du besoin d'information, rejet implicite et saut en avant

Cette section présente les expériences portant sur les mises à jour du besoin d'information, de rejet implicite des résultats d'une requête et de saut en avant. Nous avons donné une probabilité à chaque altération du comportement conventionnel de l'utilisateur. Cette probabilité est choisie pour faire apparaître l'altération apportée au comportement un

État	Initiatives système	Initiatives utilisateur	Total
Ouverture	1495 (100,00%)	0 (0,00%)	1495
Formulation de la verbalisation	0 (0,00%)	3495 (100,00%)	3495
Demande de précisions	1495 (100,00%)	0 (0,00%)	1495
Précision de la verbalisation	0 (0,00%)	1495 (100,00%)	1495
Alignement V/T	1645 (33,85%)	3214 (66,15%)	4859
Lancement de la requête	904 (29,99%)	2110 (70,01%)	3014
Évaluation des résultats	1495 (10,94%)	12176 (89,06%)	13671
Réparation de la requête	1618 (40,64%)	2363 (59,36%)	3981
Sortie	1495 (100,00%)	0 (0,00%)	1495
Total	10147 (28,99%)	24853 (71,01%)	35000

TABLEAU 6.6 – Synthèse des initiatives prises par les interlocuteurs pour chaque état dans le cas où l'utilisateur peut violer un engagement en action.

nombre significatif de fois lors de simulations tout en retranscrivant les observations faites lors de l'analyse du corpus. Dans la modalité de modification du besoin d'information, l'utilisateur fera une précision de sa verbalisation dans 25% des initiatives qu'il prendra lorsqu'il se trouvera dans l'état d'évaluation des résultats. Dans la modalité de rejet implicite des résultats, nous fixons la possibilité pour l'utilisateur de faire un rejet implicite à 50%. Dans modalité de saut en avant, nous fixons la probabilité pour l'utilisateur de faire un saut en avant quand il a l'initiative à 25%.

Nous nous focalisons uniquement sur les dialogues comportant des altérations du comportement conventionnel de l'utilisateur et ignorons les autres. Le tableau 6.7 présente le nombre de dialogues présentant des altération au comportement conventionnel de l'utilisateur sur l'ensemble des dialogues simulés réalisés pour chaque modalité.

Modalité	$\#S$	$\#S_a$
Mise à jour du besoin d'information	20000	13330 (66.65%)
Rejet implicite des résultats de la requête	20000	12727 (63.635%)
Saut en avant	20000	17927 (89.64%)

TABLEAU 6.7 – Nombre de dialogues présentant des altérations sur l'ensemble des dialogues simulés pour chaque modalité. $\#S$ représente le nombre de dialogues simulés et $\#S_a$ représente le nombre de dialogues simulés comportant des altérations.

Pour chaque modalité il est possible que l'utilisateur réalise plusieurs fois une altération à son comportement conventionnel. En effet, un saut en avant peut avoir lieu n'importe où au cours de l'interaction à partir du moment où l'utilisateur est capable de prendre deux initiatives successives et la modification du besoin d'information et le rejet implicite des résultats ont lieu au sein de la boucle lancement/évaluation/réparation qui se répète jusqu'à ce que l'utilisateur soit satisfait. La figure 6.5 donne le nombre moyen d'altérations ayant eu lieu au sein des dialogues étudiés pour chaque modalité. Ces moyennes nous sont

utiles pour mesurer l'occurrence d'événements par altération, plutôt que par dialogue.

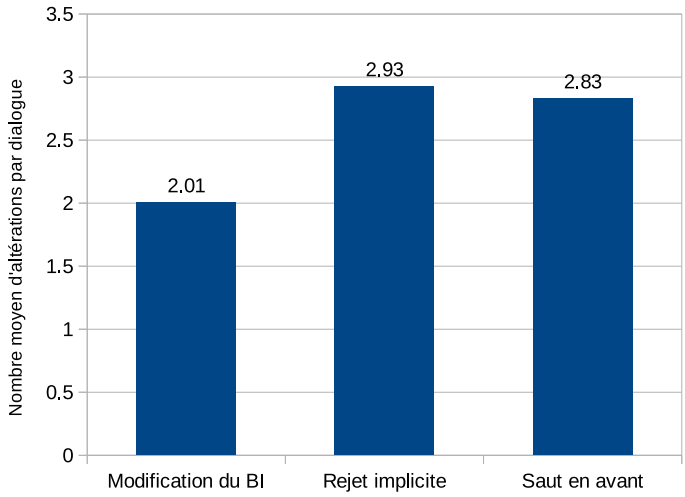

FIGURE 6.5 – Nombre moyen d'altérations du comportement conventionnel réalisées par l'utilisateur dans les dialogues contenant des comportements altérés pour chaque dialogue simulé.

Les différentes modalités étudiées ont un impact sur la longueur des dialogue simulés. La taille moyenne d'un dialogue est donnée dans la figure 6.6 en nombre de tours de parole. Afin de mettre ces informations en perspective par rapport à un déroulement purement conventionnel de la tâche, la longueur des dialogues selon la modalité d'initiative mixte est rappelée.

La légère augmentation de taille des dialogues contenant des modifications du besoin d'information par l'utilisateur s'explique par l'ajout de motifs dialogiques initiés par l'utilisateur lorsqu'il décide de modifier son besoin d'information en cours de recherche. L'allongement plus important que l'on constate pour le rejet implicite des résultats s'explique par le fait que l'utilisateur, en « court-circuitant » l'état d'évaluation des résultats, a moins souvent l'opportunité d'exprimer sa satisfaction vis-à-vis des résultats renvoyés par la requête et fait plus de cycles lancement/évaluation/réparation. Le léger raccourcissement des dialogues comprenant des sauts en avant s'explique par le fait que l'utilisateur ne verbalisant pas toutes ses actions, il lui faut moins de tours de parole pour réaliser sa tâche de recherche d'information.

Avant de nous plonger dans l'étude du déroulement des interactions simulées, nous présentons comme résultat préliminaire la proportion de dialogues arrêtés suite à une inconsistance logique au cours des simulations, en figure 6.7.

Nous remarquons ici que le système n'a jamais rencontré d'inconsistance logique dans les expériences où l'utilisateur peut modifier son besoin d'information ou faire un rejet implicite des résultats de la requête. Au cours des expériences où l'utilisateur peut réaliser un saut en avant, le système s'est arrêté suite à une inconsistance logique dans 1.51% des cas. Cela correspond au cas de figure où l'utilisateur fait un saut en avant lors d'un cycle lancement/évaluation/réparation, et ne verbalise pas de comportement de réparation.

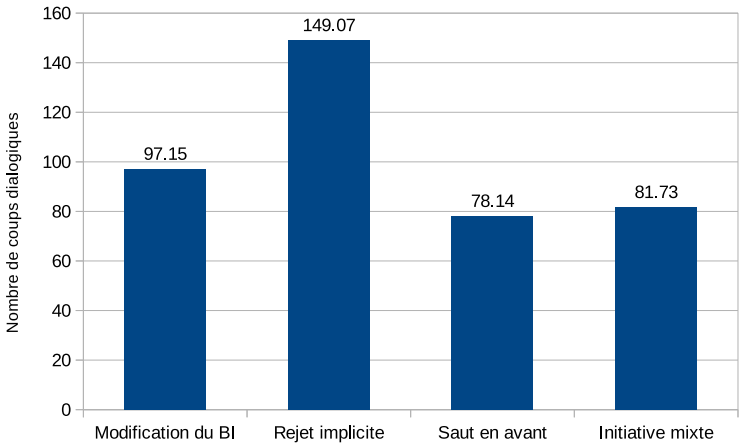

FIGURE 6.6 – Taille moyenne d'un dialogue simulé.

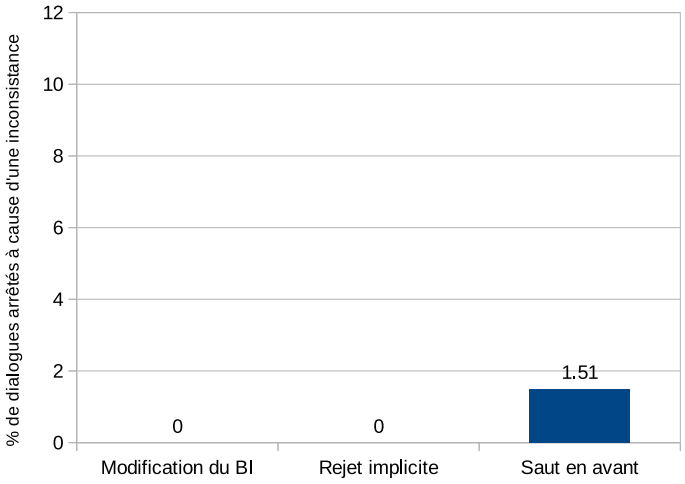

FIGURE 6.7 – Proportion de dialogues arrêtés suite à une inconsistance logique pour chaque dialogue simulé. Métrique utilisée : $L_{\perp}$.

Cela amène alors un enchaînement lancement/évaluation/lancement/évaluation et si la première évaluation est négative et la seconde positive, cela introduit la contradiction qui bloque le système. Ce type d'inconsistance logique ayant lieu suite à un saut en avant de l'utilisateur est illustré dans l'extrait de dialogue donné en annexe B.2.1 dans le tableau B.8.

Contextualisation et transitions effectuées

Nous nous intéressons maintenant à la capacité du système à contextualiser les comportements de l'utilisateur et à réaliser des transitions entre les états selon les différentes modalités possibles.

Échecs de contextualisation Les échecs de contextualisation correspondent au fait que le système n'est pas capable de trouver dans ses états un comportement possible où contextualiser une initiative prise par l'utilisateur, le menant à ignorer cette initiative. La figure 6.8 présente la proportion $C_{\varnothing d}$ de dialogues contenant des échecs de contextualisation pour chaque modalité.

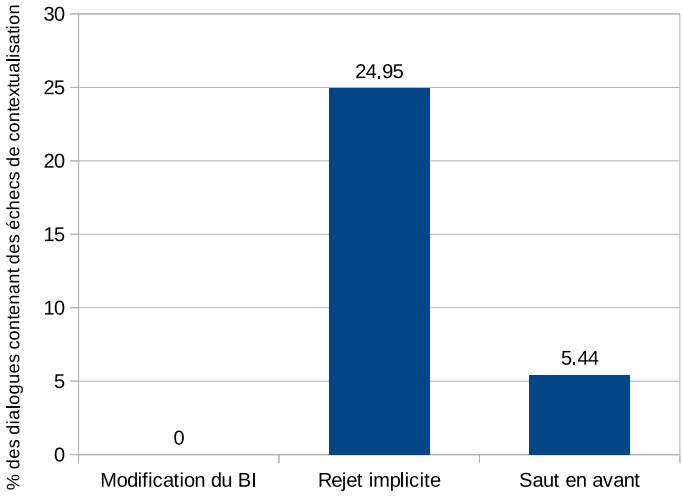

FIGURE 6.8 – Proportion de dialogues contenant des échecs de contextualisation pour chaque modalité. Métrique utilisée : $C_{\varnothing d}$.

Nous remarquons qu'au cours de l'expérience permettant à l'utilisateur de modifier son besoin d'information, aucune initiative de la part de celui-ci n'a mené à un échec de contextualisation. En effet dans cette modalité, les altérations que l'utilisateur peut apporter à son comportement conventionnel sont de rajouter des expressions à la verbalisation de son besoin d'information. Ces expressions peuvent être contextualisées dans les états de formulation ou de précision de la verbalisation. Or, ces états ne comportant pas de comportements réalisables, il n'ont pas de valeur de maturité et sont toujours accessibles, même si leurs prérequis ne sont pas satisfaits. Les comportements de l'utilisateur corres-

pondant à des altérations seront donc toujours contextualisables et ne mèneront jamais à un échec de contextualisation.

Les deux autres modalités d'expérimentation ont mené à des échecs de contextualisation, dans 24.95% des dialogues pour le rejet implicite des résultats de la requête et dans 5.44% des dialogues pour les sauts en avant. Si ces proportions semblent importantes, elles sont à relativiser par rapport à la totalité des initiatives prises par l'utilisateur au cours des dialogues. Cette perspective est donnée par la figure 6.9, qui donne la métrique $C_{\varnothing}$ d'initiatives de l'utilisateur dont la contextualisation a échoué, sur l'ensemble des dialogues étudiés, ainsi que la métrique $C_{\varnothing/\varnothing}$, qui donne la proportion d'initiatives de l'utilisateur dont la contextualisation a échoué, sur l'ensemble des initiatives prises dans des dialogues contenant des échecs de contextualisation. Nous observons ici le fait que ces proportions sont beaucoup moins importantes, et que dans le cas spécifique des dialogues contenant des échecs de contextualisation, seulement 6.73% et 4.04% des initiatives sont mal contextualisées pour respectivement les modalités de rejet implicite et de saut en avant.

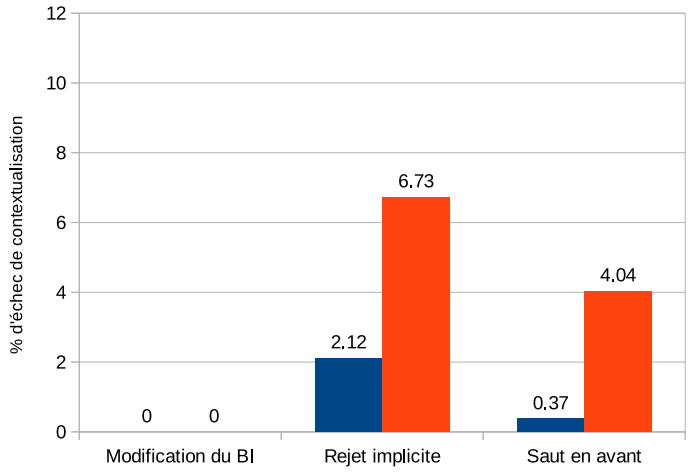

FIGURE 6.9 – Proportion d'échec de contextualisation pour chaque modalité. Les barres de gauche correspondent à la métrique $C_{\varnothing}$ et représentent la proportion d'échec de contextualisation par rapport à l'ensemble des initiatives prises par l'utilisateur dans tous les dialogues. Les barres de droite correspondent à la métrique $C_{\varnothing/\varnothing}$ et représentent la proportion d'échec de contextualisation par rapport à l'ensemble des initiatives prises par l'utilisateur dans les dialogues contenant des erreurs de contextualisation.

Afin d'illustrer ce qui peut mener à un échec de contextualisation dans le cas d'un rejet implicite, un dialogue illustrant une situation possible est disponible en annexe B.2.2, dans les tables B.9 etB.10. La grande variété d'altérations possibles menant à un échec de contextualisation dans le cas où l'utilisateur peut réaliser un saut en avant ne nous permet pas de détailler au cas par cas ce qui rend cette situation possible. Nous constatons cependant que le nombre d'initiatives de l'utilisateur ayant mené à un échec de contextualisation est très restreint et que le système est capable de rattacher une action de l'utilisateur à

l'état actuel de la tâche et de l'interaction dans une grande majorité des cas.

Nous nous penchons maintenant sur les transitions entre états qui ont eu lieu lors des simulations et sur la manière dont elles se sont déroulées.

Transitions vers des états matures Lors d'une transition vers un état candidat, le système a été capable de calculer des prédicats lui indiquant que le comportement initié par l'utilisateur est conventionnellement attendu. Une transition vers un état mature est choisie par le système lorsqu'il ne peut pas contextualiser une initiative de l'utilisateur dans l'état courant ni dans un état candidat, mais que cette contextualisation est possible dans un état dont la maturité est non-nulle. Ce type de transition correspond à un cas de figure où le comportement de l'utilisateur n'est pas conventionnellement attendu par le système. Cependant, celui-ci cherche quand même à rattacher l'initiative de l'utilisateur à un contexte possible de la tâche, notamment dans le but de maintenir une continuité de l'interaction avec l'utilisateur et être capable de lui répondre de manière cohérente plutôt que d'ignorer son initiative comme c'est le cas lors d'un échec de contextualisation.

