République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mentouri de Constantine
Faculté des Sciences de l'Ingénieur
Département d'Informatique

N° d'ordre : 398/Mag/2008
Série : 013/Inf/2008

**Mémoire présenté en vue de l'obtention du**

**Magister en Informatique**

**Option : Systèmes d'Information & Intelligence Artificielle Distribués**

---

# Vers une Architecture d'un Système de Dialogue Multi Agents Basé sur l'Argumentation

## *Application à la négociation dans le domaine de e-commerce*

---

**Présenté par :** Mr Hanine Sahki **Dirigé par** : Pr. Z. Sahnoun

**Soutenu le samedi 6/12/2008, 10.30h au Audio-visuel (chabet erssas)**

**Devant le jury composé de :**

| | | |
|---|---|---|
| *Président :* | Mme Z.Boufaida | Professeur Université de Constantine |
| *Rapporteur :* | Mr M.Boufaida | Professeur Université de Constantine |
| *Examinateurs :* | Mr O.Kazar | Maître de conférence Université de Biskra |
| | Mr R.Maamri | Maître de conférence Université de Constantine |

تلقى نظم الحوار متعدد العميل في الوقت الحاضر اهتماما متزايدا ، ونظرا لظهور شبكة الإنترنت وتطبيقات التجارة الإلكترونية ، نوع معين من الحوار أكتسب الكثير من الاهتمام على نطاق واسع لبناء نظم متعددة الوكيل و هو التفاوض .

من جهة ، عدة تطبيقات لنظم الحوار تظهر هيمنة نوع معين من الحوار وهو الإقناع .

وفي جهة أخرى ، فإن معظم النظم المقترحة تدعم تثبيت بروتوكول تفاعل واحد في الوكيل. ولذلك ، هذه النظم هي مغلقة ، وهذا غير مرغوب فيه خاصة مع ظهور تكنولوجيا الشبكة الدلالية وتبويب المعرفة .

في هذه مذاكرة نقترح هيكل لنظام حوار متعدد العميل للمفاوضات بين وكلين استنادا إلى الحجج ،

يشمل هذا النظام على مبوبة المعرفة لبروتوكولات تدعم بروتوكولات متعددة للتفاوض .و أخيرا هذا النظام سوف يطبق في حالة التجارة الإلكترونية .

**الكلمات الرئيسية :** نظام متعدد الوكلاء ، ونظام الحوار والتفاوض ، والدلالية على الشبكة العالمية ، والحجج ، وتبويب المعرفة

*

# Abstract

The dialog systems multi agents cause an increasing interest nowadays, and seen the emergence of the Internet and the electronic applications of trade, a particular type of dialogue that wins a lot of attention and considered extensively as key to construct several systems multi agents are the negotiation.

Of a leaves, several implementations of the dialog systems watch a predominance of the persuasion dialogue.

On the other hand, most proposed systems support only one interaction protocols fixed in the agent. By consequence, these systems are closed; a property especially wished with the apparition of the technology of web semantics and ontology.

In this invoice we propose architecture of a multi agent dialog system for the automatic negotiation enters two agents based on the argumentation,

This system integrates ontology of the protocols for supports several protocols of negotiation. This system will be wall lamp in a case of electronic trade.

**Keywords**: System Multi agents, Dialog system, the Negotiation, Semantic Web, The arguing, the Ontology.

# Résume

Les systèmes de dialogue multi agents suscitent de nos jours un intérêt croissant, et vu l'émergence de l'Internet et les applications de commerce électroniques, un type particulier de dialogue qui gagne beaucoup d'attention et considéré largement comme clé pour construire des systèmes multi agents est la négociation.

D'un part, plusieurs implémentations des systèmes de dialogue **montre** une prédominance de dialogue de type persuasion.

D'autre part, la plupart des systèmes proposés supportent un seul protocole d'interaction fixé dans l'agent. Par conséquence, ces systèmes sont fermés; une propriété non souhaitée surtout avec l'apparition de la technologie de web sémantiques et les ontologies.

Dans ce mémoire nous proposons une architecture d'un système de dialogue multi agent pour la négociation automatique entre deux agents basé sur l'argumentation,

Ce système intègre une ontologie des protocoles pour supporte plusieurs protocoles de négociation. Ce système sera applique dans un cas de commerce électronique.

**Mots clés**: Système Multi agents, Système de Dialogue, la Négociation, Web Sémantique, L'Argumentation, les Ontologies.

# *Remerciements*

Je remercie très sincèrement Monsieur Sahnoun Zaidi, Professeur à l'Université Mentouri de Constantine, de m'avoir proposé ce sujet de mémoire et de m'avoir encadré. Je le remercie également pour son ouverture d'esprit qui m'a conduit à tirer le meilleur parti de mes capacités intellectuelles. Son aide tant scientifique que personnelle, ainsi que ses conseils ont soutenu mon travail de recherche. Ses connaissances et son expérience dont j'ai bénéficié m'ont grandement aidé.

Aussi, je tiens à remercier Monsieur M.Boufaïda Professeur à l'Université Mentouri de Constantine, d'être mon rapporteur.

Je tiens également à remercier les membres du jury qui m'ont fait l'honneur de bien vouloir évoluer mon travail, et plus précisément :

Madame Z.Boufaïda, Professeur à l'Université Mentouri de Constantine, pour l'honneur qu'elle me fait, en acceptant la présidence de ce jury malgré les aléas d'un calendrier très chargé.

Monsieur Mr O.Kazar, Maître de Conférence à l'université de Biskra et Monsieur Mr R.Maamri, Maître de Conférence à l'université Mentouri de Constantine auxquels je suis très reconnaissant d'avoir accepté d'être examinateurs de ce travail.

# Sommaire

# Liste des figures

# Introduction générale

## Contexte de travail

Les systèmes informatiques sont devenus de plus en plus omniprésents, ils ont évolué à partir d'entités monolithiques isolées, vers de grands systèmes décentralisés et interconnectés.

Par conséquent, la technologie des logiciels a connu une transition; de systèmes monolithiques conçus pour fonctionner sur une seule plateforme, vers un ensemble de sous-systèmes (semi autonome, hétérogènes et ad hoc) conçus d'une manière indépendante.

Ceci a amené à la création d'une nouvelle classification de composants logiciels intitulée "agents logiciels autonomes". Ces agents opérants de la part d'un utilisateur et ils sont souvent imprégnés avec quelques formes d'intelligence artificielle qui permet : la prise de décision, la panification et l'action d'une manière autonome.

En parallèle, avec l'émergence de paradigme de web sémantique [1], une demande croissante pour des programmes qui peuvent inter-opérer pour échanger des informations et des services avec d'autres programmes pour résoudre des problèmes complexes.

L'application de la technologie d'agent a trouvé un grand succès dans des domaines différents. Dans la plupart des ces applications, les agents doivent être capables d'interagir avec les autres agents à cause de l'interdépendance inhérente qui existe entre eux, ils ont besoin de communiquer pour résoudre des différences d'opinions et de conflits d'intérêts, etc.

Ces exigences de communication ne peuvent pas être accomplies par l'échange d'un seul message, mais l'agent concerné doit être capable d'échanger des séquences de messages. En d'autres termes ils ont besoin de la capacité de dialogue :" Pas simplement en échangeant des données, mais en prenant part à analogues du genre d'activité sociale que nous engageons tous les jours dans notre vie : Coopération, Coordination, Négociation." [2].

Un système de dialogue est un dispositif à travers lequel un ensemble de participants communiquent en respectant certaines règles, les systèmes de dialogue définissent essentiellement le principe d'un dialogue cohérent et les conditions sous lesquelles une locution (ou acte de langage) fait par un agent est appropriée. Il spécifie aussi quand un

agent est autorisé à envoyer une locution pour achever le désiré ou le but du dialogue. Dans la littérature, on distingue plusieurs types de dialogue [3].

Vu l'émergence de l'Internet et les applications de commerce électroniques, un type particulier de dialogue qui gagne beaucoup d'attention et considéré largement comme clé pour construire des systèmes multi-agents pour la négociation.

La négociation fournit un moyen aux agents pour résoudre leurs conflits d'intérêts, en leur permettant de trouver un accord mutuellement acceptable.

Cependant, le dialogue de la négociation entre participants est typiquement plus riche et plus complexe qu'un simple échange d'offres et contre offres.

En plus, les agents dans les systèmes proposés sont souvent faces à des informations incomplètes ou incohérentes.

La technologie courante de l'agent, utilise des formes classiques formalisées par la logique pour supporter la communication et le processus de prise de décision et par conséquent les agents sont incapables de réagir efficacement avec les situations mentionnées précédemment.

L'argumentation [4] est relativement un nouveau paradigme dans l'intelligence artificielle, basée sur un framework flexible et riche. Les techniques et les résultats de l'argumentation ont trouvent une grande gamme d'applications dans les deux branches théorique et pratique de l'intelligence artificielle [5] [6].

Récemment, l'argumentation a été utilisée pour formaliser les systèmes de dialogue, par conséquent plusieurs systèmes ont été construits (voir chapitre 3).

## Problématique et motivations

Au cours des dernières années, un nombre croissant de systèmes de dialogue multi-agents basés sur l'argumentation ont été élaborés et présentés pour différents types de disciplines.

Parmi les systèmes les plus répondu, on trouve TRAINS [7], PARMA [8], HOMEY [9], Artikis [10], etc.

La plupart de ces systèmes sont centralisés dans le sens qu'ils adoptent un agent médiateur, ce qui peut engendrer un goulet d'étranglement.

En plus, les systèmes de dialogue conçu jusqu'à présent montrent une prédominance d'un type particulier de dialogue qui est la persuasion.

Mais pour les applications dans le commerce électronique, l'attention doit être donnée pour supporter d'autres types de dialogue mentionnés dans la topologie de Walton et Krabbe [3] et plus spécifiquement la négociation.

D'autre part, ces systèmes intègrent un seul protocole de dialogue, que ce soit la négociation ou autre, et le protocole est codé implicitement, fixé dans l'agent et connue à l'avance. Ces types de systèmes sont fermés dans le sens que le protocole et la stratégie sont prédéterminés.

Mais les avances en technologie de web sémantique et de commerce électronique, ont poussé vers des systèmes flexibles et dynamiques qui sont multi protocoles.

## Objectifs du travail

Étant donné ces exigences, nous proposons un système de dialogue multi agent basé sur l'argumentation pour la négociation automatique décentralisée qui supporte plusieurs protocoles de négociation entre deux agents.

Les agents impliqués dans la négociation ne doivent pas forcément supporter un seul protocole de négociation, mais ils sont capables de choisir le protocole de négociation approprie au type d'interaction où ils participent.

Donc, dans notre système le protocole de la négociation n'est pas codé implicitement dans l'agent, mais les agents choisissent le type de protocole qui règle leurs interactions et qui est décrit sous forme d'une ontologie partagée; c'est-à-dire une représentation explicite et explicative du protocole de négociation. Par conséquent, la description du protocole est maintenant acquise de l'ontologie du système.

L'avantage est que les agents n'ont plus besoin d'aller hors-ligne pour reprogrammer afin de supporter d'autres protocoles de négociation. Cette approche est convenable particulièrement pour des applications comme le commerce électronique et qui rend notre système générique et ouvert.

Pour concrétiser notre système nous concevons une architecture qui supporte ce système dont lequel un tel dialogue peut avoir lieu. Ce système s'intéresse à l'automatisation de la négociation entre agents interagissant dans le cadre du commerce électronique.

En plus, l'architecture de notre système est également a pour but de fournir un ensemble de composants génériques pour intégrer différents types de protocoles.

Donc, notre contribution dans ce contexte consiste à :

- La construction d'une architecture d'un système de dialogue multi-agents basé sur l'argumentation pour la négociation automatique entre deux agents.
- L'intégration d'une ontologie de protocoles de négociation argumentative dans notre système de dialogue. Cette ontologie joue le rôle d'un fournisseur de protocoles de négociation.
- L'instanciation de l'architecture de système dans le cadre du commerce électronique, avec une étude de cas qui comprenait un scénario d'une agence de voyages.

## Organisation du mémoire

Pour présenter le travail, nous avons élaboré le plan de lecture suivant :

Le premier chapitre a pour but de familiariser le lecteur avec les principaux concepts et technologies et survole le domaine des systèmes multi-agents.

Premièrement, il donne un aperçu général sur l'agent, ses caractéristiques et leurs différents types. Deuxièmement, nous verrons les systèmes multi-agents, leurs domaines d'applications et leurs propriétés à savoir l'interaction, l'organisation et les mécanismes misent en oeuvre pour la communication entre agents; à savoir les actes de parole et les langages de communication.

Le deuxième chapitre est consacré à la présentation des protocoles d'interaction et leurs spécifications par les ontologies, nous commençons par la définition de la notion de protocole d'interaction et ses différents types. Ensuite nous présentons les ontologies, leurs caractéristiques, leurs domaines d'application. Nous présentons les outils nécessaires de leur développement à savoir, les langages de représentation, les outils d'édition et d'interrogation. Enfin, nous discutons la représentation des protocoles d'interaction et plus spécifiquement les protocoles de négociation par les ontologies.

Le troisième chapitre est dédié à la présentation de systèmes de dialogue. Nous commençons par la définition de la notion du système de dialogue, les différents types de dialogue, les principales approches utilisées pour leurs constructions, tout en focalisant sur l'argumentation. Après, nous présentons un type particulier de dialogue qui est la négociation. Finalement, nous présentons quelques implémentations.

Le quatrième chapitre est consacré à une présentation détaillée de notre travail, dans lequel nous proposons une architecture basée agents pour la négociation, les agents et leurs

structures internes, l'ontologie de protocole de négociation. Nous terminons ce chapitre par une étude de cas dans le domaine de commerce électronique.

Le dernier chapitre a pour but la validation de l'architecture proposée par une simulation faite dans l'environnement JADE.

Le mémoire s'achève par une conclusion générale récapitulant le contexte de recherche de notre étude, la démarche suivie, nos contributions et un ensemble de perspectives.

# *Chapitre I*
# *Les agents & les systèmes multi-agents*

## 1. Introduction

L'ingénierie informatique est progressée ces dernières années à travers l'orientation vers une abstraction de plus en plus élevée et qui nous permettre de modéliser des systèmes plus complexes, ceci a conduit à l'émergence d'un nouveau paradigme informatique à savoir les systèmes multi-agents.

Les systèmes multi-agents sont devenus une nouvelle approche d'abstraction qui peut être utilisée pour l'analyse, la modélisation et le développement des systèmes informatiques complexes et distribués.

La vision basée sur l'agent offre un puissant répertoire d'outils, de techniques, et de métaphores qui y ont le potentiel d'améliorer considérablement les systèmes logiciels.

On peut définir les systèmes multi-agents comme un ensemble d'agents autonomes faiblement couplés, des composants logiciels qui exploitent des différentes techniques d'intelligence artificielle.

Les systèmes multi-agents sont à l'intersection de plusieurs domaines scientifiques : L'informatique répartie, le génie logiciel, l'intelligence artificielle, la vie artificielle. Ils inspirent également d'études issues d'autres disciplines connexes notamment la sociologie, la psychologie sociale, les sciences cognitives, etc.

Ce chapitre a l'intention de familiariser le lecteur avec les concepts clés et d'illustrer l'état de l'art concernant l'agent et les systèmes multi-agents.

Le chapitre est divisé en deux grands titres, premièrement, une présentation de concept d'agent et leurs caractéristiques à savoir l'autonomie, l'adaptabilité et la sociabilité. En second lieu, nous donnons les différents types d'agents.

Deuxièmement, nous verrons les systèmes multi-agents, leurs domaines d'applications et leurs propriétés à savoir l'interaction et l'organisation et le mécanisme mis en oeuvre pour la communication entre agents.

## 2. La notion d'agent

Le concept d'agent a été l'objet d'études pour plusieurs décennies dans différentes disciplines : Les systèmes à base de connaissances, la robotique et d'autres domaines de l'intelligence artificielle.

Mais aussi dans des disciplines comme la philosophie et la psychologie et pour cella, on trouve une multitude de définitions du concept agent selon le type d'application pour laquelle l'agent est conçu.

Ferber [11] dit : « Un agent est une entité autonome, réelle ou abstraite, qui est capable d'agir sur lui-même et sur son environnement, qui, dans un univers multi agent, peut communiquer avec d'autres agents, et dont le comportement est la conséquence de ses observations, de ses connaissances et des interactions avec les autres agents ».

Sycara [12] défini un agent, comme un système informatique dont les caractéristiques principales sont les suivantes :

- Situé : l'agent est capable d'agir sur son environnement à partir des entrées sensorielles qu'il reçoit de ce même environnement.
- Autonomie : l'agent peut agir sans intervention directe d'un tiers (par les êtres humains ou les autres agents) et contrôle ses propres actions ainsi que son état interne.
- Adaptabilité : l'agent est capable de
    1. réagir d'une manière flexible aux changements dans son environnement.
    2. Prenant des initiatives dirigées par les buts au bon moment.
    3. Apprendre de son propre expérience, son environnement, et avec interactions avec les autres agents.
- la sociabilité : Un agent est capable d'interagir avec les autres agents (agents ou humains) quand la situation l'exige afin de compléter ses taches ou aider ces agents à accomplir les leurs.

Bien entendu, dépendamment des applications certaines propriétés sont plus importantes que les autres, il peut même s'avérer que pour certains types d'applications des proportions additionnelles soient requises.

## 2.1 Types d'agent

Les différences qui existent entre les définitions de l'agent ont mené vers des différences dans l'architecture interne de l'agent.

Ces architectures peuvent être réparties entre des architectures purement réactives, des architectures cognitives basées sur les désirs de la croyance et les intensions (modèle BDI) et architecture hybrides.

### 2.1.1 Agent réactif

Possède l'architecture interne la plus simple, qui correspond à la caractéristique purement réactive d'agents. Il ne possède pas modèle interne complexe explicite de son environnement, et leur comportement est primitif et ne consiste en général qu'à répondre à des stimuli extérieurs.

Il perçoit son environnement et répond à la situation actuelle d'une manière réactive sans aucun type de raisonner.

Un système réactif est constitué d'un nombre considérable d'agents de faibles granularités.

Leur comportement compte sur deux fonctions séquentielles la perception de changement de l'état de l'environnement sur leur état interne et l'action sur l'environnement.

Les architectures purement réactives sont simples et rapides, mais ils ne peuvent pas être proactifs et ils ne peuvent pas apprendre à partir d'expérience parce qu'ils n'ont non aucun état interne. Donc dès qu'un comportement exige quelque mémoire ou abstraction de l'état interne de perceptions, la réactivité pure doit être atténuée.

Parmi les travaux appartenant à cette catégorie, nous citons : les robots de brooks et l'eco-résolution de Ferber et Dorgoul [11]. Donc, l'école réactive considère que l'intelligence émerge des interactions d'agents non intelligents.

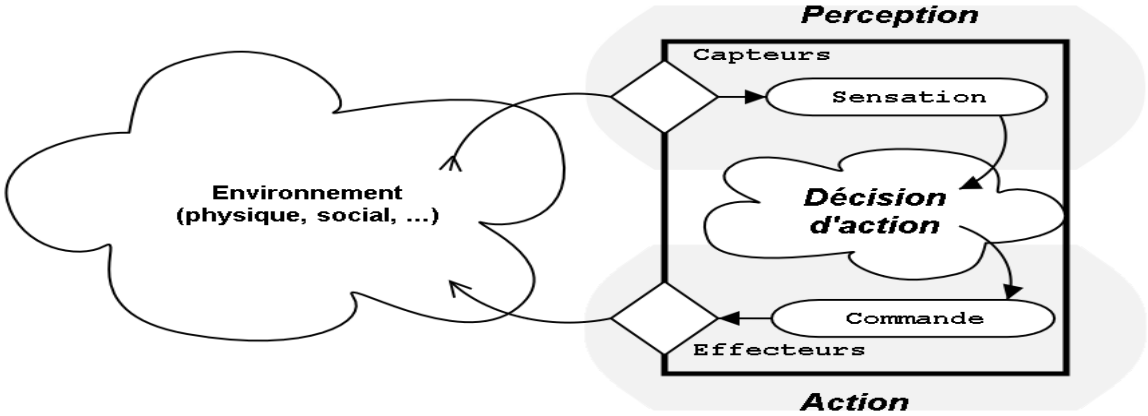


**Figure 1.1 : Une architecture d'un agent réactif [2].**

### 2.1.2 Agent cognitif

Dans le cadre du raisonnement pratique, un raisonnement orienté vers la prise en compte des états mentaux, on trouve l'architecture BDI [13] [14] (de l'anglais believe, desire, intention pour croyances, désir et intention).

L'architecture interne la plus complexe jusqu'aujourd'hui (figure 2) se caractérise par une représentation explicite de l'environnement, et possède un état mental ayant les attitudes mentales suivantes :

1. Les croyances : Ce que l'agent maintient au sujet de son environnement.
2. Les désirs : Les états possibles envers lesquels l'agent peut vouloir s'engagent et qui motive son comportement.
3. Les intentions : les états envers lesquels l'agent s'est engagé.

Son comportement est la conséquence de ses observations et des croyances qu'il a sur les autres agents et il est contrôle par fonctions du haut niveau par rapport à l'architecture réactive.

Un agent BDI doit donc mettre à jour ses croyances avec les informations qui lui proviennent de son environnement via la fonction perception, décider quelles options lui sont offertes, filtre ces options afin de déterminer des nouvelles intentions motivées par les croyances et les désires courantes et poser ses actions au vu de ses intentions sur l'envenimement externe.

Bien sûr cette architecture puissante a pour coût la difficulté de concevoir et implanter un comportement BDI complet.

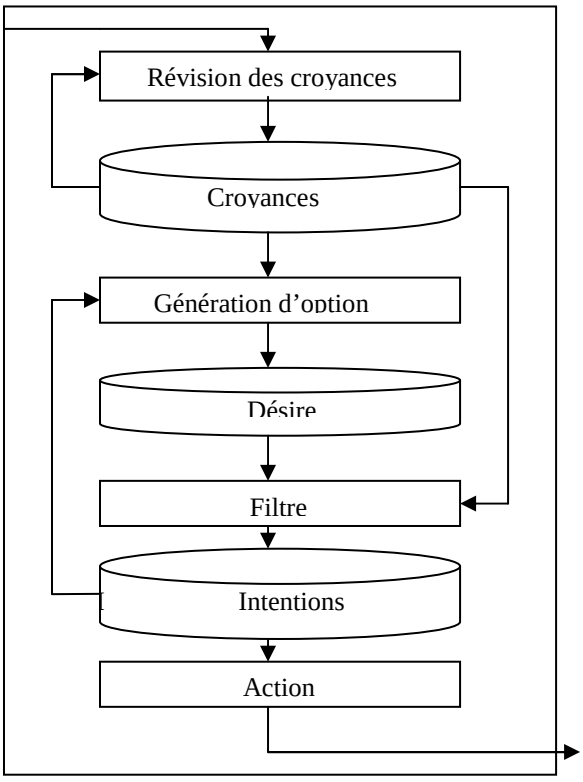


**Figure1.2 : Une architecture générique d'un agent BDI [15].**

### 2.1.3. Agent hybride

Les systèmes réactifs pouvaient bien convenir pour certains types de problèmes et moins bien pour d'autres et cela est valable pour les systèmes cognitifs.
Chaib [16][17][18][19], Fischer [20] et Georgef [21] ont investigué la possibilité de combiner les deux approches afin d'obtenir des architectures hybrides.
Une architecture d'agent hybride est un agent composer de plusieurs couches, arrangées selon une hiérarchie, et qui combine et intègre des caractéristiques faisant parties des deux types d'agents. Par exemples, on peut construire des systèmes réactifs avec des agents intelligents ou l'inverse (avoir des systèmes cognitifs dont les agents possèdent un caractère réactif aux stimuli).
On trouve plusieurs architectures basées un modèle de trois couches : Couche réactive, couches cognitives et une couche de communication entre agents.

## 2. Système multi-agents

Un système avec un seul agent est de l'intelligence artificielle classique ; un système avec les agents multiples est une société artificielle. Donc, les thèmes principaux et les fondations de l'intelligence artificielle distribuée sont l'organisation, la coordination et la coopération [22].
Aussi, on étend la définition de résolution distribuée de problèmes [23], un système multi agent est défini comme un réseau faiblement couplé d'agents qui travaillent ensemble comme une société pour résoudre des problèmes qui seraient généralement au-delà la portée de tout agent seul.
D'après Sycara [24], un SMA possède les caractéristiques suivantes :

- chaque agent a de l'information incomplète ou des capacités pour résoudre le problème total saisir par le système et, donc, à un point de vue limité.
- Il n'y a aucun contrôle global du système : Le comportement collectif est le résultat de règles sociales et interactions et pas d'un surveillant d'autorité centrale.
- Les ressources sont décentralisées : Les ressources nécessaires pour l'achèvement des tâches assigné au système sont divisées et sont distribuées.

### 2.1 Champs d'application des systèmes multi-agents

Les SMA sont des systèmes idéaux pour représenter des problèmes possédant de multiples méthodes de résolution, de multiples perspectifs et/ou de multiples résolveurs. Un système

multi agent possède les avantages de la résolution distribuée et concurrente de problèmes comme la modularité, la vitesse (via le parallélisme), et la fiabilité (due à la redondance). Ils héritent aussi des bénéfices envisageables d'IA comme le traitement symbolique, la facilite de maintenance, la réutilisation et la portabilité.

Les domaines d'application de la technologie d'agents sont très vastes [25], des exemples d'applications sont :

- Assistance du commerce électronique : les agents sont utilisés pour les négociations modérées, en comparant des propositions, etc.,
- Simulation des systèmes décentralisés: agents comme un outil de modélisation (biologie, règlement de la circulation etc.)
- Appareils portatifs et mobiles : les agents dirigent des systèmes mobiles (téléphone cellulaire, PDA, etc.) résolue les problèmes de la connectivité, délégation de tâches de la recherche de l'information, etc.,
- Robots et robot coopération : les agents sont personnifiés dans les robots et réagissent réciproquement dans le vrai monde.
- Interface homme : interface agents (être en face de l'utilisateur), assistants personnels (profil d'utilisateur, apprendre, etc.), des agents qui communiquent et réagissent réciproquement avec les êtres humains (réalité virtuelle, graphique, avatars, etc.,)
- Intégration et inter-opération: agents qui enveloppent des systèmes existants (intégration de l'entreprise, systèmes informatisés, etc.,)
- La désignation des systèmes critiques,les soins médicaux, la fabrication et gestion de la provision, la méthode de résolution distribuée des problèmes qui implique le réal a distribué des acteurs (par exemple programmer, contrôle du trafic aérien, assortiment expert, fabriquer à contrôle de procédure, work),combiné,etc.

L'interaction est la propriété la plus important dans les systèmes multi-agents : seulement à travers l'interaction, les agents sont capables de partager de l'information et exécuter des tâches pour accomplir leurs buts.

Quand un ensemble d'entités partagent un environnement, ce n'est pas trivial de contrôler leurs comportements et synchroniser leurs activités. Les besoins individuels d'un agent peuvent déterminer l'échec des buts du système total. Pour résoudre ces problèmes communs d'une manière cohérente, les agents doivent communiquer parmi entre eux et

coordonner leurs activités. La coordination et communication sont des éléments centraux aux SMA, l'interaction n'a aucun bénéfique sans eux.

En fait, aucun agent ne possède une vue globale de système à laquelle il appartient. Par conséquent, les agents ont seulement une vue locale de ces buts et ces connaissances qui interfère avec les buts et les actions d'autres agents. La coordination est vitale pour éviter le blocage pendant conflits.

Nous présentons dans cette section les composantes d'un SMA pour la création d'un collectif d'agents : la communication, la coordination et l'interaction.

## 2.2 La coordination

Un agent particulier communique avec les autres agents pour bien accomplir ses buts ou les buts de système. Ces buts peuvent être connu ou non explicitement par l'agent.

La communication permise de coordonner le comportement et les actions de différents agents de système, ceci a pour résultats de garde le système dans un état cohérent.

La coordination est une propriété de système multi agent qui exécuter des activités dans un environnement partage.

Le degré de coordination est une mesure par laquelle on évitait des activités étranges par la réduction de réclamation de ressources, éviter le blocage et de maintenir le système sain.

## 2.3 L'interaction

Pour pouvoir considérer les agents comme un collectif, il doit exister entre eux des possibilités d'interaction. Les agents interagissent en vue de réaliser conjointement une tache ou d'atteindre conjointement un but particulier. Les agents interagissant directement ou par l'intermédiaire d'un autre agent ou de l'environnement.

Ferber [26] mentionne : « On appellera situation d'interaction un ensemble de comportements résultant du regroupement d'agents qui doivent agir pour satisfaire leurs objectifs en tenant compte des contraintes provenant des ressources plus ou moins limitées dont ils disposent et de leurs compétences individuelles ».

Chaque agent peut-être caractérise par trois dimensions : ses buts, ses capacités à réaliser certaines taches et les ressources dont il dispose. Les interactions des agents d'un SMA sont motivées par leurs interdépendances selon ces trois dimensions : Leurs buts peuvent être compatibles ou non ; les agents peuvent désirer des ressources que les autres agents

possèdent ; un agent peut disposer d'une capacité nécessaire à un autre agent pour achever votre action.

En fait, on peut caractériser un système par le type de coopération mise en œuvre qui peut aller de la coopération totale à l'antagonisme total [27].

On distingue deux types d'interaction à savoir la coopération et la compétition (voir figure 3).

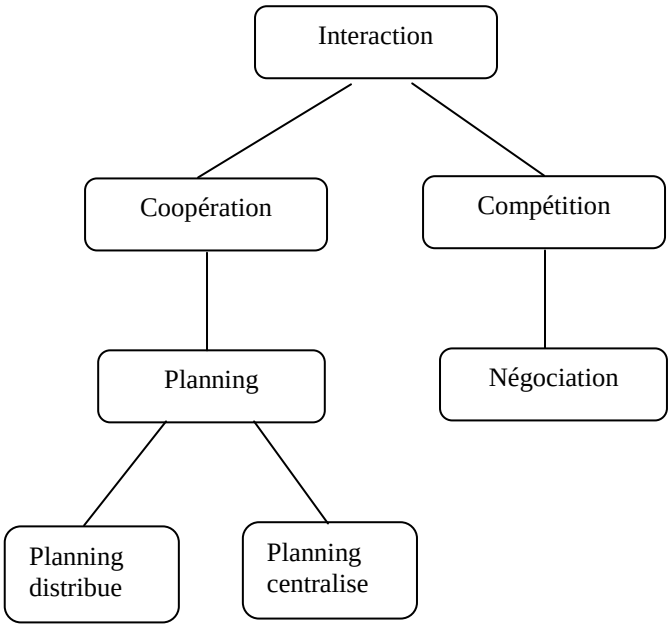


**Figure 1.3 : les différents types d'interaction entre agents [15].**

### 2.3.1 La coopération

La coopération est une caractéristique très importante des SMA. En effet, une résolution distribuée d'un problème ou le partage des tachés et de résultats est les résultats d'interactions coopératives entre les différents agents qui disposent d'un ensemble des caractéristiques qui leur rendent coopératifs.