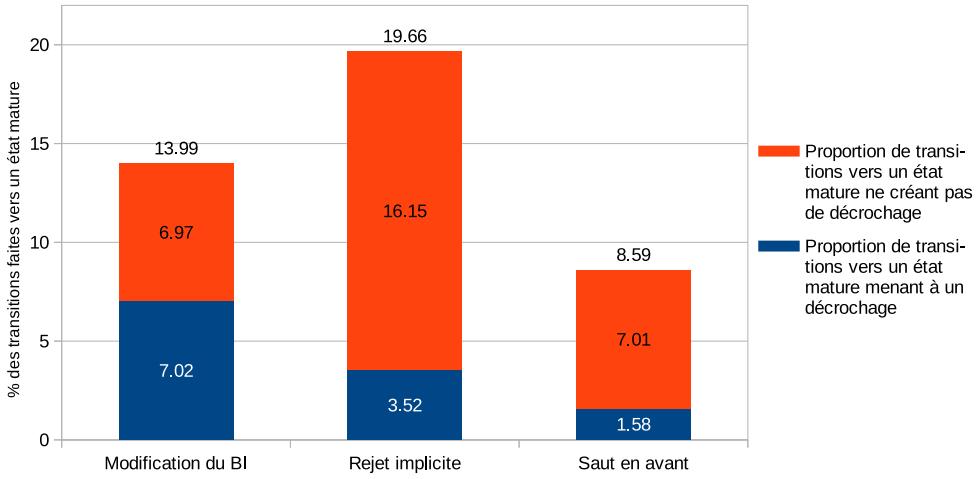

FIGURE 6.10 – Proportion des transitions réalisées vers un état mature pour chaque dialogue simulé. La partie basse des colonnes (en bleu) correspond à la métrique T_{md}. La partie haute des colonnes (en orange) correspond à la métrique T_{ms}. Le total présent au-dessus des colonnes correspond à la métrique T_m.

La figure 6.10 présente pour chaque modalité les proportions T_m de transitions réalisées vers des états matures. Une première remarque sur cette figure est le fait que, au cours des trois expériences réalisées, la proportion des transitions réalisées vers des états matures est faible, la modalité de rejet implicite étant celle en comptant le plus avec 19,66% du nombre total de transitions.

Cette figure sous-divise les transitions faites vers un état mature en deux catégories : celles ayant mené à un décrochage (T_{md}, en bleu, en bas) et celles ayant permis une contextualisation avec succès (T_{ms}, en orange, en haut). Nous observons pour les modalités

de rejet implicite et de saut en avant qu'une grande majorité (respectivement 82.15% et 81.61%) des transitions vers des états matures ont mené à une contextualisation avec succès du comportement de l'utilisateur.

En ce qui concerne la modification du besoin d'information, on remarque qu'une transition sur deux est réalisée avec succès. Cela s'explique par le fait que ces transitions sont déclenchées par une altération du comportement conventionnel de l'utilisateur, qui ajoute des expressions à la verbalisation de son besoin d'information au cours du cycle lancement/évaluation/réparation. Or, ces expressions peuvent être contextualisées soit dans l'état de précision de la verbalisation (contextualisation correcte), soit dans l'état de formulation d'une verbalisation (contextualisation incorrecte). Comme nous l'avons vu précédemment, ces deux états n'ayant pas de valeur de maturité, ils sont accessibles en permanence. Lorsqu'il doit contextualiser un comportement altéré de l'utilisateur, le système choisit au hasard parmi ces deux états et contextualise donc avec succès le comportement de l'utilisateur dans la moitié des cas.

Ces résultats montrent que le mécanisme des transitions par état mature est une force pour le système, car il lui donne la possibilité de continuer de suivre l'utilisateur malgré un contexte insuffisant pour calculer les prérequis permettant de réaliser une transition vers un état candidat. Cela rend le système plus flexible et plus robuste aux altérations du comportement conventionnel de l'utilisateur en lui évitant de décrocher dans une majorité des cas.

Décrochages

Comme illustré dans l'algorithme 4, un décrochage a lieu lorsque l'état dans lequel se trouve le système est différent de celui dans lequel se trouve l'utilisateur. Les décrochages sont la conséquence d'un échec ou d'une erreur de contextualisation d'un comportement de l'utilisateur par le système. Le diagramme 6.11 présente la proportion $D_{\#}$ de dialogues présentant des décrochages parmi les dialogues étudiés. Nous rappelons ici que les dialogues considérés pour cette étude contiennent tous des altérations du comportement conventionnel de l'utilisateur.

Malgré les altérations au comportement conventionnel de l'utilisateur, au moins la moitié des dialogues se sont déroulés sans décrochage, ce chiffre étant presque aux deux tiers dans le cas où l'utilisateur peut faire des sauts en avant lorsqu'il réalise sa tâche.

Le diagramme 6.12 montre le rapport D_r entre le nombre de déviations du comportement conventionnel de l'utilisateur et le nombre de comportements altérés du système.

Une altération du comportement de l'utilisateur mène à un décrochage dans moins d'un cas sur cinq pour la modalité de saut en avant et dans moins d'un cas sur quatre dans la modalité d'un rejet implicite. Nous retrouvons ici, dans le cas de la modification de besoin d'information de l'utilisateur, le fait que le système se trompe dans un cas sur deux lorsqu'il contextualise une initiative de l'utilisateur issue d'une altération de son comportement conventionnel. Nous avons donc la moitié de ces altérations qui mène à un décrochage de la part du système.

Ces résultats sont à mettre en rapport avec la longueur des décrochages selon chaque modalité. Le diagramme 6.13 présente la longueur moyenne D_l des décrochages ayant eu

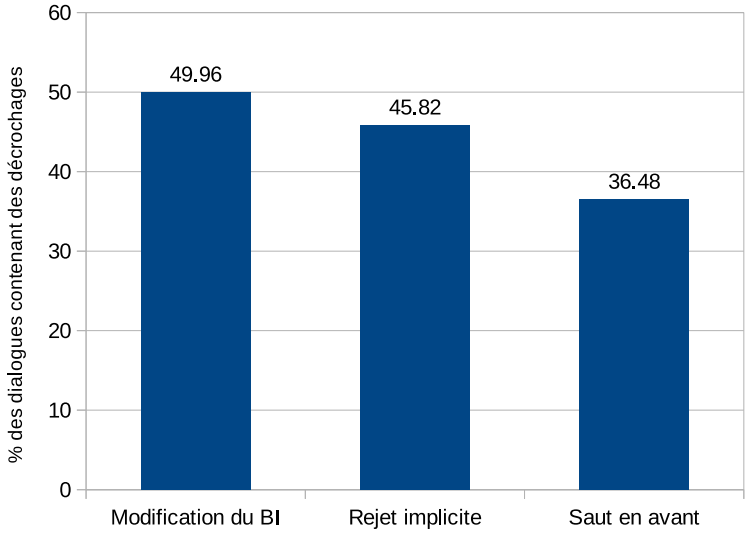

FIGURE 6.11 – Proportion de dialogues contenant des décrochages sur l'ensemble des dialogues présentant des altérations du comportement conventionnel de l'utilisateur pour chaque dialogue simulé. Métrique utilisée : $D_{\#}$.

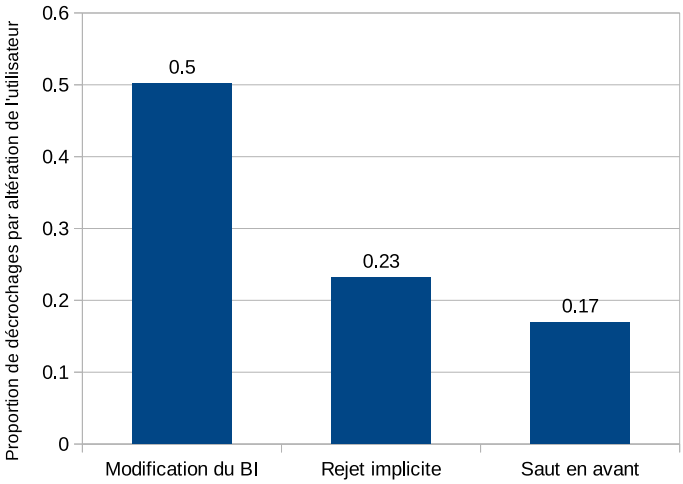

FIGURE 6.12 – Proportion du nombre de décrochages réalisés par le système rapporté au nombre de comportements altérés de l'utilisateur pour chaque dialogue simulé. Métrique utilisée : D_r.

lieu au cours des simulations en nombre de coups dialogiques.

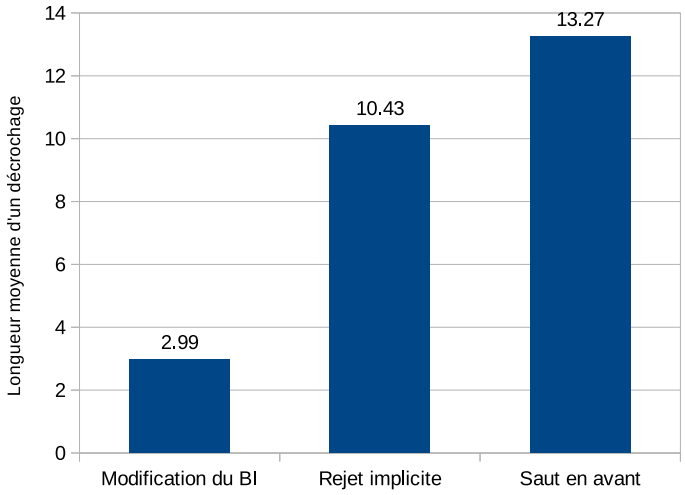

FIGURE 6.13 – Longueur moyenne des décrochages au sein de dialogues contenant des décrochages pour chaque dialogue simulé. Métrique utilisé : D_l.

Dans le cas de la modification du besoin d'information de l'utilisateur, la longueur moyenne d'un décrochage correspond à la durée passée par l'utilisateur à mettre à jour son besoin d'information. Cependant, lorsque cette mise à jour est terminée et que l'utilisateur revient à un comportement conventionnel, le système est tout de suite capable de le re-contextualiser correctement. Les décrochages ayant eu lieu lors de ces simulations ont, malgré leur fréquence relativement élevée, un impact négligeable sur le bon déroulement des dialogues.

Dans le cas du rejet implicite des résultats de la requête, la longueur moyenne d'un décrochage ne correspond pas forcément à la longueur d'un cycle de réparation de la requête. En effet, le système peut rester en décrochage pendant plusieurs cycles de lancement/évaluation/réparation ou, au contraire, de re-contextualiser l'utilisateur dans le bon état rapidement. Cela est notamment possible dans le cas où l'utilisateur demande la suppression d'un mot-clé de la requête, qui n'est attendu que dans l'état de réparation de la requête et qui permet au système de faire une transition vers cet état dans le cas où il se trouverait dans l'état d'alignement verbalisation/terminologie. L'impact d'un décrochage dans cette situation peut être faible s'il ne dure que quelques coups, ou plutôt important s'il s'étend sur plusieurs lancement de requêtes (dans ce cas, il inclut des échecs de contextualisation, le système n'ayant pas été capable de contextualiser le lancement de la requête par l'utilisateur, ce qui terminerait le décrochage).

Dans le cas du saut en avant de la part de l'utilisateur, on constate que la durée des décrochages est particulièrement importante. Cela est causé par le fait que l'utilisateur peut omettre des comportements critiques au cours de la réalisation de la tâche (comme préciser qu'il a terminé de formuler sa verbalisation). Ce type d'omission a un fort impact sur la manière dont le système calcule ses prédicats et peut le mener à prendre des initiatives

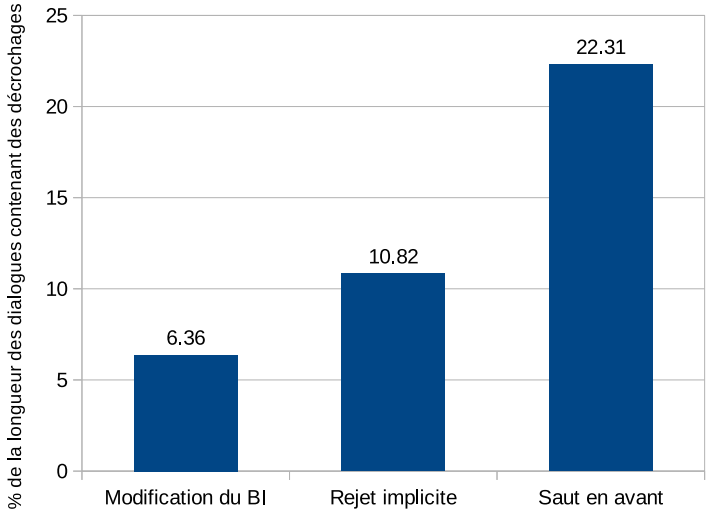

FIGURE 6.14 – Proportion de la durée passée en décrochage dans les dialogues contenant des décrochages pour chaque dialogue simulé. Métrique utilisée : D_p.

incohérentes avec l'état dans lequel se trouve l'utilisateur. Or, celui-ci est collaboratif, il va accepter de répondre au système même si de son point de vue le comportement de celui-ci n'est pas cohérent. Il faut alors attendre que l'utilisateur puisse à nouveau prendre l'initiative d'ouvrir un motif dialogique pour que le système ait la possibilité de le re-contextualiser correctement.

Le diagramme 6.14 présente la proportion D_p de temps passé en décrochage pour les dialogues contenant des décrochages.

Un décrochage lors d'un dialogue où l'utilisateur peut faire un saut en avant a une durée particulièrement importante, les dialogues de cette modalité étant plutôt courts (cf. diagramme 6.6). La longueur importante d'un décrochage dans le cas où l'utilisateur peut faire un saut en avant est cependant compensée par le fait que, comme nous l'avons vu dans les diagrammes 6.11 et 6.12, ce cas de figure est celui qui a le moins de chance de se produire.

Ces résultats confortent le fait que, dans la plupart des cas, le système est capable de maintenir une cohérence à la fois interactive et contextuelle des actions réalisées par l'utilisateur.

Synthèse

Les différentes expériences présentées dans ce chapitre permettent d'évaluer la capacité du système à mener à bien une interaction avec un utilisateur afin de l'aider à réaliser une recherche d'information.

Nous avons dans un premier temps montré la capacité du système à suivre un utilisateur ayant un comportement purement conventionnel. Nous avons aussi montré comment notre système est capable d'interpréter la violation d'un engagement en action par l'utilisateur

et de continuer à interagir avec celui-ci malgré cette altération.

Dans un second temps, nous avons étudié le comportement du système lors des simulations où les altérations apportées par l'utilisateur à son comportement conventionnel pouvaient amener à des inconsistances logiques, des échecs de contextualisation ou des décrochages. Les analyses réalisées montrent que les différents mécanismes mis en place par le système pour contextualiser les initiatives de l'utilisateur permettent à celui-ci de suivre l'utilisateur dans une majorité des cas malgré les altérations et que dans le cas où le système échoue ou se trompe lors de la contextualisation d'un comportement de l'utilisateur, il est capable de résilience et le décrochage qui s'en suit ne mène pas à un échec de l'interaction. Le système est capable dans la plupart des cas d'assurer une continuité interactive avec un faible nombre d'échecs de contextualisation. Dans le cas d'un décrochage, menant à une rupture de la continuité contextuelle de l'interaction, le système est capable de se servir des actions réalisées par la suite par l'utilisateur pour rectifier ce décrochage et revenir à un contexte similaire à celui de l'utilisateur.

Conclusion

L'utilisation de l'intelligence artificielle dans un nombre croissant de domaines fait émerger le besoin de reconsidérer la place occupée par la machine lors de la résolution d'un problème par un humain. En effet, celle-ci tend désormais à être considérée par les humais comme un partenaire avec lequel collaborer plutôt qu'un outil à exploiter. Ce changement de perspective fait de l'interaction humain-machine un sujet de recherche clé pour le développement de systèmes répondant de manière pertinente et constructive aux attentes de leurs utilisateurs.

Dans cette thèse, nous avons proposé un modèle pour la réalisation d'une tâche collaborative complexe entre un humain et une machine, dans un contexte ne permettant pas de planification. Nous y avons défendu l'utilisation de motifs dialogiques, une représentation conventionnelle de la structure des échanges humain-humain, pour fonder un modèle permettant les échanges avec un utilisateur en respectant ses habitudes d'interaction.

Cette représentation conventionnelle de l'interaction nous a servi de base pour le développement de notre modèle de tâche collaborative humain-machine. Nous avons introduit des structures, appelées *états*, qui enrichissent les motifs dialogiques et les organisent afin de retranscrire la contribution des interactions des interlocuteurs à la réalisation de la tâche.