Ferber a donné une définition plus précise de la coopération : «  On dira que deux agents coopèrent, ou encore qu'ils sont en situation de coopération, si l'une des deux conditions est vérifiée » :

1. L'ajout d'un nouvel agent permet d'accroître d'une manière différentielle les performances du système.
2. L'action des agents sert à éviter ou à résoudre des conflits potentiels ou actuels.

C'est surtout le dernier point qui rend la négociation une interaction coopérative dans le cas où elle interviendrait pour résoudre les conflits.

#### 2.3.1.1 Caractéristiques d'un agent coopératif

Un agent coopératif doit pouvoir :

- Mettre à jour le modèle du monde environnant.
- Déléguer les taches qu'il ne sait pas résoudre à un autre agent dont il connaît les compétences.
- interrompre son plan pour aider les autres.
- Intégrer les informations des autres agents.

#### 2.3.1.2 Types de coopération

Quelle que soit l'organisation d'une société d'agents, un agent peut coopérer suivant les types suivants [28].

- Coopération par partage des taches et de résultats : avec la possibilité de prendre en compte localement les plans des autres.
- Coopération en mode commende, où un agent supérieur A décompose le problème en sous-problèmes qu'il répartit entre les autres agents, ceux-ci les résolvent et renvoient les solutions partielles à A.
- Appel d'offres, où A décompose le problème en sous-problèmes et diffuse la liste aux autres agents. Ceux qui souhaitent contribuer à la résolution du problème envoie des offres à A. ce dernier choisissent parmi elles et distribuent les sous-problèmes, le système fonctionne ensuite en mode commande.

Quant à Werner, il a identifié deux types de coopération à tâches partages :

1. **Coopération négative**, où les agents individuels évitent d'utiliser quelques taches ensembles. Ce type de coopération est similaire aux régions critiques traditionnelles dans les systèmes concurrents et évite que deux entités pour réaliser la même opération simultanément ou partager la même ressource.

2. **Coopération positive** : Un agent peut réaliser une opération particulière seulement si d'autres agents réalisent certaines autres opérations [29].

Le premier type nécessite seulement des mécanismes de communication très simples. Par contre la dernière nécessite des mécanismes de communication plus complexes c'est-à-dire la connaissance sur les actions et les intentions futures.

La coopération positive nécessite quelques formes de coordination pour assurer que les opérations des agents ne sont pas en conflits. Les stratégies de coordination choisies dépendent du système d'agents, mais généralement n'importe quelle stratégie de coordination nécessite quelques formes de coopération communicatives. Le même auteur a identifié les types suivants des coopérations communicatives :

Maître esclave avec une manière de communication, où le maître ordonne à l'esclave ce qu'il fait.

Coopération unilatérale : Où l'émetteur met à jour les intentions de récepteurs. Donc, quand les intentions des deux agents sont combinées, leurs actions achèvent leurs buts.

Coopération mutuelle, ce type de coopération consiste en une conversation qui inclut des actes de langage informationnels et directifs. Les agents coopérants négocient en communiquant leurs désirs.

#### 2.3.1.3 Modes de coopération

Les méthodes de coopération permettent de définir la manière de coopérer, les plus distingués sont : la coordination d'actions, la collaboration par partage de taches et de ressources et la résolution de conflits.

**-La coordination d'actions est la gestion et l'organisation de l'exécution des activités entre les agents** : Elle s'exprime par un ensemble de taches qui ne sont pas directement productifs mais qui servent à permettre aux actions productives de s'accomplir dans les meilleures conditions.

Donc, la coordination permet aux agents de considérer toutes les taches et de ne pas dupliquer le travail. De ce fait, elle est liée à la synchronisation, la planification et la résolution de conflits.

**-La collaboration par partage de taches et de ressources** est l'ensemble des techniques permettant aux agents de se répartir des taches,des informations, et des ressources de manière à résoudre un problème commun,c'est-à-dire préciser « qui fait quoi » pour apporter une solution à ce problème. Parmi les mécanismes utilisés pour l'allocation, on

note les techniques de répartitions centralisées, les mécanismes décentralisés (réseau contractuels, réseaux d'accointances…).

**-La résolution de conflits** est une manière d'empêcher les désaccords. La négociation (qui fait l'objet de chapitre suivant) est la technique la plus intéressante pour pallier à tels conflits.

### 2.3.2 La compétition

C'est une interaction entre des agents égocentrés se basent principalement sur la négociation. La négociation joue un rôle fondamental dans les activités de coopération en permettant aux agents de résoudre des conflits qui pourraient mettre en péril des comportements coopératifs.

Durffe [23] définis la négociation comme le processus d'améliorer les accords (en réduisent les inconsistances et l'incertitude) sur des points communs.

## 2.4 La communication enter agents

Le premier problème qui paraît quand nous considère une société d'agents [30] est la communication pour exécuter leurs taches, propager leurs informations, négocié avec les autres agents, etc.

La communication dans les SMA est la base de la résolution coopérative de problèmes.

Elle n'est pas une simple action d'entrée/de sortie, mais elle s'étend comme un acte pouvant modifier l'état interne de l'agent récepteur. L'agent peut être amené à remettre en cause son plan d'action, ou à modifier ses croyances.

Elle permet aussi de résoudre les conflits par la négociation et de synchroniser les actions des agents. Donc, elle repose explicitement sur des mécanismes d'envoi de message, de réception et de synchronisation.

La communication peut correspondre à une information, une requête ou une interrogation.

Sa sémantique doit être reconnue par l'agent récepteur.

### 2.4.1 Mode de communication

Parmi les différentes méthodes de communication utilisées, deux grandes catégories peuvent se distinguer la communication adressée et la communication par l'environnement.

**-Communication par l'environnement**

Une communication par l'environnement est une communication indirecte. Là où la communication adressée envoie un message à un ou plusieurs agents, la communication par l'environnement déposé le message dans un ou plusieurs environnements.
Le message n'atteint ensuite les autres agents que lorsque ceux-ci iront le percevoir dans l'environnement.
Dans cette situation, l'ensemble des agents a un rôle actif, la simple réception, passive dans le cas de la communication adressée, se transformant en une consultation de l'environnement. Le message est déplacé jusqu'à une étape intermédiaire où le destinataire va le chercher. C'est le cas de la communication par tableaux noirs ou par dépôt et consultation de traces dans l'environnement.

**-Communication adressée**

La plupart des systèmes de communication entre entités informatiques s'appuient sur le modèle de Claude Shannon [31]. Dans ce modèle, la communication est considérée comme la reproduction exacte ou approximative d'une information entre un émetteur et un récepteur.
Selon Claude Shannon, un système informatique communiquant est constitué de cinq éléments :

1. Une source d'information qui produit le message.
2. Un émetteur qui encode voire compresse le message dans le but de générer un signal.
3. Un canal de communication, éventuellement bruité, qui transmet le signal.
4. Un récepteur qui décode voire décompresse le signal dans le but de reconstruire le message transmis.
5. Un destinataire de l'information qui reçoit le message.

La communication adressée est une communication directe. Certains des paramètres spécifiant le message portent l'identifiant du ou des destinataires.
Dans cette situation, l'agent émetteur a un rôle actif, tandis que l'agent récepteur est passif dans sa réception le message. Le message est déplacé jusqu'a son destinataire.
C'est le cas des messages **:** basé sur KQML ou FIPA-ACL.
Cette approche bien qu'ayant prouvée son efficacité est insuffisante pour la définition d'un modèle d'interaction directe entre agents. En effet, la sémantique des messages n'est pas considérée comme pertinente par Claude Shannon.

Or cette dimension est essentielle pour la génération et l'interprétation automatique de messages par des agents logiciels.

Pour communiquer, les agents ne doivent pas seulement lire les messages, ils doivent comprendre ces messages et savoir les manipuler. Dans cette optique, des langages de communication d'agents (ACL, i.e. Agent Communication Language) ont été définis pour automatiser la génération, l'interprétation voire le séquencement des messages.

### 2.4.2 Actes de langage

Plusieurs langages de communication entre agent ont été proposées, mais il y a seulement deux langages de communication standardisée (ACL) qui sont émergés à partir des études linguistiques et de modélisation de connaissances: KQML (Knowledge Query and Manipulation Language) [32] et FIPA ACL (Foundation for Intelligent Physical Agents) [33]. Ces langages de communication sont basés sur la théorie des actes de langage.

La théorie des actes de langage est une théorie de la communication fondée dans les années soixante par Austin [35], Searle [34] ainsi que Vanderveken [36] [37].

Cette théorie affirme que la production d'un énoncé s'assimile à l'accomplissement d'une action. En d'autre terme, parler c'est agir.

En plus, la théorie permet d'identifier les constituants élémentaires de la communication, C'est la raison pour laquelle, elle a été d'une grande influence dans le domaine des systèmes multi-agents.

L'intérêt de cette théorie provient surtout de l'uniformité des mécanismes qu'elle proposées et l'accessibilité de l'intégré dans d'autres disciplines comme les SMA et l'IA.

- **Actes locataires** : ce sont des activités mentales et physiques de formation et d'articulation de l'énoncé. L'acte réussit si l'énoncé est formulé et articulé correctement.
- **Actes illocutoires** : ce sont des actes effectués par l'interlocuteur lors de la formation de l'énoncé.
- **Actes perlocutoires** : ce sont des conséquences indirectes des actes locutoires et illocutoires sur l'auditeur.

#### 2.4.2.1 Les différents types d'actes de langage

La théorie des actes de langage tente de classifier les verbes selon l'acte illocutoire qu'ils accomplissent.

Searle [34] a proposé cinq catégories de performatifs a savoir les actes assertifs, les actes commissifs, les actes directifs, les actes déclaratifs et les actes expressifs. Habermas [38] a révisé les catégories de Searle en ajoutent les actes sociaux et actes pour l'inférence et les actes argumentatives.

McBurney et Parsons [39] ont défini une typologie basée sur les catégories de Habermas. Premièrement les entités référencées par la catégorie de performatif, par exemple : L'état de l'environnement, les états mentaux internes du locuteur, ou les relations sociales entre les participants, deuxièmement, la nature des attaques qui peuvent être faites sur les performatifs de chaque catégorie.

**1. Les actes Assertifs** : Ce sont des actes qui servent à représente l'état de l'environnement et ils sont vérifiés objectivement. Par exemple des expressions sur les croyances d'une proposition. Pour cette proposition, les raisons pour cette croyance doivent être demandée et fournie.

**2. Les actes Expressives** : Ce sont des actes qui cherchent à représenter et exprimer l'état interne de locateur (i.e. des préférences subjectives, attribution des valeurs, ou une intention).

Les raisons sur les états mentaux révélés par le locuteur doivent être demandé et fourni. Cependant, les déclarations font par le locuteur ne peuvent pas être objectivement vérifiées ou réfuter ; Seulement la sincérité de locuteur et leur état interne peut-être défie et évalue**.**

**3. Les Actes Sociales** : ce sont des actes qui sert à affirmer certaine relation sociale ou non entre les différents participants, des exemples de tels actes incluent : Patron, employé, client, fournisseur, etc.

Les raisons pour le locateur qui affirme une relation particulière doivent être demandées et fournies; la contestation de ces expressions prendre la forme de défier l'exactitude normative (normative rightness) de la relation.

**4. Les actes Commissives :** Ici, le locuteur désire que l'environnement soit dans un état particulier, donc l'auditeur s'engage pour entreprendre quelques actions au cours d'action pour établir ou maintenir cet état.

Les promesses et la menace sont des exemples de ces actes, s'ils sont acceptés, ils créent des obligations sur le locuteur dans le processus de dialogue. Par conséquent, ils font la référence aux états mentaux internes (par exemple, désirs, intentions) de locuteur et aux relations sociales qui existent entre ces derniers et l'auditeur.

Parce qu'ils font référence aux états internes, ils peuvent être défiés par la sincérité. Ils peuvent aussi être défiés sur des raisons substantives, par exemple : leur direct ou indirect coût et bénéfiques ; leurs coûts d'opportunité ; leurs conséquences ; leur faisabilité pratique ; etc.

**5. Les actes directifs** : le locuteur désire que l'environnement soit dans un état particulier et donc l'auditeur s'engage à entreprendre quelques action ou cours d'action pour établir ou maintenir cet état. Requests, Commands, Warnings et Entreaties sont des exemples de ces actes.

Ces actes créent des obligations sur l'auditeur vers le locuteur dans le processus de dialogue. Ils font la référence aux états mentaux internes de locateur (désirs par exemple) et l'auditeur (les intentions par exemple) et aux relations sociales qui existent entre eux.

Ils peuvent être défiés sur toutes les raisons substantives que les actes commissives peuvent être aussi défiés en plus de la contestation de l'exactitude normative des relations sociales exigée pour leurs expressions valide.

**6. Actes d'inférence** : Ce sont des actes qui font référence plutôt au contenu des actes précédents, ils permettent de tirer des conclusions ou d'évaluer leurs implications.

La contestation de ce type d'actes ce fait en examinant la convenance ou la validation de l'inférence qui a été faite.

**7. Actes argumentatives** : Ce sont des actes qui font référence au contenu des actes définis précédemment, par exemple : questions, challenges et requests. Ces actes peuvent être attaqués et ils peuvent créer des obligations dialectiques de la part du locuteur et/ou l'auditeur.

**8. Actes de contrôle :** sont des actes qui font référence à l'acte elle-même, ils visent à synchroniser la communication. Des exemples de tels actes sont la demande de répéter une locution.

### 2.4.3 Langages de communication

Pour communiquer, des agents doivent connaître, comprendre et utiliser un langage commun, appelé langage de communication d'agents (ACL, i.e. Agent Communication Language).

Dans les SMAs cognitifs, l'hypothèse la plus répandue est que la communication inter-agent sera plus fructueuse si elle se fait par l'intermédiaire d'un langage de communication

explicite [40]. La propriété essentielle qui rend le langage utile, c'est que le sens de ses signes soit partagé. À cet effet, un tel langage doit avoir :

1. Une syntaxe : on définit la structure des symboles.
2. Une sémantique : on définit le sens de ces symboles.
3. Une pragmatique : on définit la manière dont ils devront être utilisés.

Les langages de communication agent, notés ACL [Agent Communication Language] adressent le niveau intentionnel et social des agents. Ils se différencient donc des mécanismes de la théorie de l'information non seulement par leur structure et leur syntaxe plus complexe, mais aussi par leurs spécifications génériques et précises. Leur puissance expressive, héritée de la théorie des actes de langage permet d'envisager et de développer des SMAs théoriquement hétérogènes et ouverts. Les deux ACLs le plus connus sont KQML et FIPA-ACL. Voyons en quoi ils consistent.

#### 4.3.3.1 KQML [Knowledge Query and Manipulation Language]

Le langage KQML [32] a été proposé pour supporter la communication inter agents. Il est issu d'un projet de la DARPA, le KSE [Knowledge Sharing Initiative], initialement prévu comme moyen d'échange d'informations entre programmes à base de connaissances. Cependant, sa structure orientée message et la généricité de ses primitives lui permet d'être utilisé comme ACL.

C'est l'ACL le plus utilisé (et implémenté) dans la communauté SMA. Le développement de KQML n'est pas centralisé et ainsi plusieurs variantes incompatibles sont nées.

Ce langage définit un ensemble de types de messages (appelés performatifs) et des règles qui définissent les comportements suggérés pour les agents qui reçoivent ces messages.

Premier apparu, KQML fourni un ensemble d'actes de langage standard et utiles. Le langage est structuré selon trois niveaux enchâssés :

1. La couche de communication : renseigne la communication (identité du récepteur, de l'émetteur et nature de la communication). Elle est minimale car KQML ne prend pas en charge le transport lui-même (TCP/IP, SMTP, IIOP ou autres).
2. la couche message : donne des indications sur le contenu du message (langage et ontologie utilisé pour le contenu, type d'acte de langage attaché au contenu). C'est la couche centrale de KQML qui définit le type d'interaction que les agents KQML peuvent avoir. La couche de contenu : contenu du message exprimé en KIF [Knowledge Interchange Format], Prolog, KQML ou autre. Notons que KQML ne

traite pas cette couche si ce n'est pour savoir où le contenu commence et se termine. Comme le contenu du message est opaque, c'est à la couche message de le renseigner.

Les primitives de KQML sont appelées performatifs même si elles ne sont pas des actes performatifs au sens linguistique du terme. Cette appellation qui prête à confusion a d'ailleurs été critiquée. Les performatifs KQML sont divisés en trois catégories :

1. 7 performatifs de régulation de conversation traitent quelques cas particuliers (sorry, error) et permettent quelques variantes de la conversation (standby, ready, next, rest, discard).
2. 17 performatifs de discours permettent l'échange d'informations et de connaissances (ask-if, tell, deny, stream-all,...).
3. 11 Performatifs d'assistance et de réseau pour étendre la conversation à plus de deux agents (forward, broker-all,...).

#### 4.3.3.2 FIPA-ACL

C'est le seul effort réellement organisé pour créer un ACL standard. Ce langage a été proposé par la corporation de chercheurs [33] FIPA (Foundation for Intelligent Physical Agent), organisation à but non lucratif, crée en 1996 et qui propose des standards pour l'interopérabilité des agents logiciels.

Il a été conçu pour palier aux faiblesses des différentes versions de KQML.

FIPA-ACL diffère de KQML en ce qu'il a été directement doté d'une sémantique. En effet, la version originale de KQML ne décrivait que la syntaxe de ses messages et rien n'était dit sur leur sens précis (indépendamment qu'ils correspondaient grossièrement à différents types d'actes de langage). Ce n'est que plus tard, qu'une sémantique a été proposée pour KQML.

La sémantique d'un message FIPA-ACL est définie dans [41]. Un message FIPA-ACL est constitué d'une liste non ordonnée de 14 champs dont nous précisons ici le sens.

Un message FIPA-ACL comporte 2 champs relatifs à la communication. sender et receiver correspondent respectivement à l'agent émetteur et à l'agent récepteur du message.

Un message FIPA-ACL comporte 4 champs relatifs au contenu.

- Le champ content indique le contenu propositionnel du message.

- Le langage d'encodage du contenu est spécifié par le champ encoding. Ce contenu doit être une expression bien formée du langage de représentation spécifié par le champ language.
- Les constantes du contenu propositionnel doivent être définies dans une ontologie spécifiée par le champ ontology.
- L'enchaînement des messages, appelé conversation, est géré à l'aide de cinq champs.
- Protocole fait référence à l'un des onze protocoles proposés (par exemple, le protocole Contract Net [42]; conversation-id comporte l'identifiant de la conversation à laquelle appartient ce message ; reply-with comprend l'identifiant de ce message auquel il pourra être fait référence par la suite ; in-reply-to englobe l'identifiant du message auquel celui-ci repond et reply-by renferme la date butoire de réponse.

La pragmatique des messages FIPA-ACL, i.e., la sémantique des performatifs est formulée en termes d'états mentaux. On parle alors d'approche mentale.

La sémantique de ces performatifs est spécifiée non seulement en langage naturel mais également de manière formelle en SL (Semantic Language). La FIPA propose dans [43] une bibliothèque de 22 performatifs.

# 3. Conclusion

Ce chapitre a l'intention de familiariser le lecteur avec les concepts clés des systèmes multi-agents à travers un état de l'art.

Le chapitre est divisé en deux grands titres. Premièrement, une présentation de concept d'agent et leurs caractéristiques à savoir l'autonomie, l'adaptabilité et la sociabilité, en second lieu, les différents types d'agents sont introduis.

Deuxièmement, nous avons présenté les systèmes multi-agents, leurs domaines d'applications et leurs propriétés à savoir : l'interaction, l'organisation et les mécanismes misent en œuvre pour la communication.

# *Chapitre II*
# *Protocoles d'Interaction & les Ontologies*

## 1. Introduction

Dans les systèmes distribués, l'interaction est un mécanisme important pour supporter le partage des ressources et de coordonner les activités entre les différents composants de ces systèmes. Le concept de protocole d'interaction est un moyen efficace pour structurer et organiser les échanges entre ces composants.

Au-delà des protocoles de bas niveau qui assurent la communication, l'intérêt se porte aujourd'hui sur des protocoles de haut niveau, issus des Systèmes multi-agents (SMA) comme la négociation, enchère, délégation, etc.

De l'autre côté, les applications du commerce électronique deviennent de plus en plus populaires, et ils sont caractérisés par scénarios flexibles, dynamiques, et les règles

D'interaction peuvent changer dans une interaction ou entre interactions.

Dans ce contexte, la seule pré-condition requise est que les agents partageront de la connaissance de fond et s'engager à quelques règles communes de rencontre.

Pour permettre de créer des systèmes ouverts est alors utile de les isoler afin de bien les étudier, les modéliser et les implémenter comme des entités à part entière pour permettre ainsi leur partage, leur recherche et leur invocation.

Cela repose naturellement sur l'exploitation du Web comme technologie supportant la distribution, et sur le Web Sémantique [44] en particulier comme moyen de produire des représentations sémantiquement riches, partageables et accessibles. Plus précisément, les ontologies.

Ce chapitre est consacré à la présentation des protocoles d'interaction et leurs spécifications par les ontologies, nous commençons par la définition de la notion protocole d'interaction et leurs différents types. Ensuite, nous présentons les ontologies, ces caractéristiques, leurs domaines d'application. Puis, Nous présentons les outils nécessaires de leur développement à savoir, les langages de représentation, les outils d'édition et d'interrogation. Enfin, nous discutons la représentation des protocoles d'interaction et plus spécifiquement les protocoles de négociation par les ontologies.

# 2. Protocole d'interaction

Le chapitre précédent décrit les mécanismes qui permettaient aux agents de communiquer en échangeant des messages simples. Selon Weiss [15] les protocoles d'interaction gouverner l'échange d'une série des messages entre agents. Une conversation est une séquence de messages échangés entre deux ou plusieurs agents. Elle doit être conforme à un protocole d'interaction.

On appelle protocole d'interaction un ensemble de règles plus ou moins génériques qui sont partagées par les agents du système et auxquelles les interactions doivent se conformer.

Les protocoles permettent de guider l'interaction des agents. Ce sont des plans définis a priori pour coordonner leurs interactions. On affecte un nom à chacun des protocoles pour les identifier.

Un certain nombre d'agents, deux ou plus, participent à une conversation. Le protocole spécifie le rôle qui leur est affecté.

Par exemple, l'agent qui débute une conversation joue le rôle de l'initiateur, le comportement d'un protocole décrit la séquence des messages qui peuvent être échangées.

Ce comportement est spécifié dans un langage formel ou semi formel à savoir les automates finis déterministes pour décrire de manière plus formelle le comportement d'un protocole et l'AUML respectivement.

Plusieurs protocoles d'interaction sont conçus pour les agents. Dans le cas où les agents auraient des buts conflictuels ou sont simplement ego contres, l'objective de protocole est de maximiser l'utilité de l'agent [45].

Dans le cas où les différents agents auraient des buts similaires ou des problèmes communs, comme dans la résolution distribuer des problèmes (DSP), l'objective de protocole est de maintenir globalement la cohérence de performances de l'agent sans la violation de l'autonomie, c'est-à-dire sans un contrôle global explicite [46].

Dans la littérature il existe plusieurs types de protocoles d'interaction, les plus connus sont :

## 2.1 Les protocoles de coordination

Dans un environnement avec des ressources limitées, les différents agents du système doivent coordonner leurs activités pour assister leurs propres intérêts ou de satisfaire le groupe des buts.

Les différentes actions des différents agents nécessitent la coordination à cause de l'interdépendance entre agents.

On a besoin de satisfaire les contraintes globales du système, mais aucun agent n'a une suffisance en termes de compétences, ressources, ou informations pour achever les buts du système. Des exemples de coordination inclure l'approvisionnement opportun par des informations d'un autre agent, pour assure que les leurs actions sont synchrones.

## 2.2 Les protocoles de coopération

Plusieurs protocoles de coopération partagent la stratégie fondamentale : La décomposition et la distribution des tâches. Cette approche peut réduire la complexité des tâches: Une sous- tâches simple demande moins compétent agent et peu des ressources.

Mais, le système doit décider parmi des décompositions alternatives, si réalisables.

Le processus de décomposition doit considérer les ressources et les capacitives possède par l'agent. Aussi, peuvent être des interactions entre les sous- tâches et des conflits entre aux.

La décomposition des taches est la responsabilité de designer du système. On trouve plusieurs protocoles d'interactions coopératives qui nécessitent la décomposition et la distribution des tâches. Les mécanismes de marché, Contract net protocole, la planification multi agent, etc.

Le plus connu est le protocole Contract Net [47] [48], c'est un protocole coopératif pour la résolution des problèmes entre agents. Proposé par Smith en 1980, est un protocole de haut niveau pour la communication entre les noeuds d'un résolveur de problèmes distribué.

Il facilite le contrôle distribué de l'exécution de tâches coopératives avec une communication efficace entre les noeuds.

Ce protocole est utilisé par le domaine de commerce électronique pour l'échanger des services et les biens, le protocole fournit des solutions pour la distribution des sous-taches sur les différents agents. Pour Smith, la négociation comporte quatre composantes importantes :

- C'est un processus local qui n'implique pas un contrôle centralisé,
- L'échange d'information a lieu dans les deux sens,
- Chaque partie dans la négociation évalue les informations de sa propre perspective,
- L'accord final est réalisé par une sélection mutuelle (le contractant a choisi de répondre au manager et le manager a choisi d'affecter la tâche au contractant).

Le manager va :

- Annoncer la tâche dont il veut qu'elle soit accomplie,
- Recevoir et évaluer les propositions des contractants,
- Choisir l'une de ces offres,
- recevoir et synthétiser les résultats.

Un contractant va :

- Recevoir la demande de travail,
- Évaluer sa capacité à y répondre,
- Répondre (je ne peux pas, faire une offre),
- Si son offre est retenue, fait le travail,
- Renvoyer ses résultats.

## 2.3 Les protocoles de négociation

C'est la forme d'interaction la plus fréquente entre des agents avec des buts conflictuels Weiss [15], la négociation est un processus avec lequel une décision commune est atteinte entre deux ou plusieurs agent pour l'accomplissement d'un but ou objective individuel. Premièrement, les agents du système échangés leurs conflictuelles positions, et essayer de trouve un accord acceptable entre eux.

Les éléments importants de négociation sont, langage de communication entre agents, le protocole de négociation, et la stratégie de négociation. Nous voyant la négociation en détail dans le chapitre suivant.

# 3 Technologie de web sémantique et des ontologies

L'initiative a été inspirée par la vision de son fondateur Tim Berners-Lee [1] ; d'un Web plus flexible, automatisé, automatique et auto-adaptif, qui fournit une expérience plus riche et plus interactive pour les utilisateurs.

L'objectif de l'initiative Web Sémantique est d'avancer l'état du Web actuel à travers l'usage de sémantique. Le Web actuel est essentiellement syntaxique, la structure des ressources été bien définie, mais leur contenu reste inaccessible aux traitements machines, seuls les humains étant capables de l'interpréter.

Le Web sémantique a alors l'ambition de lever cette difficulté en associant aux ressources du Web des entités ontologiques comme références sémantiques, ce qui permettra aux différents agents logiciels d'accéder et d'exploiter directement le contenu des ressources et

de raisonner dessus. Ce référencement sémantique peut aussi résoudre les problèmes d'interprétation des ressources informationnelles provenant des applications hétérogènes et réparties [49] et de permettre ainsi à ces applications d'être intégrées sémantiquement [50].
En plus, le Web Sémantique [44] en particulier est un moyen de production des représentations sémantiquement riches, partageables et accessibles. Plus précisément, les ontologies, constituent une solution complète pour décrire d'une façon consensuelle les connaissances d'un domaine d'application, et de les partager [51] ainsi que pour résoudre les problèmes d'interopérabilité sémantique.
Les ontologies visent à décrire de façon consensuelle l'ensemble des informations pertinentes d'un domaine d'application.

## 3.1 Notion d'ontologie

La littérature d'Intelligence Artificielle contient beaucoup de définitions d'une ontologie ; beaucoup de ceux-ci contredisent l'un l'autre. Il existe plusieurs définitions de terme ontologie selon le contexte de domaine employé à savoir la philosophie, la linguistique ou l'intelligence artificielle.
De nombreuses définitions ont été offertes pour donner un éclaircissement sur ce terme, mais aucune de ces définitions ne s'est explicitement imposée. Cependant, le terme d'ontologie est employé parfois de façon confuse et abusive [52]. Les définitions de ce terme ne sont pas toujours consistantes et cela dépend des domaines spécifiques [53] [54] [55].
La définition la plus commune présente une ontologie comme étant « une spécification explicite et formelle d'une conceptualisation partagée » [56]. En d'autres termes, une ontologie est une description formelle d'un domaine du monde réel.
C'est une conceptualisation dans le sens où elle fournit un vocabulaire formalisé de concepts, de leurs relations, et des axiomes qui leur sont associés.
L'ontologie fournit ainsi une base de compréhension commune à plusieurs communautés ou systèmes. Dans ce qui suit, la notion d'ontologie sera adaptée au cadre de la conception des protocoles de coordination dans les systèmes distribués.
**Définition1** : « une ontologie définit les termes et les relations de base du vocabulaire d'un domaine ainsi que les règles qui indiquent comment combiner les termes et les relations de façon à pouvoir étendre le vocabulaire » [57].

Dans le cadre de l'intelligence artificielle, Neches [57] propose cette définition afin de montrer comment l'élaboration des ontologies s'effectuent, et cela en présentant les directives relativement floues : repérer les termes de base et les relations entre les termes, identifier les règles servant à les combiner, fournir des définitions de ces termes et de ces relations. Notons que d'après cette définition, une ontologie inclut non seulement les termes qui y sont explicitement définis, mais aussi les termes qui peuvent être créés par déduction en utilisant les règles.

Dans ce mémoire une ontologie est une description formelle explicite de concepts dans un domaine de discours (classe quelquefois appelée concept), bien que, chacun des propriétés de chaque concept décrit plusieurs caractéristiques et attributs du concept (appelés rôles ou propriétés), et restrictions sur les emplacements (facettes appelé des restrictions du rôle).

Une ontologie avec un ensemble d'instances individuelles de classes constitue une base de connaissances.

Les classes sont le centre de la plupart des ontologies; elles décrivent des concepts dans le domaine.

Une classe peut avoir des sous-classes qui représentent des concepts qui sont plus spécifiques que la superclasse.