Notre modèle permet la représentation du déroulement d'une tâche collaborative humain-machine en décrivant les processus intentionnels mis en œuvre pour faire avancer celle-ci. Du côté de la machine, ces processus décisionnels tirent parti de la représentation de la tâche sous forme d'états. Ils se basent sur la notion de *maturité* d'un état, qui décrit la capacité du système à prendre des initiatives dans cet état, pour interpréter les actions de l'utilisateur et choisir de manière pertinente la prochaine action à réaliser. En nous basant sur un corpus de recherche d'information collaborative humain-humain, nous avons présenté une mise en œuvre de ce modèle et l'avons testé à l'aide d'un utilisateur simulé. Cette évaluation a montré que le système est capable de suivre l'utilisateur au cours de la réalisation de la tâche et fait preuve de résilience en cas de décrochage.

Nous avons apporté des éléments de réponse aux quatre problématiques données en introduction :

Problématique 1 : comment s'inspirer des interactions humain-humain pour modéliser l'interaction humain-machine ?

Réponse 1 : nous avons discuté dans le chapitre 2 des différents modèles formels existant pour représenter les interactions dialogiques. Nous avons retenu le fait que le

modèle des *motifs dialogiques* permettent de représenter les conventions régissant les interactions dialogiques humaines tout en laissant une flexibilité suffisante pour être utilisés au sein de processus intentionnels. Nous nous sommes accordés sur le fait que le modèle de dialogue de DOGMA est celui répondant au mieux à notre besoin de formalisation des interactions dialogiques pour une interaction humain-machine.

Problématique 2 : comment représenter la structure d'une interaction humain-machine véhiculant la réalisation collaborative d'une tâche ?

Réponse 2 : nous avons introduit un modèle organisant les motifs dialogiques en structures cohérentes appelées *état*. Chaque état représente une sous-tâche dont la réalisation contribue à l'avancée de la tâche sous-jacente au dialogue. Les états regroupent des *comportements possibles*, qui décrivent les motifs dialogiques conventionnellement attendus de la part de chaque interlocuteur, ainsi que leurs effets dans le contexte particulier de cet état.

Problématique 3 : comment la machine peut-elle tirer parti de la représentation de la tâche et de l'interaction qui la supporte pour interpréter les actions de l'utilisateur et décider de ses propres actions ?

Réponse 3 : les états contiennent des *prérequis*, qui sont des formules décrites à l'aide de prédicats. Ces prédicats sont calculés par le système en fonction de l'état actuel de l'interaction et de la tâche. Le fait que les prérequis d'un état soient satisfaits donne la possibilité à notre système de réaliser une *transition d'état* pour contextualiser une action de l'utilisateur dans cet état.

Notre modèle d'état intègre de plus la notion de *maturité* pour les comportements réalisables par le système et pour les états de la tâche. La maturité d'un état nous permet de déterminer le *potentiel d'interaction* du système dans cet état donné.

Différents mécanismes sont mis en place pour tirer parti des prérequis des états et de leur maturité afin de contextualiser un acte de dialogue de l'utilisateur dans un comportement présent dans un état, en réalisant une *accommodation* de l'état d'information si nécessaire. La description conventionnelle des états de la tâche permet au système de savoir, à n'importe quel moment de l'interaction, quelles sont les actions qui sont conventionnellement attendues de sa part et quelles sont les actions qu'il est conventionnellement acceptable qu'il entreprenne. L'utilisation du concept de *maturité* d'un comportement réalisable par le système lui permet de savoir, parmi les comportements qu'il peut réaliser, lequel est le plus à même d'être pertinent au niveau de l'interaction et constructif au niveau de la tâche.

Problématique 4 : étant donné un modèle d'interaction humain-machine pour la réalisation collaborative d'une tâche, comment est-il possible d'évaluer la capacité de ce modèle à suivre correctement le déroulement d'une tâche ?

Réponse 4 : nous avons donné dans le chapitre 5 une description complète de la tâche de recherche d'information collaborative humain-machine sous la forme d'états. Nous avons montré, dans le chapitre 6, comment le système se comporte face à un utilisateur simulé lors du déroulement d'une interaction sur cette tâche. Nous avons vu que lorsque l'utilisateur applique un comportement respectant strictement la

structure de la tâche, le système contextualise toujours de manière pertinente ses actions, et cela dans différents contextes d'initiative.

Nous avons ensuite introduit différentes altérations au comportement conventionnel de l'utilisateur et observé que, dans le cas où le système échoue ou se trompe en contextualisant une action de l'utilisateur, il est capable de se rattraper et de revenir à une interprétation correcte de ses actions dans la suite de l'interaction.

Perspectives

Comme nous l'avons montré dans les chapitres 5 et 6 l'utilisation de ces travaux pour représenter une interaction humain-machine sur une tâche de recherche d'information collaborative est possible et donne des résultats encourageants. Cependant, des extensions à ces travaux sont envisageables et permettraient d'enrichir le modèle proposé.

Apprendre la définition des prédicats Nous avons utilisé une représentation formelle pour définir les prédicats utilisés dans nos tables d'état. Cela traduit le besoin, pour la définition des prédicats, de la représentation d'une connaissance experte sur le domaine. Cependant, nous pensons que cette approche n'est pas toujours adaptée, notamment parce que ces définitions sont faites « en dur », ne fournissant aucune souplesse à l'activation ou non d'un prédicat. Cette approche risque d'être peu adaptée dans un cas réel d'interaction avec un être humain, qui ne raisonne pas de manière binaire ou rationnelle.

Une alternative possible pour le calcul de ces prédicats est l'utilisation de techniques d'apprentissage pour permettre au système de prendre des décisions partiellement basées sur des statistiques issues d'interactions passées. L'utilisation de techniques d'apprentissage pour le calcul des prédicats semble d'autant plus intéressante qu'elle illustrerait la capacité de notre système à articuler d'un côté une approche formelle de l'interaction, structurée autour de règles définissant des conventions d'interaction à différents niveaux (jeux de dialogue, comportements possibles et états), et d'un autre côté une prise de décision tirant parti d'un apprentissage réalisé sur des interactions passées pour choisir la meilleure option. On aurait alors une organisation unifiant une structure conventionnelle symbolique et une structure intentionnelle statistique. Ce qui nous a empêché de travailler dans ce sens et de mettre en œuvre cette approche est l'absence, à l'heure actuelle, d'un jeu de données suffisamment grand pour réaliser cet apprentissage.

Introduire de la *multimodalité* Notre travail s'est focalisé sur l'utilisation de motifs dialogiques pour modéliser l'interaction. Cependant, si l'on souhaite doter les machines de capacités d'interactions plus proches de celles des humains, il est nécessaire d'étendre les supports de communication utilisés et de tenir compte de l'aspect multimodal des interactions humaines. La gestuelle de l'utilisateur ou ses expressions faciales sont des exemples d'indicateurs qui peuvent être exploités pour enrichir les échanges avec la machine. Cela nécessite d'employer des représentations tenant compte des différents moyens de communication utilisés par les humains. Notre modèle pourrait alors être étendu avec l'utilisation de *motifs interactifs* plutôt que de motifs dialogiques, qui seraient

une représentation des interactions conventionnelles humaines tenant compte des différents canaux de communication utilisés par les humains.

Implémenter et tester le modèle en conditions réelles La finalité des travaux présentés dans ce manuscrit est la conception d'un système interactif pour assister un humain sur la réalisation de tâches complexes. Cet objectif n'est pas encore atteint puisque la seule mise en œuvre de notre modèle a été faite sans permettre le recours à des ressources du domaine et sans permettre l'interaction avec un être humain. Deux pistes sont à explorer. La première consiste à poursuivre le développement du modèle de recherche d'information collaborative afin de permettre son utilisation avec un outil du domaine, comme le moteur de recherche de la plate-forme CISMeF [2], et de le rendre accessible à des utilisateurs humains. Cela permettrait de mettre en œuvre une évaluation expérimentale de ce modèle avec des données tirées d'interactions humain-machine.

La seconde piste à explorer est l'utilisation de ce modèle pour d'autres domaines d'application que la recherche d'information collaborative. Notre modèle a vocation à s'adapter à différents types de tâche complexes qui nécessitent d'être résolues collaborativement. L'introduction de multimodalité pour la représentation de l'interaction conventionnelle peut ouvrir de nouveaux champs d'application, comme des assistants personnels ou des intelligences artificielles domestiques.

2. Disponible à l'adresse `http://www.chu-rouen.fr/cismef/`, consulté le 7/05/2019.

Liste des figures

Liste des tableaux

Liste des tables d'état

Liste des définitions

Annexe A

Définition des prédicats

Cette annexe donne une définition, formelle si possible, des prédicats utilisés dans le chapitre 5. La valeur des prédicats varie en fonction du contexte (principalement de l'état du tableau de conversation). Dans certaines définitions, nous considérons que nous avons accès à des ressources extérieures enrichissant les connaissances et capacités de raisonnement du système, notamment sur des aspects de traitement automatique des langues et relatifs au domaine de la recherche d'information.

A.1 Ouverture, verbalisation et construction de la requête

A.1.1 Ouverture

Le prédicat InteractionNotStarted servant de prérequis à l'état signifie que l'interaction n'a pas encore commencé (on se trouve au tout début de l'interaction). Il est vrai si aucune interaction n'a encore eu lieu. Il vérifie donc simplement que le tableau de conversation est vide.

Prédicat 1 – InteractionNotStarted **:**

$\mathrm{T}_i = \varnothing$

La précondition du comportement réalisable est la négation du prédicat SystemFunctionShared. Ce prédicat est vrai si le système a partagé avec l'utilisateur le fait que sa fonction est de l'aider sur une tâche de RI. Il vérifie donc que l'engagement du système sur sa fonction a été créé.

Prédicat 2 – SystemFunctionShared **:**

$\mathrm{T}_i \ni \mathrm{C}(y, \text{function}(\textit{helpForIR}), \mathbf{Crt})$

A.1.2 Formulation d'une verbalisation

Le prédicat VerbalisationMissing présent en précondition indique que l'utilisateur n'a pas encore formulé de verbalisation. Il vérifie donc que le tableau de conversation ne contient aucun engagement de l'utilisateur sur une expression de verbalisation.

Prédicat 3 – VerbalisationMissing **:**

$\nexists e$, $\mathrm{T}_i \ni \mathrm{C}(x, \text{verbExpression}(e), \mathbf{Crt})$

A.1.3 Reformulation

Le prédicat VerbalisationComplete est vrai si l'utilisateur a affirmé que sa verbalisation est complète. Il vérifie donc que l'utilisateur s'est engagé sur cette proposition.

Prédicat 4 – VerbalisationComplete **:**

$\mathrm{T}_i \ni \mathrm{C}(x, \text{verbComplete}, \mathbf{Crt})$

Le prédicat VerbExpression(*expr*) est vrai si l'expression *expr* fait partie de la verbalisation du besoin d'information de l'utilisateur.

Prédicat 5 – VerbExpression(*expr*) **:**

$\mathrm{T}_i \ni \mathrm{C}(x, \text{verbExpression}(expr), \mathbf{Crt})$

Le prédicat VerbalisationSufficient est vrai si la verbalisation de l'utilisateur est suffisamment riche. Nous considérons qu'elle est suffisamment riche si elle comporte trois expressions différentes.

Prédicat 6 – VerbalisationSufficient **:**

$$\wedge \left| \begin{array}{l} \mathrm{T}_i \ni \mathrm{C}(x, \text{verbExpression}(a), \mathbf{Crt}) \\ \mathrm{T}_i \ni \mathrm{C}(x, \text{verbExpression}(b), \mathbf{Crt}) \\ \mathrm{T}_i \ni \mathrm{C}(x, \text{verbExpression}(c), \mathbf{Crt}) \\ \mathrm{a} \neq \mathrm{b} \neq \mathrm{c} \neq \mathrm{a} \end{array} \right.$$

Le prédicat Reformulation(*verb*, *ref*) est vrai si les ressources linguistiques du système donnent *ref* comme reformulation de *verb*.

Prédicat 7 – Reformulation(*verb*, *ref*) **:**

Utilisation d'une ressource de traitement automatique des langues et/ou du domaine pour calculer ce prédicat.

Discussion sur ces prédicats

Le calcul correct des prédicats VerbalisationSufficient et Reformulation(*expr*, *ref*) nécessite de faire appel à des ressources du domaine ou linguistiques dont nous ne disposons

pas dans le cadre de nos travaux. Nous laissons donc de côté le calcul exact de ces prédicats. Même si nous ne traitons pas cet aspect, nous sommes conscient de l'importance qu'il revêt dans le déroulement de la tâche sous-jacente à l'interaction. En effet, la construction d'une représentation structurée du besoin d'information de l'utilisateur peut apporter un bénéfice important à la suite de l'interaction, notamment au moment de l'alignement verbalisation/terminologie. L'interprétation d'une formulation en langue naturelle vers un langage d'interrogation de base de données a fait l'objet d'un développement approfondi dans les travaux de thèse de Pradel [Pra13]. Ces travaux reposent notamment sur le concept de *requête pivot*, qui interprète une requête en langue naturelle en une représentation intermédiaire entre la langue naturelle et le langage de requête ciblé.

A.1.4 Demande de précisions

Le prédicat PrecisionNotAsked est vrai si le système n'a pas encore demandé à l'utilisateur de préciser sa verbalisation.

Prédicat 8 – PrecisionNotAsked **:**

$T_i \ni C(x, \text{preciseVerb}, \textbf{Ina})$

Le prédicat QueryEmpty est vrai si la requête courante est vide.

Prédicat 9 – QueryEmpty **:**

$\nexists i, T_i \ni C(\_, \text{queryKeyWord}(kw), \textbf{Crt})$

A.1.5 Précision de la verbalisation

Le prédicat UserWillPreciseVerbalisation est vrai si l'utilisateur s'est engagé à préciser sa verbalisation. Il vérifie donc que celui-ci a contracté un engagement en action sur la précision de sa verbalisation.

Prédicat 10 – UserWillPreciseVerbalisation **:**

$T_i \ni C(x, \text{preciseVerb}, \textbf{Crt})$

A.1.6 Alignement verbalisation/terminologie

Le prédicat QueryNeverLaunched est vrai si aucun lancement de requête n'a encore été réalisé.

Prédicat 11 – QueryNeverLaunched **:**

$\nexists i, T_i \ni C(\_, \text{currentQueryLaunched}, \textbf{Crt})$

Le prédicat VerbalisationPrecisionFailed est vrai si la précision de la verbalisation a échoué, c'est-à-dire que l'utilisateur a refusé de préciser sa verbalisation ou s'y est engagé

mais ne l'a pas fait.

Prédicat 12 – VerbalisationPrecisionFailed **:**

$$\vee \left| \begin{array}{l} \mathrm{T}_i \ni \mathrm{C}(x, \text{preciseVerb}, \textbf{Fal}) \\ \mathrm{T}_i \ni \mathrm{C}(x, \text{preciseVerb}, \textbf{Vio}) \end{array} \right.$$

Le prédicat QueryKeyWord(kw) est vrai si le mot-clé kw est un mot-clé de la requête q.

Prédicat 13 – QueryKeyWord(kw) **:**

$\mathrm{T}_i \ni \mathrm{C}(\_, \text{queryKeyWord}(kw), \textbf{Crt})$

Le prédicat RelevantForAddition(kw) est vrai si l'ajout du mot-clé kw à la requête est constructif pour préciser la requête.

Prédicat 14 – RelevantForAddition(kw) **:**

Il fait appel aux outils du domaine pour déterminer le ou les mots-clés les plus pertinents à ajouter à la requête.

A.2 Lancement de la requête

Le prédicat QueryLaunchable est vrai si la requête courante respecte les critères minimaux pour être lancée.

Prédicat 15 – QueryLaunchable **:**

Dépend de l'outil de recherche d'information utilisé. Typiquement, la requête courante devra être non vide et renvoyer au moins un résultat.

Le prédicat QueryLaunchRelevant est vrai si le système juge qu'il est pertinent de proposer à l'utilisateur de lancer la requête courante.

Prédicat 16 – QueryLaunchRelevant **:**

Dépend des outils de domaine et de recherche d'information utilisés ainsi de combien le système juge les résultats de cette requête comme répondant au besoin d'information de l'utilisateur.

A.3 Évaluation des résultats

A.3.1 Évaluation des résultats par l'utilisateur

Le prédicat CurrentQueryLaunched est vrai si la requête courante a été lancée. Il vérifie que le système est engagé sur le lancement de la requête courante.

Prédicat 17 – CurrentQueryLaunched **:**

$T_i \ni C(y, \text{currentQueryLaunched}, \textbf{Crt})$

Le prédicat CurrentQueryResultsSatisfying est vrai si les résultats de la requête q ont été jugés satisfaisants par l'utilisateur. Il vérifie que l'engagement par l'utilisateur sur le fait que q est satisfaisante est actif.

Prédicat 18 – CurrentQueryResultsSatisfying **:**

$T_i \ni C(x, \text{currentQueryResultsSatisfying}, \textbf{Crt})$

Le prédicat ResearchNotOver est vrai lorsque l'utilisateur ne s'est pas engagé sur le fait sa recherche était terminée.

Prédicat 19 – ResearchNotOver **:**

$T_i \not\ni C(x, \text{researchOver}, \textbf{Crt})$

Le prédicat ResearchOverNotAsked est vrai lorsque le système n'a pas demandé à l'utilisateur si sa recherche était terminée depuis la dernière fois que l'utilisateur a signalé le fait que les résultats de la requête courante sont satisfaisants.

Prédicat 20 – ResearchOverNotAsked **:**

La définition de ce prédicat oblige à manipuler des engagements en actions dialogiques pour savoir si le système à demandé à l'utilisateur si sa recherche est terminée et quand. Les engagements en actions dialogiques n'étant pas traités dans ce manuscrit, nous ne donnons pas une définition formelle de ce prédicat ici.

A.3.2 Sortie

Le prédicat ResearchOver est vrai si l'utilisateur a dit avoir terminé sa recherche. Il vérifie donc que celui-ci s'est engagé sur la proposition de fin de recherche.

Prédicat 21 – ResearchOver **:**

$T_i \ni C(x, \text{researchOver}, \textbf{Crt})$

A.4 Réparation de la requête

Le prédicat RelevantForRemoval(kw) est vrai si la suppression du mot-clé kw de la requête est pertinente.

Prédicat 22 – RelevantForRemoval(kw) **:**

Ce prédicat fait appel à des ressources de recherche d'information et du domaine pour être

calculé.

Le prédicat RelevantForSubstitution(*kw*, *skw*) est vrai si le remplacement du mot-clé *kw* par *skw* est pertinent pour modifier la requête.

Prédicat 23 – RelevantForSubstitution(*kw*, *skw*) **:**

Ce prédicat fait appel à des ressources de recherche d'information et du domaine pour être calculé.

Annexe B

Dialogues extraits des simulations

Cette annexe présente des exemples de dialogues extraits des simulations réalisées pour les expériences réalisées pour le chapitre 6. Afin de ne pas occuper trop d'espace, des dialogues de petite taille ont été sélectionnés pour illustrer certains cas de figure discutés lors de la présentation des résultats. L'intégralité des dialogues est disponible dans le dépôt hébergeant le code source de ce manuscrit, à l'adresse `https://gitlab.insa-rouen.fr/jlouvet/manuscrit-these`, dans le répertoire « Données expérimentales ».

B.1 Expériences d'initiative utilisateur/système/mixte et violation d'un engagement en action

B.1.1 Initiative utilisateur

Le dialogue B.1 est extrait des dialogues simulés lors de l'expérience sur la modalité où seul l'utilisateur prend des initiatives au cours de l'interaction.

	Acte	E_s	E_h
0	inform(Agent-XorG, function(helpForIR))	O	O
1	agreement(Human-John, function(helpForIR))	O	O
2	inform(Human-John, verbalizationExpression(Expr1))	FV	FV
3	agreement(Agent-XorG, verbalizationExpression(Expr1))	FV	FV
4	inform(Human-John, verbalizationComplete.)	FV	FV
5	agreement(Agent-XorG, verbalizationComplete.)	FV	FV
6	prop.in(Human-John, Request(addKeyWord(UKw1)))	AV/T	AV/T
7	acc.in(Agent-XorG, Request(addKeyWord(UKw1)))	AV/T	AV/T
8	request(Human-John, addKeyWord(UKw1))	AV/T	AV/T
9	acceptRequest(Agent-XorG, addKeyWord(UKw1))	AV/T	AV/T
10	prop.out(Human-John, Request(addKeyWord(UKw1)))	AV/T	AV/T
11	acc.out(Agent-XorG, Request(addKeyWord(UKw1)))	AV/T	AV/T
12	prop.in(Human-John, Request(launchCurrentQuery.))	LR	LR
13	acc.in(Agent-XorG, Request(launchCurrentQuery.))	LR	LR
14	request(Human-John, launchCurrentQuery.)	LR	LR
15	acceptRequest(Agent-XorG, launchCurrentQuery.)	LR	LR
16	prop.out(Human-John, Request(launchCurrentQuery.))	LR	LR
17	acc.out(Agent-XorG, Request(launchCurrentQuery.))	LR	LR
18	inform(Human-John, seenResource(Res1))	ER	ER
19	agreement(Agent-XorG, seenResource(Res1))	ER	ER
20	inform(Human-John, seenResource(Res2))	ER	ER
21	agreement(Agent-XorG, seenResource(Res2))	ER	ER
22	inform(Human-John, relevantResource(Res1))	ER	ER
23	agreement(Agent-XorG, relevantResource(Res1))	ER	ER
24	inform(Human-John, currentQResultsSatisfying.)	ER	ER
25	agreement(Agent-XorG, currentQResultsSatisfying.)	ER	ER
26	prop.in(Agent-XorG, Verification(researchOver.))	ER	ER
27	acc.in(Human-John, Verification(researchOver.))	ER	ER
28	checkQuestion(Agent-XorG, ?researchOver.)	ER	ER
29	confirm(Human-John, researchOver.)	ER	ER
30	prop.out(Agent-XorG, Verification(researchOver.))	ER	ER
31	acc.out(Human-John, Verification(researchOver.))	ER	ER
32	inform(Agent-XorG, interactionOver.)	ER	ER
33	agreement(Human-John, interactionOver.)	F	F

TABLEAU B.1 – Dialogue illustrant la modalité de test où seul l'utilisateur peut prendre des initiatives (dialogue simulé numéro 918848170). E_s est l'état du système à chaque tour de parole et E_h est l'état de l'humain à chaque tour de parole.

B.1.2 Initiative système

Le dialogue présenté dans les tableaux B.2 et B.3 est extrait des dialogues simulés lors de l'expérience sur la modalité où seul le système prend des initiatives au cours de l'interaction.

	Acte	E_s	E_h
0	inform(Agent-XorG, function(helpForIR))	O	O
1	agreement(Human-John, function(helpForIR))	O	O
2	inform(Human-John, verbalizationExpression(Expr1))	FV	FV
3	agreement(Agent-XorG, verbalizationExpression(Expr1))	FV	FV
4	inform(Human-John, verbalizationComplete.)	FV	FV
5	agreement(Agent-XorG, verbalizationComplete.)	FV	FV
6	prop.in(Agent-XorG, Request(preciseVerbalization.))	FV	FV
7	acc.in(Human-John, Request(preciseVerbalization.))	RP	RP
8	request(Agent-XorG, preciseVerbalization.)	RP	RP
9	acceptRequest(Human-John, preciseVerbalization.)	RP	RP
10	prop.out(Agent-XorG, Request(preciseVerbalization.))	RP	RP
11	acc.out(Human-John, Request(preciseVerbalization.))	RP	RP
12	inform(Human-John, verbalizationExpression(Expr2))	PV	PV
13	agreement(Agent-XorG, verbalizationExpression(Expr2))	PV	PV
14	inform(Human-John, verbalizationComplete.)	PV	PV
15	agreement(Agent-XorG, verbalizationComplete.)	PV	PV
16	prop.in(Agent-XorG, Offer(addKeyWord(AKw2)))	PV	PV
17	acc.in(Human-John, Offer(addKeyWord(AKw2)))	AV/T	AV/T
18	offer(Agent-XorG, addKeyWord(AKw2))	AV/T	AV/T
19	acceptOffer(Human-John, addKeyWord(AKw2))	AV/T	AV/T
20	prop.out(Agent-XorG, Offer(addKeyWord(AKw2)))	AV/T	AV/T
21	acc.out(Human-John, Offer(addKeyWord(AKw2)))	AV/T	AV/T
22	prop.in(Agent-XorG, Offer(addKeyWord(AKw1)))	AV/T	AV/T
23	acc.in(Human-John, Offer(addKeyWord(AKw1)))	AV/T	AV/T
24	offer(Agent-XorG, addKeyWord(AKw1))	AV/T	AV/T
25	acceptOffer(Human-John, addKeyWord(AKw1))	AV/T	AV/T

TABLEAU B.2 – Première partie du dialogue illustrant la modalité de test où seul le système peut prendre des initiatives (dialogue simulé numéro 918052172). E_s est l'état du système à chaque tour de parole et E_h est l'état de l'humain à chaque tour de parole.

	Acte	E_s	E_h
26	prop.out(Agent-XorG, Offer(addKeyWord(AKw1)))	AV/T	AV/T
27	acc.out(Human-John, Offer(addKeyWord(AKw1)))	AV/T	AV/T
28	prop.in(Agent-XorG, Offer(addKeyWord(AKw4)))	AV/T	AV/T
29	acc.in(Human-John, Offer(addKeyWord(AKw4)))	AV/T	AV/T
30	offer(Agent-XorG, addKeyWord(AKw4))	AV/T	AV/T
31	acceptOffer(Human-John, addKeyWord(AKw4))	AV/T	AV/T
32	prop.out(Agent-XorG, Offer(addKeyWord(AKw4)))	AV/T	AV/T
33	acc.out(Human-John, Offer(addKeyWord(AKw4)))	AV/T	AV/T
34	prop.in(Agent-XorG, Offer(addKeyWord(AKw3)))	AV/T	AV/T
35	acc.in(Human-John, Offer(addKeyWord(AKw3)))	AV/T	AV/T
36	offer(Agent-XorG, addKeyWord(AKw3))	AV/T	AV/T
37	acceptOffer(Human-John, addKeyWord(AKw3))	AV/T	AV/T
38	prop.out(Agent-XorG, Offer(addKeyWord(AKw3)))	AV/T	AV/T
39	acc.out(Human-John, Offer(addKeyWord(AKw3)))	AV/T	AV/T
40	prop.in(Agent-XorG, Offer(launchCurrentQuery.))	AV/T	AV/T
41	acc.in(Human-John, Offer(launchCurrentQuery.))	LR	LR
42	offer(Agent-XorG, launchCurrentQuery.)	LR	LR
43	acceptOffer(Human-John, launchCurrentQuery.)	LR	LR
44	prop.out(Agent-XorG, Offer(launchCurrentQuery.))	LR	LR
45	acc.out(Human-John, Offer(launchCurrentQuery.))	LR	LR
46	inform(Human-John, seenResource(Res1))	ER	ER
47	agreement(Agent-XorG, seenResource(Res1))	ER	ER
48	inform(Human-John, currentQResultsSatisfying.)	ER	ER
49	agreement(Agent-XorG, currentQResultsSatisfying.)	ER	ER
50	prop.in(Agent-XorG, Verification(researchOver.))	ER	ER
51	acc.in(Human-John, Verification(researchOver.))	ER	ER
52	checkQuestion(Agent-XorG, ?researchOver.)	ER	ER
53	confirm(Human-John, researchOver.)	ER	ER
54	prop.out(Agent-XorG, Verification(researchOver.))	ER	ER
55	acc.out(Human-John, Verification(researchOver.))	ER	ER
56	inform(Agent-XorG, interactionOver.)	ER	ER
57	agreement(Human-John, interactionOver.)	F	F

TABLEAU B.3 – Seconde partie du dialogue illustrant la modalité de test où seul le système peut prendre des initiatives (dialogue simulé numéro 918052172). E_s est l'état du système à chaque tour de parole et E_h est l'état de l'humain à chaque tour de parole.

B.1.3 Initiative mixte

Le dialogue présenté dans les tableaux B.4 et B.5 est extrait des dialogues simulés lors de l'expérience sur la modalité où l'initiative est mixte au cours de l'interaction.

	Acte	E_s	E_h
0	inform(Agent-XorG, function(helpForIR))	O	O
1	agreement(Human-John, function(helpForIR))	O	O
2	inform(Human-John, verbalizationExpression(Expr1))	FV	FV
3	agreement(Agent-XorG, verbalizationExpression(Expr1))	FV	FV
4	inform(Human-John, verbalizationComplete.)	FV	FV
5	agreement(Agent-XorG, verbalizationComplete.)	FV	FV
6	prop.in(Human-John, Request(addKeyWord(UKw1)))	AV/T	AV/T
7	acc.in(Agent-XorG, Request(addKeyWord(UKw1)))	AV/T	AV/T
8	request(Human-John, addKeyWord(UKw1))	AV/T	AV/T
9	acceptRequest(Agent-XorG, addKeyWord(UKw1))	AV/T	AV/T
10	prop.out(Human-John, Request(addKeyWord(UKw1)))	AV/T	AV/T
11	acc.out(Agent-XorG, Request(addKeyWord(UKw1)))	AV/T	AV/T
12	prop.in(Agent-XorG, Offer(addKeyWord(AKw2)))	AV/T	AV/T
13	acc.in(Human-John, Offer(addKeyWord(AKw2)))	AV/T	AV/T
14	offer(Agent-XorG, addKeyWord(AKw2))	AV/T	AV/T
15	acceptOffer(Human-John, addKeyWord(AKw2))	AV/T	AV/T
16	prop.out(Agent-XorG, Offer(addKeyWord(AKw2)))	AV/T	AV/T
17	acc.out(Human-John, Offer(addKeyWord(AKw2)))	AV/T	AV/T
18	prop.in(Agent-XorG, Offer(addKeyWord(AKw1)))	AV/T	AV/T
19	acc.in(Human-John, Offer(addKeyWord(AKw1)))	AV/T	AV/T
20	offer(Agent-XorG, addKeyWord(AKw1))	AV/T	AV/T
21	acceptOffer(Human-John, addKeyWord(AKw1))	AV/T	AV/T
22	prop.out(Agent-XorG, Offer(addKeyWord(AKw1)))	AV/T	AV/T
23	acc.out(Human-John, Offer(addKeyWord(AKw1)))	AV/T	AV/T
24	prop.in(Agent-XorG, Offer(launchCurrentQuery.))	AV/T	AV/T
25	acc.in(Human-John, Offer(launchCurrentQuery.))	LR	LR
26	offer(Agent-XorG, launchCurrentQuery.)	LR	LR
27	acceptOffer(Human-John, launchCurrentQuery.)	LR	LR

TABLEAU B.4 – Première partie du dialogue illustrant la modalité de test où l'initiative est mixte(dialogue simulé numéro 857304365). E_s est l'état du système à chaque tour de parole et E_h est l'état de l'humain à chaque tour de parole.

	Acte	E_s	E_h
29	acc.out(Human-John, Offer(launchCurrentQuery.))	LR	LR
30	inform(Human-John, seenResource(Res1))	ER	ER
31	agreement(Agent-XorG, seenResource(Res1))	ER	ER
32	inform(Human-John, currentQResultsNotSatisfying.)	ER	ER
33	agreement(Agent-XorG, currentQResultsNotSatisfying.)	ER	ER
34	prop.in(Agent-XorG, Offer(substituteKeyWord(UKw1, AKw3)))	ER	ER
35	acc.in(Human-John, Offer(substituteKeyWord(UKw1, AKw3)))	RR	RR
36	offer(Agent-XorG, substituteKeyWord(UKw1, AKw3))	RR	RR
37	acceptOffer(Human-John, substituteKeyWord(UKw1, AKw3))	RR	RR
38	prop.out(Agent-XorG, Offer(substituteKeyWord(UKw1, AKw3)))	RR	RR
39	acc.out(Human-John, Offer(substituteKeyWord(UKw1, AKw3)))	RR	RR
40	prop.in(Human-John, Request(launchCurrentQuery.))	LR	LR
41	acc.in(Agent-XorG, Request(launchCurrentQuery.))	LR	LR
42	request(Human-John, launchCurrentQuery.)	LR	LR
43	acceptRequest(Agent-XorG, launchCurrentQuery.)	LR	LR
44	prop.out(Human-John, Request(launchCurrentQuery.))	LR	LR
45	acc.out(Agent-XorG, Request(launchCurrentQuery.))	LR	LR
46	inform(Human-John, seenResource(Res2))	ER	ER
47	agreement(Agent-XorG, seenResource(Res2))	ER	ER
48	inform(Human-John, relevantResource(Res1))	ER	ER
49	agreement(Agent-XorG, relevantResource(Res1))	ER	ER
50	inform(Human-John, seenResource(Res3))	ER	ER
51	agreement(Agent-XorG, seenResource(Res3))	ER	ER
52	inform(Human-John, relevantResource(Res2))	ER	ER
53	agreement(Agent-XorG, relevantResource(Res2))	ER	ER
54	inform(Human-John, seenResource(Res4))	ER	ER
55	agreement(Agent-XorG, seenResource(Res4))	ER	ER
56	inform(Human-John, currentQResultsSatisfying.)	ER	ER
57	agreement(Agent-XorG, currentQResultsSatisfying.)	ER	ER
58	prop.in(Agent-XorG, Verification(researchOver.))	ER	ER
59	acc.in(Human-John, Verification(researchOver.))	ER	ER
60	checkQuestion(Agent-XorG, ?researchOver.)	ER	ER
61	confirm(Human-John, researchOver.)	ER	ER
62	prop.out(Agent-XorG, Verification(researchOver.))	ER	ER
63	acc.out(Human-John, Verification(researchOver.))	ER	ER
64	inform(Agent-XorG, interactionOver.)	ER	ER
65	agreement(Human-John, interactionOver.)	F	F

TABLEAU B.5 – Seconde partie du dialogue illustrant la modalité de test où l'initiative est mixte(dialogue simulé numéro 857304365). E_s est l'état du système à chaque tour de parole et E_h est l'état de l'humain à chaque tour de parole.