Dans la pratique, développer une ontologie inclut :

- Définissant les classes dans l'ontologie.
- Arranger les classes dans une taxonomie (sous-classe, super hiérarchie).
- Définissant les emplacements et décrire les valeurs permises pour ces emplacements.
- Remplacer les valeurs des emplacements pour les instances.
- Nous pouvons créer alors une base de connaissances en définissant les instances individuelles de ces classes.

### 3.1.1 Les avantages des ontologies

Une ontologie définit un vocabulaire commun pour partager des informations dans un domaine. Elle inclut les définitions de concepts de base d'un domaine interprétables par la machine et les relations entre eux. L'ontologie permet de :

- Partager la structure d'information parmi humains ou agents du logiciel.
- Valider la réutilisation de connaissance du domaine.
- Rendre des estimations du domaine explicite.

- Séparer la connaissance du domaine de la connaissance opérationnelle.
- Analyser la connaissance du domaine.

### 3.1.2 Quelques ontologies

L'OntoWeb, le réseau européen a construit une application web pour autoriser la communauté a enregistré leurs ontologies, méthodologies, outils et langages pour construire des ontologies, aussi bien que les applications dans les domaines semblables : Le web sémantique, commerce électronique, gestion de la connaissance, langage naturel, etc. Les ontologies ont été construites selon leurs domaines d'application [25]. Parmi elles on trouve :

**Ontologie Entreprise** : L'Ontologie de l'Entreprise est une collection de termes et définitions pertinentes à des entreprises commerciales. L'ontologie a été développée dans le Projet Entreprise par l'Institue de l'application de l'intelligence Artificiel à l'Université d'Edimbourg avec ses partenaires : IBM, Lloyd, Register, Logica UK Limited, et Unilever. Le projet a été supporté par le Département du Royaume-Uni de Commerce et d'Industrie sous le Programme de l'Intégration des Systèmes Intelligent (projet IED4/1/8032).
Conceptuellement, l'Ontologie de l'Entreprise est divisé en plusieurs sections principales : Activités et processus, organisme, stratégie et marketing.

**Dublin Core Ontologie** : Ontologie du standard Méta information pour l'interopérable a développé un vocabulaire spécialisé du Méta information pour décrire des ressources qui permettent de construire des systèmes intelligents pour la recherche d'informations.

**TOVE** : Le but du TOronto le projet de l'Entreprise Virtuel est de créer un modèle de description de données qui fournisse une terminologie partagée pour l'entreprise.

**FIPA Langage de communication entre agent** : contient une ontologie qui décrit des actes de discoure pour la communication entre des agents artificiels.

**Ooen Cyc** : Une ontologie de haut niveau pour toute réalité du consensus humaine.

**ProPer** : ontologie pour gérer les qualifications et les compétences des hommes.

**SurveyOntology** : ontologie décrivait les grands questionnaires ; il a été développé pour les clients au bureau du recensement et au département de travail.

**Ontologie UMDL** : ontologie pour décrire le contenu de bibliothèque numérique.

**UMLS** : Le Système de la langue médical Unifié fournit un vocabulaire biomédical de disparates sources telles que terminologies cliniques, sources de la drogue, vocabulaires dans les langues différentes, et terminologies cliniques.

## 3.2 Différentes sortes d'ontologies

Cette section présente les types les plus généralement utilisés d'ontologies. On peut avoir une idée de la connaissance qui est incluse dans chaque type d'ontologie. Les ontologies peuvent être classifiées selon plusieurs dimensions.

### 3.2.1 Objet de conceptualisation

Les ontologies classifiées selon leur objet de conceptualisation c'est-à-dire selon le but de leur utilisation [58].

**Ontologie de haut niveau** : décrit des concepts très généraux comme l'espace, le temps, la matière, les objets, les événements, les actions, etc. Ces concepts ne dépendent pas d'un problème ou d'un domaine particulier, et doivent être, du moins en théorie, consensuels à de grandes communautés d'utilisateurs [59]. Des exemples d'ontologies de haut niveau sont Dolce ou Sumo.

**Ontologie de domaine** : Contrairement aux ontologies de haut niveau, les ontologies de domaine sont plus spécifiques. Elles synthétisent les connaissances spécifiques à un domaine particulier. Elles décrivent le vocabulaire ayant trait à un domaine générique (ex. : l'enseignement, la médecine…), notamment en spécialisant les concepts d'une ontologie de haut niveau [59].

**Ontologie de tâches** : Ce type d'ontologies est utilisé pour conceptualiser des tâches spécifiques dans les systèmes, comme les tâches de diagnostic, de planification, de conception, de configuration, de tutorat. Soit tout ce qui concerne la résolution de problèmes. Ce type d'ontologies décrit le vocabulaire concernant une tâche générique (ex. : enseigner, diagnostiquer, etc.), notamment en spécialisant les concepts d'une ontologie de haut niveau. Certains auteurs emploient le nom « ontologie du domaine de la tâche » pour faire référence à ce type d'ontologie [60].

**Ontologie d'application** : Cette ontologie est la plus spécifique, elle contient des concepts dépendants d'un domaine et d'une tâche particulière, qui sont généralement subsumés par des concepts de ces deux ontologies. Ces concepts correspondent souvent aux rôles joués par les entités du domaine lors de l'exécution d'une certaine activité [59]. Il s'agit donc ici de mettre en relation les concepts d'un domaine et les concepts liés à une tâche particulière, de manière à en décrire l'exécution (ex. : apprendre les statistiques, effectuer des recherches dans le domaine de l'astronomie, etc.).

### 3.2.2 Niveau de formalisme de représentation

D'autre part, selon le niveau du formalisme de représentation du langage utilisé pour décrire l'ontologie, l'auteur dans [61] propose une classification comprenant quatre catégories :

**1. Informelles** : ontologies opérationnelles dans un langage naturel (sémantique ouverte) ;

**2. Semis-informelles** : utilisation d'un langage naturel structuré et limité ;

**3. Semis-formelles** : langage artificiel défini formellement ;

**4. Formelles** : Utilisation d'un langage artificiel contenant une sémantique formelle, ainsi que des théorèmes et des preuves de propriétés telle la robustesse et l'exhaustivité [58].

## 3.3 Les ontologies

Dans cette section, nous allons voir pourquoi a-t-on besoin des ontologies ?. Les ontologies sont utilisées dans plusieurs domaines, les plus répandus sont :

1. Communication.
2. Interopérabilité entre les systèmes.
3. Ingénierie des systèmes.

### 3.3.1 Communication

Les humains peuvent communiquer efficacement s'ils ont des connaissances ou des points de vue partagés. Ces connaissances partagées peuvent être obtenues si le domaine est explicitement décrit sans confusion terminologique ou conceptuelle pour être compris de la même façon par tout le monde. Une ontologie facilite la communication en fournissant une spécification explicite d'un domaine qui représente un modèle normatif. De plus, les ontologies permettent d'assurer la consistante et d'enlever l'ambiguïté dans les descriptions des connaissances concernant un domaine spécifique. Finalement, les ontologies peuvent intégrer différentes perspectives des utilisateurs. Quand les utilisateurs (qui ont différentes perspectives d'un domaine) partagent une ontologie, ils ont une perspective standard.

### 3.3.2 Interopérabilité

L'interopérabilité implique la possibilité de pouvoir demander et recevoir des services entre des systèmes interopérables. Deux systèmes sont considérés interopérables s'ils vérifient les deux conditions suivantes :

- Ils opèrent comme une unité afin de réaliser une tâche commune.
- Ils peuvent échanger des messages et des requêtes.

Les ontologies permettent de faciliter l'interopérabilité en intégrant les connaissances concernant différents domaines dont l'objectif est de décrire un domaine unifié ou accomplir une tâche commune. Elles permettent aussi d'intégrer les différents vocabulaires concernant certains domaines. Pour ce faire, les ontologies de ces domaines doivent être intégrées par les méthodes d'intégration d'ontologies afin de partager un même vocabulaire.

### 3.3.3 Ingénieries des systèmes

Le développement des systèmes basé sur les ontologies a donné un profit à l'ingénierie de systèmes qui peut être résumée comme suit :

- Réutilisation : l'ontologie encode les informations relatives à un domaine (y compris les composants logiciels) de sorte que le partage et la réutilisation sont possibles.
- Acquisition des connaissances : l'ontologie guide l'acquisition des connaissances.
- Sûreté : l'ontologie rend possible l'automatisation du processus de vérification de consistance.
- Spécification : l'ontologie aide le processus d'identification des besoins et la définition des spécifications des systèmes [61].

## 3.4 Outils de développement des ontologies

Les outils de développement d'ontologies qui existent sur le marché aujourd'hui sont divers et variés à bien des égards. Dans cette section, nous passons en revue les principaux outils disponibles.

### 3.4.1 Langage de spécification d'ontologies

Plusieurs langages de spécification d'ontologies (ou langages d'ontologies) ont été développés pendant les dernières années, et ils deviendront sûrement des langages d'ontologie dans le contexte du Web sémantique. Certains d'entre eux sont basés sur la syntaxe de XML, tels que XOL (Ontology Exchange Language), SHOE (Simple HTML Ontology Extension - qui a été précédemment basé sur le HTML), OML (Ontology Markup Language), RDF (Resource Description Framework), RDF Schéma. Les 2 derniers

sont des langages créés par des groupes de travail du World Wide Web Consortium (W3C).

En conclusion, trois langages additionnels sont établis sur RDF(S) pour améliorer ses caractéristiques: OIL (Ontology Inference Layer), DAML+OIL et OWL (Web Ontology Language). La figure ci-dessous présente des langages de spécification d'ontologie, qui ont été récemment développés. La figure ci-dessous représente les rapports principaux entre tous ces langages sous la forme d'une pyramide des langages du Web sémantique.

#### 3.4.1.1 RDF

RDF [62] est un langage d'assertion et d'annotations. Les assertions affirment l'existence de relations entre les objets. Elles sont donc adaptées à l'expression des annotations que l'on veut associer aux ressources du Web. RDF est un langage formel qui permet d'affirmer des relations entre des « ressources ». Le modèle RDF définit trois types d'objets:

Ressources : les ressources sont tous les objets décrits par RDF. Généralement, ces ressources peuvent être aussi bien des pages Web que tout objet ou personne du monde réel. Les ressources sont alors identifiées par leur URI (Uniform Resource Identifier) ; Propriétés : une propriété est un attribut, un aspect, une caractéristique qui s'applique à une ressource. Il peut également s'agir d'une mise en relation avec une autre ressource ; Valeurs : les valeurs en question sont les valeurs particulières que prennent les propriétés. Ces trois types d'objets peuvent être mis en relation par des assertions, c'est à dire des triplets (ressource, propriété, valeur), ou encore (sujet, prédicat, objet).

#### 3.4.1.2 RDF(S)

Comme son nom l'indique, RDFS a pour but de définir des schémas de méta-données. Il définit le sens, les caractéristiques et les relations d'un ensemble de propriétés. La définition peut inclure des contraintes pour les valeurs potentielles et l'héritage des propriétés d'autres schémas. Il est, en effet, une extension sémantique de RDF afin de fournir un mécanisme pour décrire les groupes associés de ressources et les relations entres les ressources [63]. L'intérêt de RDFS est qu'il facilite l'inférence sur les données et renforce la recherche sur ces données.

#### 3.4.1.3 DAML-OIL

DAML [64] est un langage qui a comme but de fournir les fondations pour la génération suivante du Web sémantique. Le langage a adopté d'abord RDFS comme langage d'ontologie pour l'interopérabilité sémantique entre projets. Comme RDFS n'est pas assez expressif relativement aux exigences du Web sémantique, un nouveau langage nommé DAML-ONT a été développé en tant qu'extension de RDF avec les capacités d'un langage de représentation du savoir : orienté objet et basé sur cadre [65]. En même temps, un groupe des chercheurs (la plupart d'entre eux sont européens) au sein d'IST d'Union européenne, avec le même but, développe un autre langage d'ontologie appelé OIL. Ce langage a une syntaxe basée sur RDF et il est explicitement construit pour que sa sémantique puisse être spécifiée à travers une description logique très expressive, la logique de description de type SHIQ [65]. DAML+OIL est la combinaison de ces deux langages. Il hérite des avantages de ces deux langages. En conséquence, DAML+OIL est un langage très expressif et lisible par la machine ainsi que par un être humain avec une syntaxe basée sur RDF.

#### 3.4.1.4 OWL

OWL signifie Web Ontology Language, défini par le W3C [66], Le langage OWL est basé sur la recherche effectuée dans le domaine de la logique de description. OWL permet de décrire des ontologies, c'est-à-dire qu'il permet de définir des terminologies pour décrire des domaines concrets. Une terminologie se constitue de concepts et de propriétés (aussi appelés rôles en logiques de description). Un domaine se compose d'instance de concepts.
OWL est un langage de représentation des connaissances d'un domaine défini pour être utilisé dans le cadre du Web sémantique. Il permet de définir des concepts et des relations à partir d'un ensemble de connecteurs spécifiques au langage et dispose d'une syntaxe XML. Ce dernier repose sur la logique des descriptions pour décrire les hiérarchies, les axiomes et les contraintes. OWL permet de représenter les classes (owl : class), les liens sémantiques (owl: objectProperty), les types de données (owl : dataTypeProperty) et aussi les cardinalités. Il supporte aussi les hiérarchies de spécialisation, l'équivalence entre les concepts (owl : equivalentClass), l'intersection et l'union entre classes ou concepts (owl : unionof, owl : intersectionof).
Le langage OWL se compose de trois sous-langages qui proposent une expressivité croissante, chacun conçu pour des communautés de développeurs et des utilisateurs

spécifiques: OWL Lite, OWL DL, OWL Full. Chacun est une extension par rapport à son prédécesseur plus simple. OWL Lite répond à des besoins de hiérarchie de classification et de fonctionnalités de contraintes simples de cardinalité 0 ou 1. Une cardinalité 0 ou 1 correspond à des relations fonctionnelles, par exemple, une personne a une adresse. Toutefois, cette personne peut avoir un ou plusieurs prénoms, OWL Lite ne suffit donc pas pour cette situation.

OWL DL concerne les utilisateurs qui souhaitent une expressivité maximum couplée à la complétude du calcul (cela signifie que toutes les inférences seront assurées d'être prises en compte) et la décidabilité du système de raisonnement (c'est- à -dire que tous les calculs seront terminés dans un intervalle de temps fini). Ce langage inclut toutes les structures

OWL avec certaines restrictions, comme la séparation des types: une classe ne peut pas aussi être un individu ou une propriété. Il est nommé DL car il correspond à la logique descriptive. OWL Full se destine aux personnes souhaitant une expressivité maximale. Il a l'avantage de la compatibilité complète avec RDF/RDFS, mais l'inconvénient d'avoir un haut niveau de capacité de description, quitte à ne pas pouvoir garantir la complétude et la décidabilité des calculs liés à l'ontologie [66].

### 3.4.2 Quelques implémentations de framework RDF(S) et Editeurs des ontologies

Plusieurs implémentations de cadre RDF (S) et des nombreux éditeurs d'ontologies utilisant des formalismes variés et offrant différentes fonctionnalités ont été développées. Parmi ces outils on trouve : OILed, OntoEdit, Web ode, DOE, PROTEGE, etc.

Autres que les outils d'implémentation et d'édition d'ontologies, nous pouvons trouver aussi les outils de raisonnement tels que Racer, Fact, etc.

Ainsi que des outils permettant de construire des applications basées sur les ontologies, ils fournissent également un environnement de programmation pour RDF, RDFS et OWL, ainsi qu'un moteur d'inférence basé sur les règles. Parmi ces outils nous pouvons citer Jena, KAON,…etc.