B.1.4 Violation d'un engagement en action

Le dialogue présenté dans les tableaux B.6 et B.7 est extrait des dialogues simulés lors de l'expérience sur la modalité où l'utilisateur peut violer son engagement en action sur la précision de sa verbalisation.

	Acte	E_s	E_h
0	inform(Agent-XorG, function(helpForIR))	O	O
1	agreement(Human-John, function(helpForIR))	O	O
2	inform(Human-John, verbalizationExpression(Expr1))	FV	FV
3	agreement(Agent-XorG, verbalizationExpression(Expr1))	FV	FV
4	inform(Human-John, verbalizationExpression(Expr2))	FV	FV
5	agreement(Agent-XorG, verbalizationExpression(Expr2))	FV	FV
6	inform(Human-John, verbalizationComplete.)	FV	FV
7	agreement(Agent-XorG, verbalizationComplete.)	FV	FV
8	prop.in(Agent-XorG, Request(preciseVerbalization.))	FV	FV
9	acc.in(Human-John, Request(preciseVerbalization.))	RP	RP
10	request(Agent-XorG, preciseVerbalization.)	RP	RP
11	acceptRequest(Human-John, preciseVerbalization.)	RP	RP
12	prop.out(Agent-XorG, Request(preciseVerbalization.))	RP	RP
13	acc.out(Human-John, Request(preciseVerbalization.))	RP	RP
14	inform(Human-John, verbalizationComplete.)	PV	PV
15	agreement(Agent-XorG, verbalizationComplete.)	PV	PV
16	prop.in(Human-John, Request(addKeyWord(UKw1)))	AV/T	AV/T
17	acc.in(Agent-XorG, Request(addKeyWord(UKw1)))	AV/T	AV/T
18	request(Human-John, addKeyWord(UKw1))	AV/T	AV/T
19	acceptRequest(Agent-XorG, addKeyWord(UKw1))	AV/T	AV/T
20	prop.out(Human-John, Request(addKeyWord(UKw1)))	AV/T	AV/T
21	acc.out(Agent-XorG, Request(addKeyWord(UKw1)))	AV/T	AV/T
22	prop.in(Agent-XorG, Offer(addKeyWord(AKw2)))	AV/T	AV/T
23	acc.in(Human-John, Offer(addKeyWord(AKw2)))	AV/T	AV/T
24	offer(Agent-XorG, addKeyWord(AKw2))	AV/T	AV/T
25	acceptOffer(Human-John, addKeyWord(AKw2))	AV/T	AV/T
26	prop.out(Agent-XorG, Offer(addKeyWord(AKw2)))	AV/T	AV/T
27	acc.out(Human-John, Offer(addKeyWord(AKw2)))	AV/T	AV/T

TABLEAU B.6 – Première partie du dialogue illustrant la modalité de test de violation d'un engagement en action par l'utilisateur (dialogue simulé numéro 908880938). E_s est l'état du système à chaque tour de parole et E_h est l'état de l'humain à chaque tour de parole.

	Acte	E_s	E_h
28	prop.in(Agent-XorG, Offer(addKeyWord(AKw1)))	AV/T	AV/T
29	acc.in(Human-John, Offer(addKeyWord(AKw1)))	AV/T	AV/T
30	offer(Agent-XorG, addKeyWord(AKw1))	AV/T	AV/T
31	acceptOffer(Human-John, addKeyWord(AKw1))	AV/T	AV/T
32	prop.out(Agent-XorG, Offer(addKeyWord(AKw1)))	AV/T	AV/T
33	acc.out(Human-John, Offer(addKeyWord(AKw1)))	AV/T	AV/T
34	prop.in(Human-John, Request(addKeyWord(UKw2)))	AV/T	AV/T
35	acc.in(Agent-XorG, Request(addKeyWord(UKw2)))	AV/T	AV/T
36	request(Human-John, addKeyWord(UKw2))	AV/T	AV/T
37	acceptRequest(Agent-XorG, addKeyWord(UKw2))	AV/T	AV/T
38	prop.out(Human-John, Request(addKeyWord(UKw2)))	AV/T	AV/T
39	acc.out(Agent-XorG, Request(addKeyWord(UKw2)))	AV/T	AV/T
40	prop.in(Human-John, Request(addKeyWord(UKw3)))	AV/T	AV/T
41	acc.in(Agent-XorG, Request(addKeyWord(UKw3)))	AV/T	AV/T
42	request(Human-John, addKeyWord(UKw3))	AV/T	AV/T
43	acceptRequest(Agent-XorG, addKeyWord(UKw3))	AV/T	AV/T
44	prop.out(Human-John, Request(addKeyWord(UKw3)))	AV/T	AV/T
45	acc.out(Agent-XorG, Request(addKeyWord(UKw3)))	AV/T	AV/T
46	prop.in(Human-John, Request(launchCurrentQuery.))	LR	LR
47	acc.in(Agent-XorG, Request(launchCurrentQuery.))	LR	LR
48	request(Human-John, launchCurrentQuery.)	LR	LR
49	acceptRequest(Agent-XorG, launchCurrentQuery.)	LR	LR
50	prop.out(Human-John, Request(launchCurrentQuery.))	LR	LR
51	acc.out(Agent-XorG, Request(launchCurrentQuery.))	LR	LR
52	inform(Human-John, seenResource(Res1))	ER	ER
53	agreement(Agent-XorG, seenResource(Res1))	ER	ER
54	inform(Human-John, seenResource(Res2))	ER	ER
55	agreement(Agent-XorG, seenResource(Res2))	ER	ER
56	inform(Human-John, currentQResultsSatisfying.)	ER	ER
57	agreement(Agent-XorG, currentQResultsSatisfying.)	ER	ER
58	prop.in(Agent-XorG, Verification(researchOver.))	ER	ER
59	acc.in(Human-John, Verification(researchOver.))	ER	ER
60	checkQuestion(Agent-XorG, ?researchOver.)	ER	ER
61	confirm(Human-John, researchOver.)	ER	ER
62	prop.out(Agent-XorG, Verification(researchOver.))	ER	ER
63	acc.out(Human-John, Verification(researchOver.))	ER	ER
64	inform(Agent-XorG, interactionOver.)	ER	ER
65	agreement(Human-John, interactionOver.)	F	F

TABLEAU B.7 – Seconde partie du dialogue illustrant la modalité de test de violation d'un engagement en action par l'utilisateur (dialogue simulé numéro 908880938). E_s est l'état du système à chaque tour de parole et E_h est l'état de l'humain à chaque tour de parole.

B.2 Mise à jour du besoin d'information, rejet implicite et saut en avant

B.2.1 Saut en avant : inconsistance logique

Le dialogue présenté dans le tableau B.8 est extrait des dialogues simulés lors de l'expérience sur la modalité où l'utilisateur peut réaliser des sauts en avant au cours de la réalisation de la tâche.

Au cours de ce dialogue, l'utilisateur a réalisé deux sauts en avant :

— au tour de parole 4, il a sauté au-dessus du comportement annonçant que sa verbalisation était complète en initiant un comportement de demande d'ajout de mot-clé à la requête ;

— au tour de parole 30, il a sauté au-dessus du comportement demandant l'ajout d'un mot-clé à la requête.

L'occurrence de ce second saut, suivi de l'expression de la satisfaction de l'utilisateur par les résultats de la requête courante a mené le système à considérer que l'utilisateur est à la fois satisfait et non-satisfait par les résultats d'une même requête, créant une inconsistance logique et menant le système à s'arrêter.

	Acte	E_s	E_h
0	inform(Agent-XorG, function(helpForIR))	O	O
1	agreement(Human-John, function(helpForIR))	O	O
2	inform(Human-John, verbalizationExpression(Expr1))	FV	FV
3	agreement(Agent-XorG, verbalizationExpression(Expr1))	FV	FV
4	prop.in(Human-John, Request(addKeyWord(UKw1)))	AV/T	AV/T
5	acc.in(Agent-XorG, Request(addKeyWord(UKw1)))	AV/T	AV/T
6	request(Human-John, addKeyWord(UKw1))	AV/T	AV/T
7	acceptRequest(Agent-XorG, addKeyWord(UKw1))	AV/T	AV/T
8	prop.out(Human-John, Request(addKeyWord(UKw1)))	AV/T	AV/T
9	acc.out(Agent-XorG, Request(addKeyWord(UKw1)))	AV/T	AV/T
10	prop.in(Human-John, Request(addKeyWord(UKw2)))	AV/T	AV/T
11	acc.in(Agent-XorG, Request(addKeyWord(UKw2)))	AV/T	AV/T
12	request(Human-John, addKeyWord(UKw2))	AV/T	AV/T
13	acceptRequest(Agent-XorG, addKeyWord(UKw2))	AV/T	AV/T
14	prop.out(Human-John, Request(addKeyWord(UKw2)))	AV/T	AV/T
15	acc.out(Agent-XorG, Request(addKeyWord(UKw2)))	AV/T	AV/T
16	prop.in(Human-John, Request(launchCurrentQuery.))	LR	LR
17	acc.in(Agent-XorG, Request(launchCurrentQuery.))	LR	LR
18	request(Human-John, launchCurrentQuery.)	LR	LR
19	acceptRequest(Agent-XorG, launchCurrentQuery.)	LR	LR
20	prop.out(Human-John, Request(launchCurrentQuery.))	LR	LR
21	acc.out(Agent-XorG, Request(launchCurrentQuery.))	LR	LR
22	inform(Human-John, seenResource(Res1))	ER	ER
23	agreement(Agent-XorG, seenResource(Res1))	ER	ER
24	inform(Human-John, relevantResource(Res1))	ER	ER
25	agreement(Agent-XorG, relevantResource(Res1))	ER	ER
26	inform(Human-John, relevantResource(Res2))	ER	ER
27	agreement(Agent-XorG, relevantResource(Res2))	ER	ER
28	inform(Human-John, currentQResultsNotSatisfying.)	ER	ER
29	agreement(Agent-XorG, currentQResultsNotSatisfying.)	ER	ER
30	prop.in(Human-John, Request(launchCurrentQuery.))	LR	LR
31	acc.in(Agent-XorG, Request(launchCurrentQuery.))	LR	LR
32	request(Human-John, launchCurrentQuery.)	LR	LR
33	acceptRequest(Agent-XorG, launchCurrentQuery.)	LR	LR
34	prop.out(Human-John, Request(launchCurrentQuery.))	LR	LR
35	acc.out(Agent-XorG, Request(launchCurrentQuery.))	LR	LR
36	inform(Human-John, currentQResultsSatisfying.)	ER	ER

TABLEAU B.8 – Dialogue illustrant une inconsistance logique arrivant au cours de la modalité de test où l'utilisateur peut réaliser un saut en avant (dialogue simulé numéro 837888354). E_s est l'état du système à chaque tour de parole et E_h est l'état de l'humain à chaque tour de parole.

B.2.2 Rejet implicite : échec de contextualisation

Le dialogue présenté dans les tableaux B.9 et B.10 est extrait des dialogues simulés lors de l'expérience sur la modalité où l'utilisateur peut rejeter implicitement les résultats de la requête. Ce dialogue illustre un enchaînement possible menant à deux échecs de contextualisation (actes 30 et 37) de la part du système. Chaque échec de contextualisation fait suite à une altération du comportement conventionnel de l'utilisateur où celui-ci rejette implicitement les résultats de la requête. Dans ces situations, lorsqu'il demande une modification de la requête (ajouter ou supprimer un mot-clé), les prérequis de l'état de réparation de la requête ne sont pas satisfaits et ne permettent donc pas d'y contextualiser sa demande en réalisant une transition vers un état candidat. De plus, le système n'étant pas capable (dans le cas de ce dialogue en particulier) de calculer des modifications pouvant être apportées à la requête (aucun prédicat n'est déduit permettant de satisfaire les préconditions d'un comportement réalisable), la maturité des états où il serait possible de contextualiser un comportement de l'utilisateur est nulle, et donc il n'est pas possible de réaliser une transition vers un état mature. Cela contraint le système à ignorer les initiatives prises par l'utilisateur et mène à un échec de contextualisation.

Le cas de figure présenté ici n'est pas le seul possible et dans les simulations faites selon la modalité du rejet implicite, une autre possibilité a été observée. Celle-ci a lieu suite à un décrochage du système, ce dernier se retrouvant dans l'état d'alignement verbalisation/terminologie alors que l'utilisateur est dans l'état de réparation de la requête. Les règles des comportement attendus de ces deux états n'étant pas les mêmes, le système n'interprète pas correctement les modifications apportées par l'utilisateur sur la requête, et notamment que celles-ci annulent le fait que la requête courante a été lancée. Dans ce cas, lorsque l'utilisateur réalise une demande de lancement de la requête, le système échoue à contextualiser cette initiative car les prérequis de l'état de lancement de la requête ne sont pas satisfaits et sa maturité est nulle.

	Acte	E_s	E_h
0	inform(Agent-XorG, function(helpForIR))	O	O
1	agreement(Human-John, function(helpForIR))	O	O
2	inform(Human-John, verbalizationExpression(Expr1))	FV	FV
3	agreement(Agent-XorG, verbalizationExpression(Expr1))	FV	FV
4	inform(Human-John, verbalizationComplete.)	FV	FV
5	agreement(Agent-XorG, verbalizationComplete.)	FV	FV
6	prop.in(Human-John, Request(addKeyWord(UKw1)))	AV/T	AV/T
7	acc.in(Agent-XorG, Request(addKeyWord(UKw1)))	AV/T	AV/T
8	request(Human-John, addKeyWord(UKw1))	AV/T	AV/T
9	acceptRequest(Agent-XorG, addKeyWord(UKw1))	AV/T	AV/T
10	prop.out(Human-John, Request(addKeyWord(UKw1)))	AV/T	AV/T
11	acc.out(Agent-XorG, Request(addKeyWord(UKw1)))	AV/T	AV/T
12	prop.in(Agent-XorG, Offer(addKeyWord(AKw2)))	AV/T	AV/T
13	acc.in(Human-John, Offer(addKeyWord(AKw2)))	AV/T	AV/T
14	offer(Agent-XorG, addKeyWord(AKw2))	AV/T	AV/T
15	acceptOffer(Human-John, addKeyWord(AKw2))	AV/T	AV/T
16	prop.out(Agent-XorG, Offer(addKeyWord(AKw2)))	AV/T	AV/T
17	acc.out(Human-John, Offer(addKeyWord(AKw2)))	AV/T	AV/T
18	prop.in(Agent-XorG, Offer(addKeyWord(AKw1)))	AV/T	AV/T
19	acc.in(Human-John, Offer(addKeyWord(AKw1)))	AV/T	AV/T
20	offer(Agent-XorG, addKeyWord(AKw1))	AV/T	AV/T
21	acceptOffer(Human-John, addKeyWord(AKw1))	AV/T	AV/T
22	prop.out(Agent-XorG, Offer(addKeyWord(AKw1)))	AV/T	AV/T
23	acc.out(Human-John, Offer(addKeyWord(AKw1)))	AV/T	AV/T
24	prop.in(Agent-XorG, Offer(launchCurrentQuery.))	AV/T	AV/T
25	acc.in(Human-John, Offer(launchCurrentQuery.))	LR	LR

TABLEAU B.9 – Première partie du dialogue illustrant les échecs de contextualisation dans la modalité de test où l'utilisateur peut faire des rejets implicites (dialogue simulé numéro 772406920). E_s est l'état du système à chaque tour de parole et E_h est l'état de l'humain à chaque tour de parole.

	Acte	E_s	E_h
26	offer(Agent-XorG, launchCurrentQuery.)	LR	LR
27	acceptOffer(Human-John, launchCurrentQuery.)	LR	LR
28	prop.out(Agent-XorG, Offer(launchCurrentQuery.))	LR	LR
29	acc.out(Human-John, Offer(launchCurrentQuery.))	LR	LR
30	prop.in(Human-John, Request(removeKeyWord(UKw1)))	LR	RR
31	prop.in(Human-John, Request(launchCurrentQuery.))	LR	LR
32	acc.in(Agent-XorG, Request(launchCurrentQuery.))	LR	LR
33	request(Human-John, launchCurrentQuery.)	LR	LR
34	acceptRequest(Agent-XorG, launchCurrentQuery.)	LR	LR
35	prop.out(Human-John, Request(launchCurrentQuery.))	LR	LR
36	acc.out(Agent-XorG, Request(launchCurrentQuery.))	LR	LR
37	prop.in(Human-John, Request(addKeyWord(UKw2)))	LR	RR
38	prop.in(Human-John, Request(launchCurrentQuery.))	LR	LR
39	acc.in(Agent-XorG, Request(launchCurrentQuery.))	LR	LR
40	request(Human-John, launchCurrentQuery.)	LR	LR
41	acceptRequest(Agent-XorG, launchCurrentQuery.)	LR	LR
42	prop.out(Human-John, Request(launchCurrentQuery.))	LR	LR
43	acc.out(Agent-XorG, Request(launchCurrentQuery.))	LR	LR
44	inform(Human-John, seenResource(Res1))	ER	ER
45	agreement(Agent-XorG, seenResource(Res1))	ER	ER
46	inform(Human-John, currentQResultsSatisfying.)	ER	ER
47	agreement(Agent-XorG, currentQResultsSatisfying.)	ER	ER
48	prop.in(Agent-XorG, Verification(researchOver.))	ER	ER
49	acc.in(Human-John, Verification(researchOver.))	ER	ER
50	checkQuestion(Agent-XorG, ?researchOver.)	ER	ER
51	confirm(Human-John, researchOver.)	ER	ER
52	prop.out(Agent-XorG, Verification(researchOver.))	ER	ER
53	acc.out(Human-John, Verification(researchOver.))	ER	ER
54	inform(Agent-XorG, interactionOver.)	ER	ER
55	agreement(Human-John, interactionOver.)	F	F

TABLEAU B.10 – Second partie du dialogue illustrant les échecs de contextualisation dans la modalité de test où l'utilisateur peut faire des rejets implicites (dialogue simulé numéro 772406920). E_s est l'état du système à chaque tour de parole et E_h est l'état de l'humain à chaque tour de parole.

Annexe C

Figures annexes

C.1 Hiérarchie DIT++

La figure C.1 pérsente les fonctions générales définies dans la taxonomie DIT++.

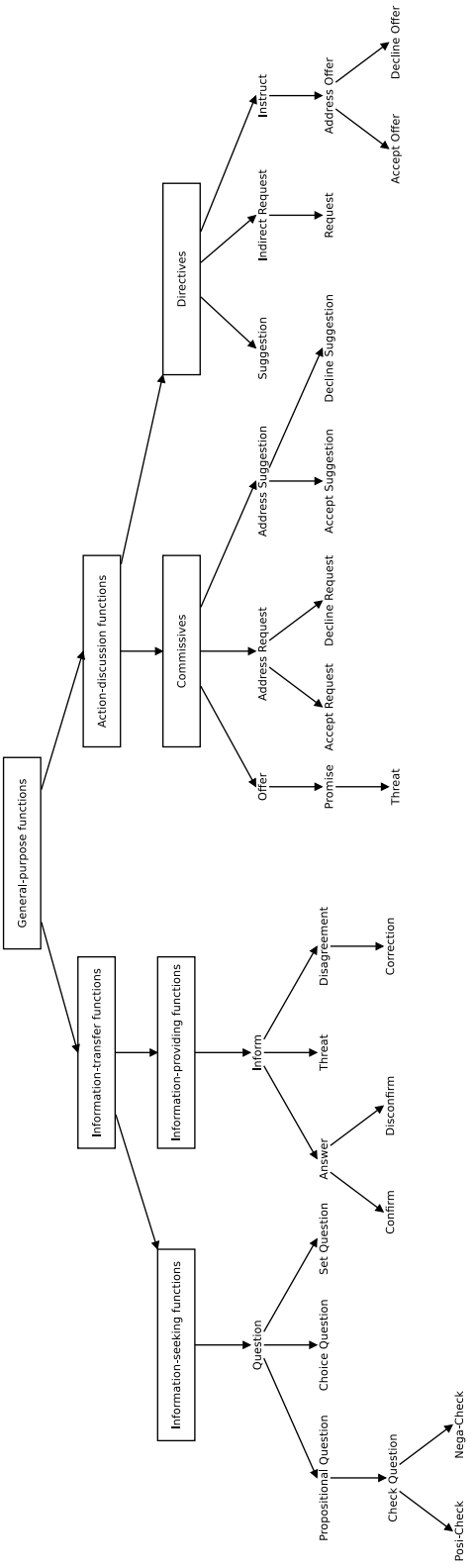

FIGURE C.1 – Fonctions générales de DIT++ (Repris et adapté de [Bun09]).

Bibliographie

[ABF02] James Allen, Nate Blaylock, and George Ferguson. A problem solving model for collaborative agents. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems : Part 2*, AAMAS '02, pages 774–781, New York, NY, USA, 2002. ACM. 111

[AD94] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2) :183–235, 1994. 59

[AFM+00] James F. Allen, George Ferguson, Bradford W. Miller, Eric K. Ringger, and Teresa Sikorski. *Dialogue Systems : From Theory to Practice in TRAINS-96*, chapter 14, pages 347–376. Marcel Dekker, 1 edition, July 2000. 25, 58

[Ann03] John Annett. Hierarchical task analysis. In *Handbook of Cognitive Task Design*, pages 17–36. CRC Press, jun 2003. 27

[AP80] James F. Allen and C.Raymond Perrault. Analyzing intention in utterances. *Artificial Intelligence*, 15(3) :143–178, 1980. 54

[APC+07] John Adcock, Jeremy Pickens, Matthew L Cooper, Lisa Anthony, Francine Chen, and Pernilla Qvarfordt. Fxpal interactive search experiments for trecvid 2007. In *TRECVid*. Citeseer, 2007. 38

[ASF+95] James F. Allen, Lenhart K. Schubert, George Ferguson, Peter Heeman, Chung Hee Hwang, Tsuneaki Kato, Marc Light, Nathaniel Martin, Bradford Miller, Massimo Poesio, and David R. Traum. The trains project : a case study in building a conversational planning agent. *Journal of Experimental & Theoretical Artificial Intelligence*, 7(1) :7–48, 1995. 25, 58

[ATP95] Marilyn Jager Adams, Yvette J. Tenney, and Richard W. Pew. Situation awareness and the cognitive management of complex systems. *Human Factors*, 37(1) :85–104, 1995. 21

[Aus62] John Langshaw Austin. *How to Do Things with Words*. The William James lectures ; 1955. Clarendon Press, 1962. 44, 47

[BA05] Nate Blaylock and James Allen. A collaborative problem-solving model of dialogue. In *6th SIGdial Workshop on Discourse and Dialogue*, 2005. 111

[Bat79] Marcia J. Bates. Information search tactics. *JASIS*, 30(4) :205–214, 1979. 32, 130

[Bat89] Marcia J. Bates. The design of browsing and berrypicking techniques for the online search interface. *Online Review*, 13(5) :407–424, 1989. 32, 130

[Bat90] Marcia J. Bates. Where should the person stop and the information search interface start ? *Inf. Process. Manage.*, 26(5) :575–591, 1990. 32, 130, 152

[BHSB16] Susanne Biundo, Daniel Höller, Bernd Schattenberg, and Pascal Bercher. Companion-technology : An overview. *KI - Künstliche Intelligenz*, 30(1) :11–20, Feb 2016. 29

[BKLO03] Johan Bos, Ewan Klein, Oliver Lemon, and Tetsushi Oka. Dipper : Description and formalisation of an information-state update dialogue system architecture. In *SIGDIAL Workshop*, pages 115–124. The Association for Computer Linguistics, 2003. 111

[BLB+17] Gregor Behnke, Benedikt Leichtmann, Pascal Bercher, Daniel Höller, Verena Nitsch, Martin Baumann, and Susanne Biundo. Help me make a dinner ! challenges when assisting humans in action planning. In *Proceedings of the International Conference on Companion Technology, Ulm*, volume 11, page 2017, 2017. 29, 86

[BR09] Dan Bohus and Alexander I. Rudnicky. The ravenclaw dialog management framework : Architecture and systems. *Computer Speech & Language*, 23(3) :332–361, 2009. 110

[Bre13] Susan E. Brennan. *Conversation and Dialogue*, volume 1. SAGE Publications, 2013. 44

[Bro02] A. Broder. A taxonomy of web search. *ACM SIGIR Forum*, 36(2) :3–10, 2002. 31

[BTNB00] HC Bunt, MM Taylor, F Neel, and DG Bouwhuis. Dynamic interpretation and dialogue theory. *The Structure of Multimodal Dialogue, Vol. 2*, pages 139–166, 2000. 49, 50

[Bun09] Harry Bunt. The dit++ taxonomy for functional dialogue markup. In Dirk Heylen, Catherine Pelachaud, Roberta Catizone, and David Traum, editors, *AAMAS 2009 Workshop, Towards a Standard Markup Language for Embodied Dialogue Acts*, pages 13–24, 2009. 49, 50, 51, 192, XXII

[Bun11a] Harry Bunt. *Interpretation and Generation of Dialogue with Multidimensional Context Models*, pages 214–242. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. 49

[Bun11b] Harry Bunt. Multifunctionality in dialogue. *Computer Speech & Language*, 25(2) :222–245, 2011. Language and speech issues in the engineering of companionable dialogue systems. 50

[BW16] Susanne Biundo and Andreas Wendemuth. Companion-technology for cognitive technical systems. *KI - Künstliche Intelligenz*, 30(1) :71–75, Feb 2016. 29

[BW17] S. Biundo and A. Wendemuth. *Companion Technology : A Paradigm Shift in Human-Technology Interaction*. Cognitive Technologies. Springer International Publishing, 2017. 29, 84

[Car90] Sandra Carberry. *Plan Recognition in Natural Language Dialogue*. MIT Press, Cambridge, MA, USA, 1990. 23, 55

[Cas95] Cristiano Castelfranchi. Guarantees for autonomy in cognitive agent architecture. In Michael J. Wooldridge and Nicholas R. Jennings, editors, *Intelligent Agents*, pages 56–70, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg. 16

[Cas16] D.O. Case. *Looking for Information : A Survey of Research on Information Seeking, Needs, and Behavior*. Library and information science. Elsevier/Academic Press, 4 edition, 2016. 30

[CCF+00] R. Scott Cost, Ye Chen, Tim Finin, Yannis Labrou, and Yun Peng. *Using Colored Petri Nets for Conversation Modeling*, pages 178–192. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000. 59

[CDH+10] Nathalie Chaignaud, Valérie Delavigne, Maryvonne Holzem, Jean-Philippe Kotowicz, and Alain Loisel. Étude cognitive des processus de construction d'une requête dans un système de gestion de connaissances médicales. *Revue des Sciences et Technologies de l'Information - Série TSI : Technique et Science Informatiques*, 29 :991–1021, 2010. 29 pages. 128

[CDLBP06] Brahim Chaib-Draa, Marc-André Labrie, Mathieu Bergeron, and Philippe Pasquier. Diagal : An agent communication language based on dialogue games and sustained by social commitments. *Autonomous Agents and Multi-Agent Systems*, 13(1) :61–95, July 2006. 53, 59, 61, 66, 67, 72, 79, 191

[CIS] Cismef – catalogue et index des sites médicaux de langue française. `http://www.chu-rouen.fr/cismef/`. Consulté le 13/09/2018. 127

[CKH06] Beth Crandall, Gary A Klein, and Robert R Hoffman. *Working minds : A practitioner's guide to cognitive task analysis*. Mit Press, 2006. 27

[CL90a] Philip R. Cohen and Hector J. Levesque. Intention is choice with commitment. *Artif. Intell.*, 42(2-3) :213–261, March 1990. 53, 65

[CL90b] Philip R. Cohen and Hector J. Levesque. *Rational Interaction as the Basis for Communication*, pages 221–255. MIT Press, 1990. 53

[Cla96] Herbert H. Clark. *Using Language*. 'Using' Linguistic Books. Cambridge University Press, 1996. 44, 46, 58, 61, 62, 77

[CP79] Philip R. Cohen and C. Raymond Perrault. Elements of a plan-based theory of speech acts. *Cognitive Science*, 3(3) :177–212, 1979. 54

[CS89] Herbert H. Clark and Edward F. Schaefer. Contributing to discourse. *Cognitive Science*, 13(2) :259–294, 1989. 62

[DCK+13] Guillaume Dubuisson Duplessis, Nathalie Chaignaud, Jean-Philippe Kotowicz, Alexandre Pauchet, and Jean-Pierre Pécuchet. Empirical specification

of dialogue games for an interactive agent. In *Advances on Practical Applications of Agents and Multi-Agent Systems*, pages 49–60. Springer, 2013. 71

[DD15] Guillaume Dubuisson Duplessis and Laurence Devillers. Towards the consideration of dialogue activities in engagement measures for human-robot social interaction. In *International Conference on Intelligent Robots and Systems*, Designing & Evaluating Social Robots for Public Settings Workshop, pages 19–24, Hambourg, Germany, September 2015. 99

[DLD+01] Stefan Jacques Darmoni, Jean-Philippe Leroy, Magali Douyère, Josette Piot, Saida Ouazir, Benoit Lacoste, Christophe Godard, Isabelle Rigolle, Martial Brisou, Stphane Videau, Myriam Quéré, Eric Goupy, Habib Abdulrab, and Benoit Thirion. Doc'CISMeF : un outil de recherche internet orienté vers l'enseignement et la formation à distance en médecine. *Pédagogie Médicale*, 2(3) :170–178, aug 2001. 128

[DPCK17] Guillaume Dubuisson Duplessis, Alexandre Pauchet, Nathalie Chaignaud, and Jean-Philippe Kotowicz. A conventional dialogue model based on dialogue patterns. *International Journal on Artificial Intelligence Tools*, 26(1) :1–23, 2017. 71, 111

[Dub14] Guillaume Dubuisson Duplessis. *Modèle de comportement communicatif conventionnel pour un agent en interaction avec des humains : Approche par jeux de dialogue.* PhD thesis, INSA de Rouen, 2014. 3, 64, 69, 71, 72, 73, 74, 75, 76, 77, 78, 89, 104, 111, 129, 130, 133, 191, 193

[DvdTYS17] Mehdi Dastani, Leendert van der Torre, and Neil Yorke-Smith. Commitments and interaction norms in organisations. *Autonomous Agents and Multi-Agent Systems*, 31(2) :207–249, Mar 2017. 67

[DY08] Peter J. Denning and Peter Yaholkovsky. Getting to "we". *Commun. ACM*, 51(4) :19–24, April 2008. 12, 14

[EE68] Douglas C. Engelbart and William K. English. A research center for augmenting human intellect. In *Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, Part I*, AFIPS '68 (Fall, part I), pages 395–410, New York, NY, USA, 1968. ACM. 1

[End95a] Mica R. Endsley. Measurement of situation awareness in dynamic systems. *Human Factors*, 37(1) :65–84, 1995. 19

[End95b] Mica R. Endsley. Toward a theory of situation awareness in dynamic systems. *Human Factors*, 37(1) :32–64, 1995. 19, 21

[End11] Mica R. Endsley. *Designing for Situation Awareness : An Approach to User-Centered Design.* CRC Press, Inc., Boca Raton, FL, USA, 2nd edition, 2011. 27

[End17] Mica R. Endsley. From here to autonomy : Lessons learned from human–automation research. *Human Factors*, 59(1) :5–27, 2017. PMID : 28146676. 21

[Eng62] Douglas C. Engelbart. Augmenting human intellect : A conceptual framework. Technical report, oct 1962. 1

[Eps15] Susan L. Epstein. Wanted : Collaborative intelligence. *Artificial Intelligence*, 221 :36–45, 2015. 10, 20, 24