#### 3.4.2.1 Oiled

Oiled [67] a été conçu pour éditer des ontologies dans le langage de représentation OIL, il est souvent considéré comme une simple interface de la logique de description SHIQ. Cet éditeur offre également les services d'un raisonneur, FaCT qui permet de tester la satisfiabilité des définitions de classes et de découvrir des subsomptions restées implicites

dans l'ontologie. L'outil dispose de mécanismes pour la classification et le contrôle de la cohérence des ontologies. La version 3.4 d'OILEd est gratuite et disponible sur le site web d'OILEd.

#### 3.4.2.2 OntoEdit

OntoEdit [68] est également un environnement de construction d'ontologies indépendant de tout formalisme. Des outils graphiques dédiés à la visualisation d'ontologies sont inclus dans l'environnement. ONTOEDIT intègre un serveur destiné à l'édition d'une ontologie par plusieurs utilisateurs. Un contrôle de la cohérence de l'ontologie est assuré à travers la gestion des ordres d'édition.

#### 3.4.2.3 Ontolingua

Ontolingua [69] de l'Université Stanford. Le serveur Ontolingua est le plus connu des environnements de construction d'ontologies en langage Ontolingua. Il consiste en un ensemble d'environnements et de services qui supportent la construction en coopération d'ontologies, entre des groupes séparés géographiquement. Il supporte plusieurs langages et dispose de traducteurs permettant de passer de l'un à l'autre. Il est aussi doté d'une bibliothèque d'ontologies, accessible à distance ou localement via des éditeurs d'ontologies ou des applications.

#### 3.4.2.4 ICS-FORTH RDF Suite

ICS-FORTH RDF Suite [70] fournit un outil open source pour le Web Sémantique. C'est le résultat d'un travail sur comment sauvegarder un fichier manipule RDF(S) dans une base de données relationnelle pur améliorer la vitesse et scalabilité de traitement. Cependant, il n'y a aucun caractère expressif supplémentaire et capacités de la déduction construite sur RDF. Cette suite inclut :

Le Validant de l'analyseur RDF (VRP): un Analyseur RDF qui supporte validation sémantique des de ressources descriptions et schémas.

Le Schéma RDF Spécifique à la base de données (RSSDB): RDF stocker en utiliser la connaissance du schéma qui automatiquement produise un Objet-Relationnel (SQL3) représentation de metadata RDF et charge la description de la ressource.

Le langage RDF Query (RQL): Langage déclaratif pour mettre en doute des schémas RDF et descriptions de la ressource.

RQL est basé sur OQL (ODMG standard pour interroger des bases de données orientées objet). Il permet d'interroger les annotations dans RDF et le schéma dans RDFS. Cette approche est naturelle dans RDF depuis classes et les propriétés sont considérées des ressources et peuvent être rapportées de la même façon comme annotation ressource.
Interroger un schéma inconnu autorise à le découvrir pour l'utiliser après pour interroger les annotations.

RQL supporte des expressions de chemin généralisées qui caractérisent des variables sur les étiquettes pour noeuds (c'est a dire, classes) et bords (c'est a dire, propriétés); l'expression peut faire usage d'expression régulière pour exprimer les contraintes.
L'Interpréteur RQL est développé dans C++ et consiste en trois modules :
L'Analyseur, analyser la syntaxe de questions,
Le Constructeur du Graphe, capturer la sémantique de questions en termes de classification et l'interdépendance d'expressions impliquées;
Le Moteur d'évaluation, accéder aux descriptions RDF de la base de données sous-jacente par des requêtes SQL3,

### 3.4.2.5 PROTÉGÉ-OWL

PROTEGE-OWL [71] est une interface modulaire, développée au Stanford Medical Informatics de l'Université de Stanford7, permettant l'édition, la visualisation, le contrôle (vérification des contraintes) d'ontologies, l'extraction d'ontologies à partir de sources textuelles, et la fusion semi-automatique d'ontologies. Le modèle de connaissances de PROTEGE-OWL est issu du modèle des frames et contient des classes (concepts), des slots (propriétés) et des facettes (valeurs des propriétés et contraintes), ainsi que des instances des classes et des propriétés. PROTEGE-OWL autorise la définition de méta-classes, dont les instances sont des classes, ce qui permet de créer son propre modèle de connaissances avant de bâtir une ontologie. De nombreux plugins sont disponibles ou peuvent être ajoutés par l'utilisateur. L'interface, très bien conçue, et l'architecture logicielle permettant l'insertion de plugins pouvant apporter de nouvelles fonctionnalités (par exemple, la possibilité d'importer et d'exporter les ontologies construites dans divers langages opérationnels de représentation tels que OWL ou encore la spécification d'axiomes) ont participé au succès de PROTEGE-OWL, qui regroupe une communauté d'utilisateurs très importantes et constitue une référence pour beaucoup d'autres outils.

Grâce à toutes ces spécificités que dispose PROTEGE-OWL, notre choix portera sur cet outil pour faciliter l'implémentation de l'ontologie des protocoles d'interactions.

#### 3.4.2.6 Racer

Le système Racer (Renamed ABox and Concept Expression Reasoner ou Raisonneur d'expression de concepts et de ABox renommées). Il se présente sous la forme d'un serveur qui peut être accédé par le protocole TCP ou HTTP. Considéré aujourd'hui comme le plus intéressant, c'est un système de représentation de connaissances, qui implémente un calcul de tableaux hautement optimisé, pour une logique de description très expressive. Il peut interpréter des documents OWL et offre des services de raisonnement aussi bien pour le niveau terminologique de l'ontologie, que pour le niveau assertionnel. Il permet aussi de vérifier la consistance et la subsomption des concepts et d'autres tests plus élaborés sur les instances, les rôles et les restrictions. Bien que le système Racer soit plus facilement connecté à l'outil PROTEGE-OWL, donc, nous allons adopter ce système pour tester la consistance de l'ontologie d'application.

#### 3.4.2.7 Jena

Jena [72] est une API qui offre une abstraction pour les ontologies OWL et qui aide les agents à les manipuler et à les activer à l'aide d'un raisonneur.

Cette API, développée en Java, constitue un cadre applicatif pour le développement des applications pour le Web sémantique.

Jena est un cadre de travail Java open-source permettant de construire des applications de Web sémantique. Elle fournit un environnement de programmation pour RDF, RDFS et OWL, ainsi qu'un moteur d'inférence base sur les règles.

Le cadre de travail inclut :

- API RDF; lecture et écriture RDF en RDF/XML, N3, et N-Triples;
- API OWL; stockage en mémoire et persistant;
- RDQL - un langage d'interrogation du RDF.

Le sous-système d'inférence de Jena est conçu pour permettre à certains moteurs d'inférence ou raisonneurs d'être connecté à Jena.

Le terme inférence est utilisé pour se référer au processus abstrait de dérivation d'informations supplémentaires, et le terme raisonneur pour faire référence à un code objet spécifique qui effectue cette tâche.

Le développement des applications basées sur les ontologies est une tâche très lourde et qui nécessite, préalablement, des infrastructures ou bien des plateformes déjà construites et qui sont prêtes à l'emploi tel que Jena. De ce fait, nous allons opter la plateforme Jena pour développer les scénarios de communications entre les applications de l'entreprise.

# 4 La description des protocoles d'interaction par une ontologie

Cette approche de description des protocoles d'interaction et plus spécifiquement la négociation [73] rendre la représentation des règles de rencontre c'est-à-dire le protocole explicite, compréhensible et partageable par la machine.

Les agents participeront à une session de la négociation s'engagent à l'ontologie partagée qui les explique le mécanisme qui gouverne le processus de la négociation.

L'ontologie est aussi utilisée comme entrée à un algorithme d'apprentissage qui est utilisé par les agents pour adapter leur stratégie au protocole de la négociation spécifique.

De nombreux travaux actuels se sont intéressés aux protocoles d'interaction [74] [75], mais très peu d'entre eux ont envisagé la définition d'une ontologie pour la représentation de ces protocoles.

Deux travaux méritent d'être mis en avant : [76] [77] proposent une ontologie limitée à une classe de protocoles de négociation spécifique au commerce électronique.

Amit [76] propose une ontologie de protocoles et un langage OWL-P pour spécifier,

Si celle que nous utilisons dans notre système.

**Ontologie pour l'interaction**

L'ontologie que nous exploitons dans notre système de dialogue est considérée comme un modèle de référence pour la description des protocoles d'interaction. Elle fournit ainsi une représentation déclarative et explicite de ces protocoles.

L'utilisation de cette ontologie est repartie sur trois phases :

1. Le premier niveau, le plus abstrait, correspond à l'ontologie des protocoles définissant la structure invariante partagée par tous les protocoles.
2. Le deuxième niveau représente la spécification d'un protocole par l'instanciation de cette ontologie.
3. Le dernier niveau, la conversation, le plus concret, correspond à l'exécution ou l'occurrence d'un.

La construction d'une ontologie formelle se révèle être un travail délicat indépendamment du domaine d'application. Le recensement des constituants de l'ontologie (les concepts, les relations et leurs significations) est généralement difficile à résoudre la représentation formelle de ces connaissances à l'aide des langages d'échange.

Pour cette raison, il est habituel de procéder par étapes, en essayant de répondrent à ces questions:

- comment acquérir les connaissances (par exemple à partir d'un corpus),
- comment modéliser les connaissances,
- comment les représenter dans un langage formel interprétable par la machine.

Cette ontologie de protocoles est crée en plusieurs étapes.

Pour la première étape, énumérer l'ensemble des termes et concepts relatifs aux protocoles de haut niveau. Et le plus important, et que cette ontologie est le résultat d'une étudié de quelques protocoles existants dans la littérature comme le protocole Contract Net [47], les protocoles d'enchères, le protocole de Marchandage, le protocole de courtier, les protocoles de négociation à base d'argumentation et d'heuristique, etc.

Elle a l'avantage d'intégrer la description du domaine de négociation (partie statique) et celle du processus de négociation (partie dynamique ou structure de contrôle).

Le comportement d'un protocole est représenté à la manière de la spécification du comportement des processus d'entreprises.

Leur ontologie regroupe des concepts importants tels que les messages, les rôles, les règles, et la notion d'engagement (commitment).

La figure 2.1 donne une représentation graphique de l'ontologie de protocoles en utilisant les concepts du langage OWL à savoir la notion de : class, objectproperty, datatypeproperty, etc.

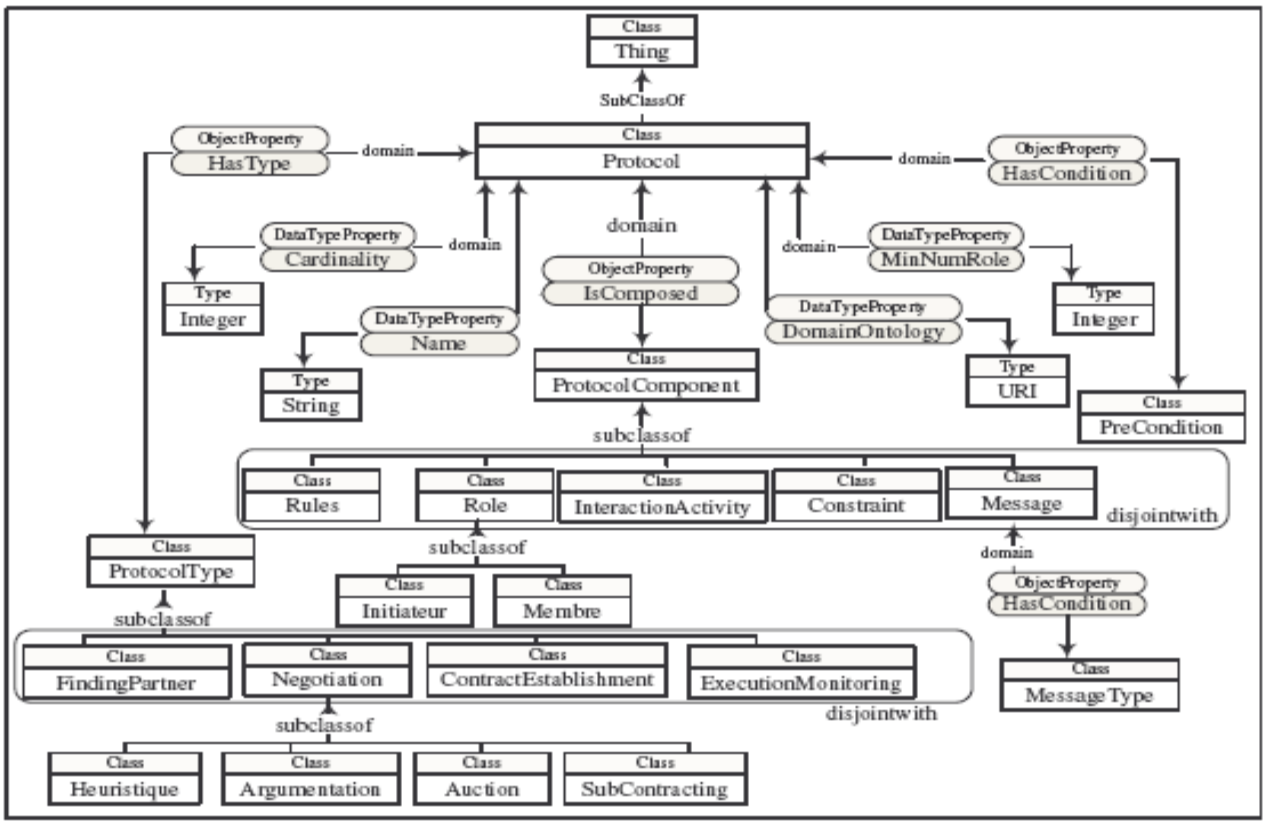


**Figure 2.1 : Ontologie de protocoles : aspects statiques.**

La figure 2.2 illustre la combinaison des concepts d'activité d'interaction et de message. Ces activités définissent les séquences d'interaction permises par un protocole.

La définition du comportement du protocole inclut à la fois les activités ou les opérations et le séquencement des messages ainsi que les rôles impliqués.

Chaque interaction dans un protocole comprend des participants qui jouent des rôles dans le protocole.