[FA98] George Ferguson and James F. Allen. Trips : An integrated intelligent problem-solving assistant. In *Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence*, AAAI '98/IAAI '98, pages 567–572, Menlo Park, CA, USA, 1998. American Association for Artificial Intelligence. 25, 58

[FA02a] FIPA-ACL. *FIPA-ACL message structure specification*, 2002. 52, 193

[FA02b] FIPA-ACL. *FIPA Communicative Act Library Specification*, 2002. 52

[FA02c] FIPA-ACL. *FIPA SL Content Language Specification*, 2002. 53

[FA02d] FIPA-ACL. Interaction protocol specifications, 2002. 59

[FA17] Mary Ellen Foster and Ronald P. A. Petrick. *Separating Representation, Reasoning, and Implementation for Interaction Management : Lessons from Automated Planning*, pages 93–107. Springer Singapore, Singapore, 2017. 110

[FAM96] George Ferguson, James Allen, and Brad Miller. Trains-95 : Towards a mixed-initiative planning assistant. In *Proceedings of the Third International Conference on Artificial Intelligence Planning Systems*, AIPS'96, pages 70–77. AAAI Press, 1996. 58

[FBP+00] Raya Fidel, Harry Bruce, Annelise Mark Pejtersen, Susan Dumais, Jonathan Grudin, and Steven Poltrock. Collaborative information retrieval (cir). 1 :235–247, January 2000. 11

[FC04] Nicoletta Fornara and Marco Colombetti. A commitment-based approach to agent communication. *Applied Artificial Intelligence*, 18(9-10) :853–866, 2004. 67

[FFMM94] Tim Finin, Richard Fritzson, Don McKay, and Robin McEntire. Kqml as an agent communication language. In *Proceedings of the Third International Conference on Information and Knowledge Management*, CIKM '94, pages 456–463, New York, NY, USA, 1994. ACM. 51

[Fis01] Gerhard Fischer. User modeling in human–computer interaction. *User Modeling and User-Adapted Interaction*, 11(1) :65–86, Mar 2001. 1, 18, 26, 103, 131

[FMCB04] Raya Fidel, Annelise Mark Pejtersen, Bryan Cleal, and Harry Bruce. A multidimensional approach to the study of human-information interaction : A case study of collaborative information retrieval. *Journal of the American Society for Information Science and Technology*, 55(11) :939–953, 2004. 33, 34, 38

[FN71] Richard E. Fikes and Nils J. Nilsson. Strips : A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2(3) :189–208, 1971. 54

[Fos06] Jonathan Foster. Collaborative information seeking and retrieval. *Annual Review of Information Science and Technology*, 40(1) :329–356, 2006. 33

[FRA+17] Jörg Frommer, Dietmar Rösner, Rico Andrich, Rafael Friesen, Stephan Günther, Matthias Haase, and Julia Krüger. *LAST MINUTE : An Empirical Experiment in User-Companion Interaction and Its Evaluation*, pages 253–275. Springer International Publishing, Cham, 2017. 30

[FS09] Colum Foley and Alan F. Smeaton. Synchronous collaborative information retrieval : Techniques and evaluation. In Mohand Boughanem, Catherine Berrut, Josiane Mothe, and Chantal Soule-Dupuy, editors, *Advances in Information Retrieval*, pages 42–53, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. 38

[FS10] Colum Foley and Alan F. Smeaton. Division of labour and sharing of knowledge for synchronous collaborative information retrieval. *Inf. Process. Manage.*, 46(6) :762–772, November 2010. 35, 37

[FSL06] C. Foley, A. F. Smeaton, and H. Lee. Synchronous collaborative information retrieval with relevance feedback. In *2006 International Conference on Collaborative Computing : Networking, Applications and Worksharing*, pages 1–4, Nov 2006. 35

[FVC07] Nicoletta Fornara, Francesco Viganò, and Marco Colombetti. Agent communication and artificial institutions. *Autonomous Agents and Multi-Agent Systems*, 14(2) :121–142, Apr 2007. 67

[GB99] Alan K Galan and Albert D Baker. Multi-agent communication in jafmas. *Trans : Propose to Q3*, 5(S6) :S7, 1999. 59

[GDD12] Gene Golovchinsky, Anthony Dunnigan, and Abdigani Diriye. Designing a tool for exploratory information seeking. In *CHI '12 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '12, pages 1799–1804, New York, NY, USA, 2012. ACM. 39

[GGB+08] Carl Gutwin, Saul Greenberg, Roger Blum, Jeff Dyck, Kimberly Tee, and Gregor McEwan. Supporting informal collaboration in shared-workspace groupware. 14(9) :1411–1434, may 2008. |http ://www.jucs.org/jucs_14_9/supporting_informal_collaboration_in|. 38

[GHB00] Mark Greaves, Heather Holmback, and Jeffrey Bradshaw. *What Is a Conversation Policy ?*, pages 118–131. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000. 60, 61

[Gin94] Jonathan Ginzburg. An update semantics for dialogue. In *Proceedings of the first International Workshop on Computational Semantics*, 1994. 63

[Gin96] Jonathan Ginzburg. Interrogatives : Questions, facts and dialogue. *The handbook of contemporary semantic theory. Blackwell, Oxford*, pages 359–423, 1996. 63

[Gin12] Jonathan Ginzburg. *The interactive stance*. Oxford University Press, 2012. 63, 108

[GK96] Barbara J. Grosz and Sarit Kraus. Collaborative plans for complex group action. *Artif. Intell.*, 86(2) :269–357, October 1996. 24, 25, 55

[GNV06] Marc Grootjen, Mark A Neerincx, and JCM Van Weert. Task-based interpretation of operator state information for adaptive support. *Foundations of augmented cognition*, pages 236–242, 2006. 26

[GPB09] Gene Golovchinsky, Jeremy Pickens, and Maribeth Back. A taxonomy of collaboration in online information seeking. *CoRR*, abs/0908.0704, 2009. 33, 128

[GQPG09] G Golovchinsky, P Qvarfordt, J Pickens, and B Gray. Collaborative information seeking. *Collaborating Finding common ground for multiparty problems San Francisco JosseyBass*, 42(3), 2009. [Original String] :Golovchinsky, G., Qvarfordt, P., & Pickens, J. (2009). Collaborative information seeking. IEEE Computer, 42(3) Gray, B. (1989). Collaborating : Finding common ground for multiparty problems. San Francisco : Jossey-Bass. 33, 35, 37, 128

[Gra89] Barbara Gray. Collaborating : Finding common ground for multiparty problems. 1989. 11, 12, 13

[Gri69] H. P. Grice. Utterer's meaning and intention. *The Philosophical Review*, 78(2) :147–177, 1969. 56

[Gri75] H. Paul Grice. Logic and conversation. In Peter Cole, editor, *Speech acts*, volume 3 of *Syntax and semantics*, pages 41–58. Academic Press, New York, 1975. 45

[GS86] Barbara J Grosz and Candace L Sidner. Attention, intentions, and the structure of discourse. *Computational linguistics*, 12(3) :175–204, 1986. 55, 58, 108

[GS90] Barbara J. Grosz and Candace L. Sidner. *Plans for Discourse*, pages 417–444. MIT Press, 1990. 25, 55, 56

[GS11] Roberto González-Ibáñez and Chirag Shah. Coagmento : A system for supporting collaborative information seeking. *Proceedings of the American Society for Information Science and Technology*, 48(1) :1–4, 2011. 36, 39

[GSW12] Roberto González-Ibáñez, Chirag Shah, and Ryen W. White. Pseudo-collaboration as a method to perform selective algorithmic mediation in collaborative ir systems. *Proceedings of the American Society for Information Science and Technology*, 49(1) :1–4, 2012. 38

[GSW14] Roberto González-Ibáñez, Chirag Shah, and Ryen W. White. Capturing collabportunities : A method to evaluate collaboration opportunities in

information search using pseudocollaboration. *Journal of the Association for Information Science and Technology*, 66(9) :1897–1912, 2014. 40

[GTAP18] Lucian Galescu, Choh Man Teng, James Allen, and Ian Perera. Cogent : A generic dialogue system shell based on a collaborative problem solving model. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 400–409. Association for Computational Linguistics, 2018. 110, 111

[Ham70] C. L. Hamblin. *Fallacies*. Methuen, London, UK, 1970. 67

[Her08] Morten Hertzum. Collaborative information seeking : The combined activity of information seeking and collaborative grounding. *Information Processing & Management*, 44(2) :957–962, 2008. Evaluating Exploratory Search Systems Digital Libraries in the Context of Users' Broader Activities. 33

[HHH15] Jette Hyldegård, Morten Hertzum, and Preben Hansen. *Studying Collaborative Information Seeking : Experiences with Three Methods*, pages 17–35. Springer International Publishing, Cham, 2015. 33

[Hil18] Stefan Hillmann. *Simulation-Based Usability Evaluation of Spoken and Multimodal Dialogue Systems*. Springer International Publishing, 2018. 156

[HJ05] Preben Hansen and Kalervo Järvelin. Collaborative information retrieval in an information-intensive domain. *Inf. Process. Manage.*, 41(5) :1101–1119, September 2005. 33, 34

[Hoc01] Jean-Michel Hoc. Towards a cognitive approach to human-machine cooperation in dynamic situations. *International Journal of Human-Computer Studies*, 54(4) :509–540, 2001. 19, 21

[Hor99] Eric Horvitz. Uncertainty, action, and interaction : In pursuit of mixed-initiative computing. pages 17–20, September 1999. 19

[Hor07] Eric Horvitz. Reflections on challenges and promises of mixed-initiative interaction. *AI Magazine*, 28(2) :13–22, 2007. 19

[Hul00] Joris Hulstijn. Dialogue games are recipes for joint action. In *Proceedings of the Forth Workshop on the Semantics and Pragmatics of Dialogue (Gotalog'00)*, 2000. 51

[HW83] Erik Hollnagel and David D. Woods. Cognitive systems engineering : New wine in new bottles. *Int. J. Man-Mach. Stud.*, 18(6) :583–600, June 1983. 24, 27

[Iiv95] Mirja Iivonen. Searchers and searchers : Differences between the most and least consistent searches. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '95, pages 149–157, New York, NY, USA, 1995. ACM. 32

[JBF+14] Matthew Johnson, Jeffrey M. Bradshaw, Paul J. Feltovich, Catholijn M. Jonker, M. Birna van Riemsdijk, and Maarten Sierhuis. Coactive design : Designing support for interdependence in joint activity. *J. Hum.-Robot Interact.*, 3(1) :43–69, February 2014. 27, 28, 29

[JHJ08] Hideo Joho, David Hannah, and Joemon M. Jose. Comparing collaborative and independent search in a recall-oriented task. In *Proceedings of the Second International Symposium on Information Interaction in Context*, IIiX '08, pages 89–96, New York, NY, USA, 2008. ACM. 37, 38

[JMN+14] N. R. Jennings, L. Moreau, D. Nicholson, S. Ramchurn, S. Roberts, T. Rodden, and A. Rogers. Human-agent collectives. *Commun. ACM*, 57(12) :80–88, November 2014. 15

[Juc92] Andreas H Jucker. Conversation : structure or process. 1992. 45, 193

[KA86] Henry A Kautz and James F Allen. Generalized plan recognition. In *AAAI*, volume 86, page 5, 1986. 23

[Kay89] Alan Kay. User interface : A personal view. 1989. 10

[KBS17] Joseph Kim, Christopher J Banks, and Julie A Shah. Collaborative planning with encoding of users' high-level strategies. In *AAAI*, pages 955–962, 2017. 26

[KCS15] Been Kim, Caleb M. Chacha, and Julie A. Shah. Inferring team task plans from human meetings : A generative modeling approach with logic-based prior. *J. Artif. Int. Res.*, 52(1) :361–398, January 2015. 25

[Kib06] Rodger Kibble. Speech acts, commitment and multi-agent communication. *Computational & Mathematical Organization Theory*, 12(2) :127–145, Oct 2006. 66

[KP13] Ryan Kelly and Stephen Payne. Division of labour in collaborative information seeking :current approaches and future directions. In *The 3rd International Workshop on Collaborative Information Seeking, held at ACM CSCW 2013*, February 2013. 38

[KS04] Andruid Kerne and Steven M. Smith. The information discovery framework. In *Proceedings of the 5th Conference on Designing Interactive Systems : Processes, Practices, Methods, and Techniques*, DIS '04, pages 357–360, New York, NY, USA, 2004. ACM. 31

[LA90] Diane J. Litman and James F. Allen. *Discourse Processing and Commonsense Plans*, pages 365–388. MIT Press, 1990. 55

[Lar02] Staffan Larsson. *Issue-based dialogue management*. PhD thesis, 2002. 63, 64, 111

[LDC+12] Alain Loisel, Guillaume Dubuisson Duplessis, Nathalie Chaignaud, Jean-Philippe Kotowicz, and Alexandre Pauchet. A conversational agent for information retrieval based on a study of human dialogues. In *International Conference on Agent and Artificial Intelligence*, pages 312–317, Vilamoura, Portugal, February 2012. 111

[LDC+16] Jean-Baptiste Louvet, Guillaume Dubuisson Duplessis, Nathalie Chaignaud, Jean-Philippe Kotowicz, and Laurent Vercouter. Recherche collaborative de documents : comparaison assistance humaine/automatique. pages 161–166. Journées Francophones d'Ingénierie des Connaissances, 2016. 129

[LDC+17] Jean-Baptiste Louvet, Guillaume Dubuisson Duplessis, Nathalie Chaignaud, Laurent Vercouter, and Jean-Philippe Kotowicz. Modeling a collaborative task with social commitments. volume 112, pages 377–386, Amsterdam, The Netherlands, The Netherlands, September 2017. Elsevier Science Publishers B. V. 84, 129

[Lew69] D. Lewis. *Convention, A Philosophical Study.* Harvard University Press, Cambridge, MA, 1969. 62

[Lew79] David Lewis. Scorekeeping in a language game. *Journal of Philosophical Logic*, 8(1) :339–359, Jan 1979. 96, 97

[Lie95] Henry Lieberman. Letizia : An agent that assists web browsing. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95, Montréal Québec, Canada, August 20-25 1995, 2 Volumes*, pages 924–929, 1995. 17, 40

[Lie97] Henry Lieberman. Autonomous interface agents. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, CHI '97, pages 67–74, New York, NY, USA, 1997. ACM. 17, 132

[Lju09] Peter Ljunglöf. trindikit.py : An open-source python library for developing isu-based dialogue systems. In *IWSDS'09, 1st International Workshop on Spoken Dialogue Systems Technology*, 2009. 111

[LKC08] Alain Loisel, Jean-Philippe Kotowicz, and Nathalie Chaignaud. An issue-based approach to information search modelling : Analysis of a human dialog corpus. In Petr Sojka, Aleš Horák, Ivan Kopeček, and Karel Pala, editors, *Text, Speech and Dialogue*, pages 609–616, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. 128

[LLC+00] Staffan Larsson, Peter Ljunglöf, Robin Cooper, Elisabet Engdahl, and Stina Ericsson. Godis : an accommodating dialogue system. In *Proceedings of the 2000 ANLP/NAACL Workshop on Conversational systems-Volume 3*, pages 7–10. Association for Computational Linguistics, 2000. 64, 111, 128

[LMM98] Yezdi Lashkari, Max Metral, and Pattie Maes. Readings in agents. chapter Collaborative Interface Agents, pages 111–116. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998. 17

[LMMC17] Pat Langley, Ben Meadows, Sridharan Mohan, and Dongkyu Choi. Explainable agency for intelligent autonomous systems. In *AAAI*, pages 4762–4764, 2017. 22, 104

[Loc94] Karen Elizabeth Lochbaum. *Using Collaborative Plans to Model the Intentional Structure of Discourse.* PhD thesis, Cambridge, MA, USA, 1994. UMI Order No. GAX95-14804. 57

[Loc98] Karen E. Lochbaum. A collaborative planning model of intentional structure. *Comput. Linguist.*, 24(4) :525–572, December 1998. 57

[Loi08] Alain Loisel. *Modélisation du dialogue Homme-Machine pour la recherche d'informations : approche questions-réponses.* PhD thesis, INSA de Rouen, 2008. 75, 128

[Lon95] Scott London. *Collaboration and community*, volume 12. 1995. 12, 13

[LRS99] Neal Lesh, Charles Rich, and Candace L. Sidner. Using plan recognition in human-computer collaboration. In Judy Kay, editor, *UM99 User Modeling*, pages 23–32, Vienna, 1999. Springer Vienna. 25

[LT00] Staffan Larsson and David R. Traum. Information state and dialogue management in the trindi dialogue move engine toolkit. *Natural Language Engineering*, 6(3&4) :323–340, 2000. 49, 62, 64, 111

[Mae94] Pattie Maes. Agents that reduce work and information overload. *Commun. ACM*, 37(7) :30–40, July 1994. 17, 21

[Mar89] Gary Marchionini. Information-seeking strategies of novices using a full-text electronic encyclopedia. *Journal of the American Society for Information Science*, 40(1) :54, 1989. 31

[Mar95] Gary Marchionini. *Information Seeking in Electronic Environments*. Cambridge University Press, New York, NY, USA, 1995. 30

[Mau01] N. Maudet. *Modéliser l'aspect conventionnel des interactions langagières : la contribution des jeux de dialogue*. Thèse de doctorat en informatique, Université P. Sabatier, Toulouse, Mai 2001. 3, 67, 70, 77, 78, 191

[Mau02] Nicolas Maudet. A la recherche de la structure intentionnelle dans le dialogue. *Traitement automatique des langues*, 43(2) :71–98, 2002. 77, 78

[MCD02] N. Maudet and B. Chaib-Draa. Commitment-based and dialogue-game-based protocols : new trends in agent communication languages. *The Knowledge Engineering Review*, 17(2) :157–179, 2002. 65, 70, 193

[MH07] Meredith Ringel Morris and Eric Horvitz. Searchtogether : An interface for collaborative web search. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST '07, pages 3–12, New York, NY, USA, 2007. ACM. 36, 37, 39

[ML02] Daniel C. McFarlane and Kara A. Latorella. The scope and importance of human interruption in human-computer interaction design. *Hum.-Comput. Interact.*, 17(1) :1–61, March 2002. 21

[MLS18] Matthew Mitsui, Jiqun Liu, and Chirag Shah. Coagmento : Past, present, and future of an individual and collaborative information seeking platform. In *Proceedings of the 2018 Conference on Human Information Interaction & Retrieval*, CHIIR '18, pages 325–328, New York, NY, USA, 2018. ACM. 36

[MLW10] Meredith Ringel Morris, Jarrod Lombardo, and Daniel Wigdor. Wesearch : Supporting collaborative search and sensemaking on a tabletop display. *Proceedings of CSCW 2010*, pages 401–410, February 2010. 37

[Mor08] Meredith Ringel Morris. A survey of collaborative web search practices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1657–1660, New York, NY, USA, 2008. ACM. 34, 35

[MPW06] M. R. Morris, A. Paepcke, and T. Winograd. Teamsearch : comparing techniques for co-present collaborative search of digital media. In *First IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP '06)*, pages 8 pp.–, Jan 2006. 37

[MTP10] Meredith Ringel Morris, Jaime Teevan, and Katrina Panovich. What do people ask their social networks, and why ? : A survey study of status message q&a behavior. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 1739–1748, New York, NY, USA, 2010. ACM. 35

[MW07] Gary Marchionini and Ryen White. Find what you need, understand what you find. *Int. J. Hum. Comput. Interaction*, 23(3) :205–237, 2007. 31, 40, 130

[Neg70] Nicholas Negroponte. *The architecture machine.* MIT press, 1970. 10

[Nwa96] Hyacinth S. Nwana. Software agents : an overview. *The Knowledge Engineering Review*, 11(3) :205–244, 1996. 15, 17, 20, 59

[OJ93] Vicki L O'Day and Robin Jeffries. Orienteering in an information landscape : how information seekers get from here to there. In *Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems*, pages 438–445. ACM, 1993. 32

[ON98] P. D. O'Brien and R. C. Nicol. Fipa - towards a standard for software agents. *BT Technology Journal*, 16(3) :51–59, Jul 1998. 52

[OPB00] J. Odell, H.v.D. Parunak, and B. Bauer. Extending UML for agents. In *Proceedings of the Agent-Oriented Information Systems Workshop at the 17th National Conference on Artificial Intelligence*, 2000. 59

[Ort14] C. L. Ortiz. The road to natural conversational speech interfaces. *IEEE Internet Computing*, 18(2) :74–78, Mar 2014. 44

[OSV08] Martin Odersky, Lex Spoon, and Bill Venners. *Programming in scala.* Artima Inc, 2008. 121

[Pas04] Philippe Pasquier. Modèles des dialogues entre agents cognitifs : un état de l'art. *In Cognito-Cahiers Romans de Sciences Cognitives*, 1(4) :77–135, 2004. 53

[Pas05] Philippe Pasquier. *Aspects cognitifs des dialogues entre agents artificiels : l'approche par la cohérence cognitive.* Theses, Université Laval, June 2005. 44

[Pau06] Alexandre Pauchet. *Modélisation cognitive d'interactions humaines dans un cadre de planification multi-agents.* Theses, Université Paris-Nord - Paris XIII, September 2006. 59, 60, 191

[PGS+08] Jeremy Pickens, Gene Golovchinsky, Chirag Shah, Pernilla Qvarfordt, and Maribeth Back. Algorithmic mediation for collaborative exploratory search. In *Proceedings of the 31st Annual International ACM SIGIR Conference*

on Research and Development in Information Retrieval, SIGIR '08, pages 315–322, New York, NY, USA, 2008. ACM. 36, 37, 39

[PM09] Sharoda A. Paul and Meredith Ringel Morris. Cosense : Enhancing sensemaking for collaborative web search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 1771–1780, New York, NY, USA, 2009. ACM. 39

[Pol90] Martha E. Pollack. *Plans As Complex Mental Attitudes*, pages 77–103. MIT Press, 1990. 25

[PR97] Raja Parasuraman and Victor Riley. Humans and automation : Use, misuse, disuse, abuse. *Human Factors*, 39(2) :230–253, 1997. 20, 21, 22

[Pra13] Camille Pradel. *D'un langage de haut niveau à des requêtes graphes permettant d'interroger le web sémantique*. PhD thesis, 2013. Thèse de doctorat dirigée par Haemmerlé, Ollivier et Hernandez, Nathalie Intelligence artificielle Toulouse 3 2013. III

[Pul98] S. G. Pulman. The TRINDI project : Some preliminary themes. In Joris Hulstein and Anton Nijholt, editors, *TWLT 13, Formal Semantics and Pragmatics of Dialogue, University of Twente, Enschede.*, pages 31–39, 1998. 78

[PW10] Julia Peltason and Britta Wrede. Pamini : A framework for assembling mixed-initiative human-robot interaction from generic interaction patterns. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, SIGDIAL '10, pages 229–232, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. 111

[RFW+17] Dietmar Rösner, Jörg Frommer, Andreas Wendemuth, Thomas Bauer, Stephan Günther, Matthias Haase, and Ingo Siegert. *The LAST MINUTE Corpus as a Research Resource : From Signal Processing to Behavioral Analyses in User-Companion Interactions*, pages 277–299. Springer International Publishing, Cham, 2017. 30

[RG91] A. S. Rao and M. P. Georgeff. Modeling rational agents within a bdi-architecture. In *Principles of Knowledge Representation and Reasoning. Proceedings of the second International Conference*, pages 473–484, San Mateo, 1991. Morgan Kaufmann. 53

[RJ08] Madhu C. Reddy and Bernard J. Jansen. A model for understanding collaborative information behavior in context : A study of two healthcare teams. *Inf. Process. Manage.*, 44(1) :256–273, January 2008. 33, 34

[Rob00] Nancy Roberts. Wicked problems and network approaches to resolution. *International public management review*, 1(1) :1–19, 2000. 11, 12, 31, 34

[Rob15] Craige Roberts. *Accommodation in a Language Game*, chapter 22, pages 345–366. John Wiley & Sons, Ltd, 2015. 97

[RS06] Charles Rich and Candace L. Sidner. From the programmer's apprentice to human-robot interaction : Thirty years of research on human-computer

collaboration. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2*, AAAI'06, pages 1530–1533. AAAI Press, 2006. 24

[RS07] Charles Rich and Candace L Sidner. Diamondhelp : A generic collaborative task guidance system. *AI Magazine*, 28(2) :33, 2007. 57

[RS12] Charles Rich and Candace L. Sidner. Using collaborative discourse theory to partially automate dialogue tree authoring. In Yukiko Nakano, Michael Neff, Ana Paiva, and Marilyn Walker, editors, *Intelligent Virtual Agents*, pages 327–340, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. 57, 110

[RSL01] Charles Rich, Candace L. Sidner, and Neal Lesh. Collagen : Applying collaborative discourse theory to human-computer interaction. *AI Mag.*, 22(4) :15–25, October 2001. 24, 55, 57, 110

[Sad91] M David Sadek. Dialogue acts are rational plans. In *The Structure of Multimodal Dialogue ; Second VENACO Workshop*, 1991. 53

[SDN12] Georg Singer, Dmitri Danilov, and Ulrich Norbisrath. Complex search : Aggregation, discovery, and synthesis. *Proceedings of the Estonian Academy of Sciences*, 61(2) :89–106, 2012. 31

[SE98] Alistair Sutcliffe and Mark Ennis. Towards a cognitive theory of information retrieval. *Interacting with computers*, 10 :321–351, 1998. 31, 130

[Sea69] John R Searle. *Speech acts : An essay in the philosophy of language*. Cambridge university press, 1969. 47, 54

[SGI11] Chirag Shah and Roberto González-Ibáñez. Evaluating the synergic effect of collaboration in information seeking. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 913–922, New York, NY, USA, 2011. ACM. 11, 37

[Sha10] Chirag Shah. *A Framework for Supporting User-centric Collaborative Information Seeking*. PhD thesis, Chapel Hill, NC, USA, 2010. AAI3418610. 13

[Sha12] Chirag Shah. *Collaborative information seeking : The art and science of making the whole greater than the sum of all*, volume 34. Springer Science & Business Media, 2012. 11

[Sha14] Chirag Shah. Collaborative information seeking. *Journal of the Association for Information Science and Technology*, 65(2) :215–236, 2014. 12, 34

[Sha15] Chirag Shah. *Collaborative Information Seeking : From 'What ?' and 'Why ?' to 'How ?' and 'So What ?'*. Springer, 2015. 13

[SHGI15] Chirag Shah, Chathra Hendahewa, and Roberto González-Ibáñez. Two's company, but three's no crowd : Evaluating exploratory web search for individuals and teams. *Aslib Journal of Information Management*, 67(6) :636–662, 2015. 37

[Shn83] Shneiderman. Direct manipulation : A step beyond programming languages. *Computer*, 16(8) :57–69, aug 1983. 16

[SHW07] K. Stubbs, P. J. Hinds, and D. Wettergreen. Autonomy and common ground in human-robot interaction : A field study. *IEEE Intelligent Systems*, 22(2) :42–50, March 2007. 22

[Sin91] M. P. Singh. Social and psychological commitments in multiagent systems. In *AAAI Fall Symposium on Knowledge and Action at Social and Organizational Levels*, pages 104–106, 1991. 64

[Sin98] M. P. Singh. Agent communication languages : rethinking the principles. *Computer*, 31(12) :40–47, Dec 1998. 53, 61, 64

[Sin99] Munindar P. Singh. An ontology for commitments in multiagent systems. *Artif. Intell. Law*, 7(1) :97–113, March 1999. 65

[Sin00] Munindar P. Singh. *A Social Semantics for Agent Communication Languages*, pages 31–45. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000. 65, 68

[SLFM07] Alan F. Smeaton, Hyowon Lee, Colum Foley, and Sinéad McGivney. Collaborative video searching on a tabletop. *Multimedia Systems*, 12(4) :375–391, Mar 2007. 37

[SM93] William R. Swartout and Johanna D. Moore. Explanation in second generation expert systems. In Jean-Marc David, Jean-Paul Krivine, and Reid Simmons, editors, *Second Generation Expert Systems*, pages 543–585, Berlin, Heidelberg, 1993. Springer Berlin Heidelberg. 22

[SMK09] Chirag Shah, Gary Marchionini, and Diane Kelly. Learning design principles for a collaborative information seeking system. In *CHI '09 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '09, pages 3419–3424, New York, NY, USA, 2009. ACM. 36

[Sou14] Laure Soulier. *Définition et évaluation de modèles de recherche d'information collaborative basés sur les compétences de domaine et les rôles des utilisateurs.* PhD thesis, Université de Toulouse, 2014. 36

[SPG10] Chirag Shah, Jeremy Pickens, and Gene Golovchinsky. Role-based results redistribution for collaborative information retrieval. *Information Processing & Management*, 46(6) :773–781, 2010. Collaborative Information Seeking. 36, 39

[SRH05] Patricia Ruma Spence, Madhu C. Reddy, and Richard Hall. A survey of collaborative information seeking practices of academic researchers. In *Proceedings of the 2005 International ACM SIGGROUP Conference on Supporting Group Work*, GROUP '05, pages 85–88, New York, NY, USA, 2005. ACM. 34

[ST17] Laure Soulier and Lynda Tamine. On the collaboration support in information retrieval. *ACM Comput. Surv.*, 50(4) :51 :1–51 :34, August 2017. 34

[STS16] Laure Soulier, Lynda Tamine, and Chirag Shah. Minerank : Leveraging users' latent roles for unsupervised collaborative information retrieval. *Information Processing & Management*, 52(6) :1122–1141, 2016. 36

[Suc07] Lucy A. Suchman. *Human-Machine Reconfigurations : Plans and Situated Actions*. Cambridge University Press, Cambridge, 2 edition, 2007. 24, 78, 85, 103, 131

[Sur05] James Surowiecki. *The wisdom of crowds*. Anchor, 2005. 11

[SV85] John R. Searle and Daniel Vanderveken. *Foundations of Illocutionary Logic*. Cambridge University Press, 1985. 48

[TA92] David R. Traum and Hinkelman Elizabeth A. Conversation acts in task-oriented spoken dialogue. *Computational Intelligence*, 8(3) :575–599, 1992. 77

[TA94] David R. Traum and James F. Allen. Discourse obligations in dialogue processing. In *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics*, ACL '94, pages 1–8, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics. 58

[Tam97] Milind Tambe. Agent architectures for flexible, practical teamwork. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Conference on Innovative Applications of Artificial Intelligence*, AAAI'97/IAAI'97, pages 22–28. AAAI Press, 1997. 24

[Ter95] Loren G. Terveen. Overview of human-computer collaboration. *Knowledge-Based Systems*, 8(2) :67–81, 1995. Human-computer collaboration. 14, 18, 103, 131

[TL03] David R. Traum and Staffan Larsson. *The Information State Approach to Dialogue Management*, pages 325–353. Springer Netherlands, Dordrecht, 2003. 49, 62, 113, 115

[TLFa] Définition de plan. `http://www.cnrtl.fr/definition/plan//2`. Consulté le 27/06/2018. 23

[TLFb] Définition de tâche. `http://www.cnrtl.fr/definition/t%C3%A2che`. Consulté le 27/06/2018. 45

[TPRG98] E Taylor-Powell, B Rossing, and J Geran. *Evaluating collaboratives : Reaching the potential*. 1998. 11, 12, 13, 14, 100, 193

[UYS17] Vaibhav V Unhelkar, X Jessie Yang, and Julie A Shah. Challenges for communication decision-making in sequential human-robot collaborative tasks. 2017. 21

[Van90] Daniel Vanderveken. *Meaning and speech acts : principles of language use*. Cambridge University Press, New York, NY, USA, 1990. 48

[VCd00] Laurent Vongkasem and Brahim Chaib-draa. *ACL as a Joint Project between Participants : A Preliminary Report*, pages 235–248. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000. 45

[Wag94] Ellen D. Wagner. In support of a functional definition of interaction. *American Journal of Distance Education*, 8(2) :6–29, 1994. 11

[WF87] Terry Winograd and Fernando Flores. *Understanding Computers and Cognition : A New Foundation for Design*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1987. 61

[WG91] Donna J. Wood and Barbara Gray. Toward a comprehensive theory of collaboration. *The Journal of Applied Behavioral Science*, 27(2) :139–162, 1991. 11, 13

[WJ95] Michael Wooldridge and Nicholas R. Jennings. Intelligent agents : theory and practice. *The Knowledge Engineering Review*, 10(2) :115–152, 1995. 15

[Woo02] Michael Wooldridge. *An introduction to multiagent systems*. John Wiley & Sons, 2002. 15, 52

[YHHJ14] Zhen Yue, Shuguang Han, Daqing He, and Jiepu Jiang. Influences on query reformulation in collaborative web search. *Computer*, 47(3) :46–53, March 2014. 37