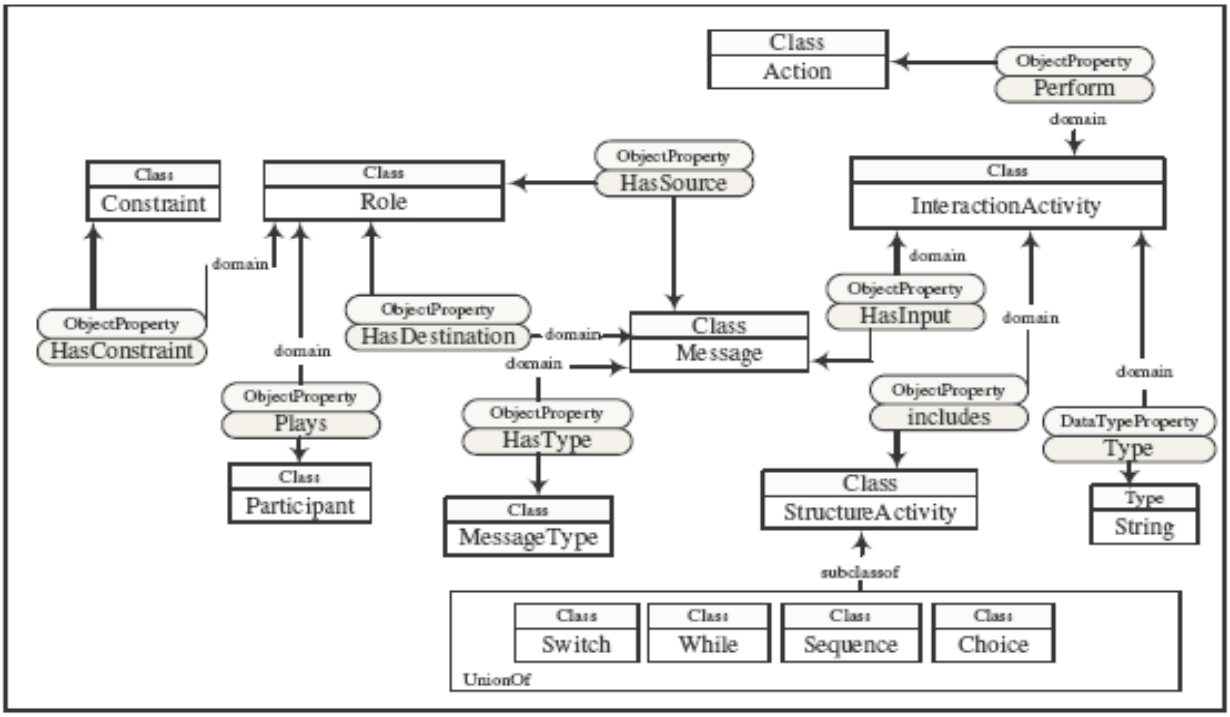


**Figure 2.2 : Ontologie de protocoles : aspect dynamique.**

**Protocole** : représente la classe principale de l'ontologie. Un protocole est défini par son nom, qui doit être unique. La propriété cardinalité indique le nombre minimal de participants dans une interaction selon le protocole considéré.

Les cardinalités 1-1, 1-N, N-1, N-M indiquent respectivement une interaction entre deux participants, un et plusieurs, etc.

Chaque protocole peut faire référence à une ontologie de domaine qui correspond au domaine d'application du protocole (e-commerce, voyage etc.). À chaque protocole correspond un protocolType qui représente la fonction ou l'objectif de chaque protocole (négocier, rechercher, contracter, etc.).

Un protocole se compose d'un ensemble d'éléments qui le définissent : les règles, les rôles, les activités, les contraintes et les messages. Détaillons chacun de ces éléments :

**Règle** : Ce concept définit les normes imposées par le protocole lors d'une conversation. Ces normes réglementent le comportement des participants qui jouent des rôles dans un protocole. Par exemple, dans un protocole d'enchère un participant ne peut pas proposer un prix inférieur au prix courant ou bien un participant ne peut pas effectuer une proposition une fois le temps d'attente écoulé.

**Rôle** : c'est le comportement que peut prendre un participant jouant ce rôle.

**Message** : Il correspond à l'interaction élémentaire permise par un protocole et que les participants d'une conversation peuvent échanger. Un message comporte un certain nombre de paramètres qui sont définis à travers des propriétés telles que l'expéditeur, le receveur et le contenu. Le message doit posséder une sémantique pour être interprété sans ambiguïté par tous les participants. Cette sémantique peut être décrite soit à travers une logique de description soit en utilisant des langages standard tel que le langage FIPA-ACL ou bien OWL pour la description du contenu du message.

**Type de Message** : Ce concept catégorise l'ensemble des messages contenu dans un protocole (information, proposition, promesse, etc.).

La description de la sémantique de l'action peut être décrite en logique de description. Avant l'envoi d'un message, un certain nombre de contraintes doivent être vérifiées. Ces contraintes limitent le comportement des participants au sein d'un protocole.

**Activité d'interaction** : Ce concept décrit les opérations effectuées par un protocole. Une activité comprend des messages et des conditions pour son déclenchement et permet l'exécution d'un ensemble d'actions. L'action représente l'opération déclenchée par le protocole suite à la vérification des conditions. Les structures de contrôle permettent la composition des activités d'interaction qui structurent le comportement de protocole.

## 4.1 Les avantages de la description des protocoles de négociation par les ontologies

Dans cette approche, l'ontologie à pour but principal de faciliter la négociation automatisée qui peut être décomposée dans les sous tâches suivantes :

- Il permet aux agents d'engage dans le processus de négociation sans connaissance préalable du mécanisme de la négociation utilisé ;
- Il permet aux agents d'échanger la connaissance au sujet de mécanismes de la négociation arbitraires ;
- Il réduit la quantité de connaissance codée dans l'agent ;
- Il facilite l'engagement à une vue partagée du domaine de la négociation où agents consentent sur la signification des concepts a utilisé pour décrire le processus de la négociation.

# 5. Conclusion

Les protocoles d'interaction sont des mécanismes utiles pour faire interagir et coordonner des entités distribuées. Ces protocoles sont efficaces et fiables seulement s'ils possèdent une sémantique claire. Un type particulier des mécanismes d'interaction qui attire beaucoup d'attention surtout avec l'émergence du web les applications de commerce électronique à savoir la négociation.

Au long de ce chapitre, nous avons essayé d'éclaircir la notion d'ontologie en présentant certaines définitions. Nous avons montré aussi leurs avantages, leurs domaines d'application et leurs principaux types.

Nous avons découvert après les outils nécessaires de leur développement à savoir les langages de représentation, les outils d'édition et d'interrogation. Tout on conclura par la spécification de protocole de négociation par les ontologies et les avantages qu'il apporte.

Le chapitre suivant présente les systèmes de dialogues.

# *Chapitre III*
# *Les Systèmes du Dialogue*

## 1 Introduction

Ce troisième chapitre adresse un état de l'art de systèmes de dialogues, le chapitre est divisé en trois parties.

Dans la première partie nous introduisons la notion de système de dialogue, les différents types de dialogue ainsi que les éléments qu'ils constituent. Deuxièmement, nous présentons l'argumentation et leurs bénéfices pour construire des systèmes de dialogue multi-agents. Après, nous focalisons sur un type particulier de dialogue qu'est la négociation. Finalement, nous illustrons quelques implantations des systèmes de dialogue.

## 2. Système de dialogue

Selon Charles L. Hamblin [78] un système dialectique est une famille de dialogues régulés, i.e. un dispositif à travers lequel un ensemble de participants communiquent en respectant certaines règles.

Dans le cas le plus simple, un dialogue se déroule entre deux participants qui prennent la parole chacune à leur tour. Elles interrogent et/ou répondent à leur partenaire via des locutions en prenant en compte les locutions précédentes.

Hamblin distingue la dialectique descriptive et la dialectique formelle. La dialectique descriptive est l'étude des systèmes de communication stylisée dans un contexte spécifiquement identifiable d'échanges linguistiques : débat parlementaire, instruction judiciaire, audition, etc.

Par contre, la dialectique formelle vise l'élaboration d'un système simple de règles précises qui ne sont pas nécessairement réalistes.

Le système dialectique proposé par Hamblin utilise deux notions essentielles : la notion de **convention** et la notion de **tableau d'engagements**.

Le dialogue doit se conformer à une convention, i.e. un ensemble de règles dialectiques avec les propriétés suivantes :

**Les règles dialectiques sont contextuelles** : elles dépendent de l'histoire passée du dialogue, le plus souvent seul le dernier coup est considéré.

**Les règles sont déontiques** : elles spécifient les réponses qui sont autorisées et celles qui sont interdites. Un système est régulier (rule-consistent) si un coup n'est pas à la fois interdit et autorisé.

**Les règles dialectiques peuvent être individuelles** : La cible est un participant ou un ensemble de participants.

Une règle dialectique peut contraindre un participant à ne pas se contredire. Il est alors dans l'obligation d'avoir un propos consistant. Formuler une telle règle dialectique n'est pas trivial. C'est la raison pour laquelle la notion de tableau d'engagements a été introduite.

On nomme tableaux d'engagements (commitment stores) les structures de données qui répertorient les engagements pris par les participants au cours du dialogue.

Les tableaux d'engagements sont mis à jour en fonction des locutions émises afin d'enregistrer toutes les déclarations.

- Un agent ou joueur est cohérent dans son propos si son tableau d'engagements est consistant.
- Un agent peut être cohérent dans son propos sans pour autant disposer de connaissances consistantes.
- Un système est sémantiquement cohérent lorsqu'un agent n'est pas obligé d'énoncer une contradiction.

Hamblin distingue deux langages : le contenu des locutions qui est conforme à un langage de domaine et les règles dialectiques sont exprimées dans un langage de dialogues.

Le terme système du dialogue dans notre mémoire couvre seulement les règles du jeu, c'est à dire. , quels actes sont permis ;  Il ne couvre pas des règles pour jouer bien le jeu, c'est-à-dire la stratégie de dialogue.

## 2.1 Les différents types de dialogue multi-agents

Les systèmes du dialogue définissent essentiellement le principe d'un dialogue cohérent et les conditions sous lesquels une locution faite par un agent est appropriée.

Un système du dialogue spécifie quand un agent est autorisé à envoyer une locution appropriée, pour achever le désiré ou but du dialogue.

Cela peut être vue comme une approche de dialogues basé sur la théorie de jeux où les actes (locutions) sont vus comme des mouvements dans un jeu ,et la sémantique de jeu

nous indique si l'acte est approprié à un moment précis est formulé comme une règle du jeu.

Comme la cohérence de dialogue dépend de son but, c'est important d'identifier les classifications de plusieurs types de dialogue.

Walton et Krabbe [3] ont proposé six catégories principales de dialogue humain basé sur les trois composants ; le but total du dialogue, les buts individuels de chaque agent et l'information que chaque agent possède au début du dialogue. Les catégories sont :

**Dialogues de recherche d'information** : où un participant cherche la réponse à quelque question d'un autre participant.

**Dialogue de l'enquête** : Avoir lieu quand les participants collaborent pour chercher une vraie réponse pour une question quelconque.

**Dialogue de persuasion** : Implique un agent qui cherche à persuader un autre de soutenir une proposition.

**Dialogue de la négociation :** des agents qui échangent sur quelque ressource, où le but de chaque agent ne satisfait pas mutuellement les autres.

**Délibération dialogue :** les agents collaborent pour décider quel cours d'action devrait être adopté dans quelque situation.

**Dialogue éristique** : où les agents débattent verbalement comme un remplaçant de combat physique.

Dans notre mémoire, nous nous intéressons qu'un seul type de dialogue qui est la négociation dans le contexte de commerce électronique.

## 2.2 Les éléments qui constituent un système de dialogue multi-agents

Dans le but d'implémenter un système du dialogue, les composants de base doivent être identifiés et définis.

Un des éléments de base d'un système de dialogue est l'ensemble des agents impliqués dans ce dialogue, chaque système du dialogue doit avoir un but.

Les agents peuvent être des humains ou des systèmes informatiques. Les travaux sur la modélisation du dialogue [79] [80] [81] [82] considèrent le dialogue uniquement entre deux agents. Cependant, il existe d'autres formes de dialogue où plusieurs agents sont invoqués.

Chaque agent a un nom et un rôle dans le dialogue.

Le rôle d'un agent peut aider à déterminer le poids des coups avancés par cet agent au cours du dialogue.

Un agent est aussi supposé avoir une base de connaissances qui contient ses croyances, ses désirs ainsi que ses intentions. Elle peut aussi contenir certaines connaissances sur les croyances des autres agents.

Comme suggéré par MacKenzie [80], chaque agent possède une autre base, accessible à tous les agents, contenant ses engagements au cours du dialogue. Cette base est appelée tableau de conversation.

Un système du dialogue contient deux langues, un langage de domaine (représenter des éléments concernant le sujet du dialogue) et un langage de communication (a utilisé par les agents pour envoyer d'une manière cohérente des locutions dans le dialogue).

Une base de connaissance est écrite dans une logique, la logique peut aussi aider un agent à maintenir la cohérence de sa base.

Finalement, les composants centraux d'un système du dialogue sont :

- Le protocole qui détermine les actes autorisés à chaque moment de processus de dialogue,
- Les règles de commencement de dialogue,
- Les règles de mise à jour, qui spécifie les effets des actes sur les bases engagements des agents (croyances),
- Les règles de terminaison qui définissent le résultat d'un dialogue.

## 2.3 Définition formule de système de dialogue multi-agents

Nous spécifions maintenant formellement les éléments qui constituent un système de dialogue.

Les définitions au-dessous de système de dialogue sont basées sur les travaux [83] [84] [85] [86] [87].

Comme notation, le complément $\neg\varphi$ d'une formule $\varphi$ est $\neg\varphi$ si est une formule positive et $\psi$ si $\varphi$ est une formule négative $\neg\varphi$.

- Un langage de domaine $\tau_t$ contenir au moins les conjonctions habituelles.
- Un langage de communication $\tau_c$.

- L'ensemble des coups de dialogue, dénote par $M^{\leq\infty}$, l'ensemble de tout les séquences des coups a partir de langage $\tau_c$, et l'ensemble finis de dialogues, dénote par $M^{\prec\infty}$ , c'est l'ensemble fini de séquences des coups a partir de $\tau_c$. Pour chaque séquence de dialogue $d = m_1,...,m_n,...,$ nous dénotons $m_1,...,m_i$ avec $d_i$.
- Un but de dialogue.
- Un ensemble des participants $\mathrm{A}$, et un ensemble des rôles $\Re$, définie comme un sous ensemble disjoint de $\mathrm{A}$. un participant a peut être ou non possède une base de connaissances inconsistant, $\sum_\alpha \subseteq Pow(\tau_t)$ peut ou non change durant le processus de dialogue. De plus, chaque participant a, une base d'engagement vide $C_\alpha \subseteq \tau_T$, qu'il change durant le dialogue.
- un contexte $\mathrm{K} \subseteq \tau_t$, contenir les informations présuppose et qui doit être respecte durent le dialogue (le contexte est suppose consistant et demeure le même pendent le dialogue).
- Un logique $L$ pour $\tau_t$ , qui peut être monotone et/ou basé sur l'argumentation.
- Un ensemble des règles de mise à jour $E$ pour $\tau_c$, spécifie pour chaque coups ou acte $\varphi \in \tau_c$ ses effets sur la base d'engagement de chaque participant.

  Ces règles sont spécifie Comme une fonction: $C_\partial : M^{\prec\infty} \to Pow(\tau_t)$

La mise à jour de base d'engagements est complètement détermine par les derniers coups avances dans le processus de dialogue et l'état de base avant l'avancement de ce acte.

Si $C_\alpha(d) = C_\alpha(d')$ ,Alors $C_\alpha(d,m) = C_\alpha(d',m)$

- Un protocole de dialogue $\mathrm{P}$ pour $\tau_c$, spécifie l'acte autorise a chaque étape de dialogue. Formellement, un protocole sur $\tau_c$ c'est une fonction $\mathrm{P}$ avec comme domaine le contexte plus un sous ensemble non vide D of $M^{\prec\infty}$ prendre un sous ensemble de $\tau_c$ comme valeur :

  $\mathrm{P} : Pow(\tau_y) \times D \to Pow(\tau_c)$ tell que $D \subseteq M^{\prec\infty}$.

  Les éléments de $D$ sont appelés la séquence finit de dialogue.

Les éléments de $P(d)$ sont appelés des coups après $d$.

Si $d$ est un dialogue légal et $P(d)=\varnothing$, alors $d$ est un dialogue terminé.

$P$ Doit satisfaire la condition suivante: pour tout dialogue finit d et des coups $m, d \in D$ Et $m \in P(d)$ Si $d, m \in D$.

- La conversation est une fonction $T : D \times Pow(\tau_t) \rightarrow A$.
- Un tour d'un dialogue est défini comme une séquence maximale de coups dans le dialogue avant lequel le même joueur avance des locutions.
- La terminaison est définie précédemment comme le cas où aucun coup n'est légal. En conséquence, une définition explicite de la terminaison devrait spécifier les conditions sous quelles $P$ retournes l'ensemble nul.
- Les règles de résultat, définissant le résultat de dialogue. Par exemple, le résultat de la négociation est l'allocation de ressources, dans le dialogue de délibération, c'est une décision sur un cours d'action.
- Une stratégie pour un participant est une stratégie gagnant si dans chaque dialogue joué est d'accord avec la stratégie pour réaliser son but de dialogue. formellement, étant donnée $D_\alpha$, un sous ensemble de $D$, est l'ensemble de tout les dialogues où $a$ est avancer, La stratégie pour un agent $a$ est défini comme une fonction $s_\alpha$ comme suit : $s_\alpha : D_\alpha \rightarrow \tau_c$.

## 3. L'argumentation dans le dialogue

Lorsqu'un agent autonome dialogue avec les autres agents de système, par exemple un dialogue de type négociation entre un acheteur et un vendeur dans le domaine de e-commerce, les agents concernent doivent avoir des mécanismes qui l'ont permets d'interagit et de raisonner pour finalement décider quelle action doit entreprendre avec le monde qui l'entoure, mais le problème est qu'il peut être confronté à différentes sources d'incohérence : croyances erronées, observations non fiables, échanges d'informations avec d'autres agents, etc.

Un agent intelligent doit donc disposer d'un mécanisme de raisonnement et d'interaction qui permet de gérer ces incohérences.

Différentes approches pour construire des modèles d'interaction et de décision artificiellement intelligente pour les systèmes multi-agents ont changé considérablement pendant les deux décennies passées.

On trouve des Agents avec de raisonnements déductifs comme les agents délibératifs de [88]. Dans ce modèle, un agent peut être vue de comme un démonstrateur de théorème où la représentation symbolique de l'environnement de l'agent est sous la forme des formules logiques et la manipulation de ces représentations correspondues à la déduction logique.

La prochaine étape évolutionnaire est l'introduction de raisonnement pratique [89] qui permettre à raisonnement d'être dirigée vers les actions.

Wooldridge [2] définir que le raisonnement pratique est essentiellement le processus de peser des considérations incompatibles et conflictuelles pour et contre options en concurrence (selon ce qui l'agent désire et croit) et décider quelle action entreprendre.

Le raisonnement pratique a produit des modèles de la prise de décision qui ont été utilisés largement dans le domaine des agents, le savoir le modèle BDI (croyance désir intention) [90].

Tous ces modèles sont ajoutés au processus évolutionnaire de systèmes multi-agents et ont fourni des modèles de prise de décision acceptable dans certains contextes.

Cependant, ils ne sont pas robustes et fiables dans le contexte de la technologie agent de la prochaine génération.

Comme il est mentionné dans [91] il peut y avoir trop d'informations pertinente mais partiellement incohérente et en cas d'interaction entre agent, les conflits d'intérêts sont inévitables.

Les modèles traditionnels de décision entre agents sont inhérents de la logique ou des modèles orientés-machine. Les inconvénients de ces modèles sont qu'ils ne peuvent pas traitent efficacement les informations et les données incomplètes ou incohérentes.

D'autre part, les êtres humains excellent au traitement de ce type d'informations en générant des arguments structurés pour et contre certaines décisions ou cours d'action et résolus leurs différences à travers la négociation, délibération, etc.

Récemment l'usage d'argumentation dans les applications informatiques s'est concentré sur les systèmes du dialogue formels, en permettant d'échanger des arguments structurés d'être échangé entre agents.

Donc, l'argumentation peut être utilisée assister les agents autonomes de système en facilitant leur interaction par l'échange et évaluation d'arguments qui supportent des opinions, des demandes, des propositions et finalement des décisions.
L'argumentation est un bon candidat. L'argumentation constitue un modèle adapté au processus cognitif d'un agent autonome et social.

## 3.1 L'argumentation

La théorie d'argumentation [92] est un domaine de recherche riche, interdisciplinaire lie la philosophie, la communication, la linguistique, et la psychologie. Ses techniques et résultats ont trouvé une grande gamme d'applications dans branches théoriques et pratiques d'intelligence artificielle et l'informatique.
Ces applications varient de spécifier la sémantique des programmes logiques [93], à génération du texte du langage naturel [94], de supporter le raisonnement légal [95], un support de pour prise de décision entre des humaines [96] et résolution de conflits [97], etc.
Ces dernières années, la théorie de l'argumentation a gagné l'intérêt croissant communauté de la recherche dans systèmes (MAS).
Dans l'autre cote, les techniques basées argumentation peuvent être utilisée pour spécifier le raisonnement des agents autonomes, tels que révision des croyances et la prise de décision sous l'incertitude et des politiques de préférence non conformes.
En plus, l'argumentation peut aussi être utilisée comme un véhicule pour faciliter l'interaction multi agent, parce que l'argumentation fournit naturellement des outils pour concevoir, implémenter et analyser des formes sophistiquées d'interaction entre des agents rationnels.
L'argumentation a fait des contributions solides pour la théorie et le pratique de dialogues multi-agents.
D'après [82] l'argumentation peut être définie comme suit :
*"Argumentation is a verbal and social activity of reason aimed at increasing (or decreasing) the acceptability of a controversial standpoint for the listener or reader, by putting forward a constellation of propositions intended to justify (or refute) the standpoint before a rational judge.***"**
Le but ultime de l'argumentation est de résoudre un point de vue "controversé"; controversé dans le sens qu'il est soumis à la "justification" ou "réfutation" selon l'information disponible.

Cela distingue l'argumentation de raisonnement déductif classique dans lequel les preuves pour les propositions ne peuvent pas être contestées.

De plus, la nature du "point de vue" peut varier. Un point de vue, en principe, varie d'une proposition pour croire, à un but essayé d'accomplir, à une valeur essayé favoriser.

L'argumentation peut être utilisée pour le raisonnement théorique (au sujet de que croire) aussi bien que raisonnement pratique (au sujet de que faire).

Deuxièmement, l'argumentation est une "activité de raisonnement", est basé sur le fait qu'un processus particulier sera suivi pour influencer l'acceptabilité du point de vue controversé.

Cette activité et les propositions proposées seront évaluées par un "juge rationnel": un système qui définit les rationalités de ces propositions d'après quelques critères.

Finalement, l'argumentation peut être vue comme l'interaction principale entre différent, potentiellement contradictoires arguments, pour en arrivant à une conclusion consistante.

L'aspect le plus crucial de l'argumentation est l'interaction entre arguments. L'argumentation peut nous donner des moyens pour permettre à un agent de réconcilier de l'information conflictuelle, pour réconcilier son état informationnel avec les nouvelles perceptions de l'environnement, et pour réconcilier des informations conflictuels entre les différents agents à travers communication.

C'est pour ces raisons que l'argumentation a commencé à recevoir le grand intérêt dans la communauté des systèmes multi agent. En particulier, l'argumentation le prête naturellement à deux principales de problèmes rencontrées dans les MAS:

**Former et réviser des croyances et des décisions**: L'argumentation fournit des moyens pour forme des croyances et des décisions avec des informations incomplètes, conflictuelles ou incertaines. C'est parce que l'argumentation fournit un moyen systématique pour résoudre des conflits entre différents arguments et arriver à des points de vue consistants et bien supportés.

**Interaction Rationnelle**: L'argumentation fournit des moyens pour structurer le dialogue entre participants qui ont des points de vue potentiellement incompatibles. En particulier, l'argumentation fournit un cadre qui assurer que l'interaction respecte certains principes (par exemple la consistance des déclarations de chaque participant).

## 3.2 Définition formelle de l'argumentation

L'argumentation est une composante majeure du raisonnement de sens commun. On essaye souvent de justifier son opinion sur un point particulier en donnant des raisons renforçant son point de vue.

Ces raisons sont basées sur la notion d'argument définie par Toulmin [98].

L'argumentation est un modèle prometteur pour raisonner avec des informations incomplètes, incertaines ou incohérentes, basé sur la construction et la comparaison des arguments afin de déterminer les plus acceptables d'entre eux.

L'idée de base derrière l'argumentation est la possibilité de dire plus sur la certitude d'un fait particulier plutôt que de le quantifier uniquement par un nombre entre 0 et 1, on évalue la raison pour laquelle le fait est vrai en construisant les arguments en sa faveur.

L'avantage de construire des arguments pour des faits et les utiliser pour quantifier leur certitude est la possibilité de raisonner sur ces arguments eux-mêmes.

L'argumentation peut aussi être utilisée dans le cadre d'un raisonnement révisable. Plusieurs systèmes d'argumentation ont été développés à cet effet [99] [100] [101] [102] [103] [104].

Dans ce chapitre, nous introduisons les concepts de base d'un système d'argumentation a savoir : le langage, le concept d'argument, la relation de contrariété entre arguments, l'acceptabilité.

L'argumentation est une approche de raisonnement avec des informations inconsistantes, basée sur la construction d'arguments et de contre arguments, la comparaison de ces arguments et enfin la sélection des arguments jugés acceptables.

Dans ce qui suit, nous présentent en détail un système formel d'argumentation défini dans [105].

Soit $\Sigma$ Une base de connaissances inconsistante et non déductivement close, contient des formules d'un langage propositionnel $L$, $\perp$ est dénote l'inférence classique et $\equiv$ dénote l'équivalence logique.

Les préférences sur les éléments de $\Sigma$ dépendent de contexte dans lequel elles sont exprimées.

$C = \{c_1,.., c_n\}$, l'ensemble de tous les contextes, muni d'un ordre $\supset$.

Ainsi pour deux contextes $c_1, c_2 \in C$ , $c_1 \supset c_2$ Veut dire qu'une proposition dans le contexte $_{c1}$ préférée à une autre proposition dans le contexte $c_2$.

Chaque contexte $c_i$ est muni d'un pré ordre $\succ\succ_i$ qui donne les préférences sur les éléments de $\Sigma$.

**Definition1**

Un argument est un paire $A = (H, h)$ où $h$ est une formule de $L$ et $H$ Un sous-ensemble de $\Sigma$ tel que :

1. $H$ est consistante ;
2. $H \perp h$ ;
3. $H$ est minimal, il n'existe pas un sous ensemble de $H$ qui satisfait 1 et 2.

$H$ est appelé le support de l'argument, dénote par $H = Support(A)$ et $h$ sa conclusion, dénotée par $h = Conclusion(A)$.

Puisque la base $\Sigma$ est inconsistante, $A(\Sigma)$, l'ensemble de tous les arguments construits à partir de $\Sigma$ ,contient des arguments qui attaquent d'autres arguments.

**Définition 2**

Soit $A_1$ et $A_2$ deux arguments de $A(\Sigma)$ , $A_1$ attaque $A_2$ si $\exists h \in Support(A_2)$

Te que $h \equiv \neg Conclusion(A_1)$

En d'autres termes, un argument est attaqué s'il existe un argument pour la négation d'un élément de son support.

Chaque préordre $\succ\succ_i$ défini sur la base $\Sigma$ peut être utilisé pour définir un préordre sur l'ensemble des arguments $A(\Sigma)$.

On peut ainsi définir une relation de préférence $Perf_i$, dans un contexte $c_i$ ,basée sur le préordre $\succ\succ_i$.

Dans [105] plusieurs relations de préférences entre arguments ont été proposées. Certaines d'entre elles supposent $\succ\succ_i$ est un préordre partiel et d'autres le supposent total. Dans ce qui suit, nous présentons un exemple d'une relation de préférences qui utilise un préordre total.

Lorsque le préordre est total, il est équivalent de considérer la base comme étant stratifiée en plusieurs sous bases $\Sigma_1,....,\Sigma_n$.les éléments de $\Sigma_i$ ont tout la même préférence et ils sont préférés à tout élément d'une sous base $\Sigma_j$ avec $j \succ i$.

Le niveau de préférence d'un sous-ensemble non vide $H$ de $\Sigma$, $niveau(H)$, est le numéro de la plus basse state rencontrée par $H$.

***Définition 3***

Soient $A_1$ et $A_2$ deux arguments de $A(\Sigma)$, $A_1$ est préféré à $A_2$ selon $Perf_i$ ssi $niveau(Support(A_1)) \leq niveau(Support(A_2))$.

$\Pr ef_1,..,\Pr ef_n$ Dénotent les différentes relations de préférences entre les arguments induites respectivement à partir des préordres $\succ\succ_1,....,\succ\succ_n$.

Notons que puisque les préordres sur $\Sigma$ sont conflictuels,les préordres sur $A(\Sigma)$ peuvent aussi être conflictuels,ainsi,pour deux arguments $A_1$ et $A_2$, $A_1$ peut être préfère à $A_2$ dans le contexte $c_i$ et $A_2$ dans le contexte $c_i$ et $A_2$ peut être préféré à $A_1$ dans un autres contexte $c_j$ tel que $i \neq j$.

Nous pouvons maintenant définir formellement le système d'argumentation que nous utiliserons dans la suite.

***Définition 4***

Un système d'argumentation basé sur des préférences contextuelles (SAPC) est un tuple $(A(\Sigma), Attaque, C, \supset, \succ\succ_1,..., \succ\succ_n)$ Tel que :

- $A(\Sigma)$ est l'ensemble des arguments ;
- $Attaque$ est une relation de contrariété entre les arguments ;
- $C$ est l'ensemble des contextes ;
- $\supset$ est un préordre (partiel ou total) sur $C \times C$ ;
- $\succ\succ_i$ est un préordre (partiel ou total) sur $\Sigma \times \Sigma$ issu de contexte $c_i$

Les différents préordres $\Pr ef_1,..., \Pr ef_n$ permettent de distinguer différents types de relations entre les arguments et cela en fonction de la façon dont les arguments s'attaquent mutuellement. Un argument se *défend* seul contre un attaquant s'il est préféré à ce dernier

(dans un contexte préféré). Un ensemble d'arguments peut *défendre* un argument en attaquant tous les arguments contre lesquels le seul argument ne peut pas se défendre seul.

***Définition 5***

Soit $A_1$ et $A_2$ deux arguments de $A(\Sigma)$.

-si $A_2$ attaque $A_1$ alors $A_1$ se défend seul contre $A_2$ ssi $\exists c_i \in C$ tel que :

1. $A_1 \Pr ef_i A_2$
2. $\forall c_j$ tel que $A_2 \Pr ef_j A_1$ alors $c_i \supset c_j$.

-un ensemble d'arguments $S$ défend $A$ ssi $\forall B$ attaque $A$ et $A$ ne se défend pas seul contre $B$

Alors $\exists C \in S$ tel que $C$ attaque $B$ et $B$ ne se défend pas seul contre $C$.

$C_{Attaque,\supset}$ Contient les arguments non attaqués et ceux qui se défendent seuls contre leurs attaquants.

L'ensemble $\underline{S}$ [105] de tous les arguments acceptables du système d'argumentation $(A(\Sigma), Attaque, C, \supset, \succ\succ_1, ..., \succ\succ_n)$ est le plus petit point fixe de la fonction $F$ définie comme suit : $S \subseteq A(\Sigma), F(S) = \{A \in A(\Sigma), Aest\_défend\_parS\}$

***Définition 6***

L'ensemble des arguments acceptables du système d'argumentation $(A(\Sigma), Attaque, C, \supset, \succ\succ_1, ..., \succ\succ_n)$ est :

$\underline{S = \cup F_{i\geq 0}(\varnothing) = C_{Attaque,\supset} \cup \left[\cup F_{i\geq 1}(C_{Attaque,\supset})\right]}$ Un argument est acceptable si et seulement s'il est membre de l'ensemble $\underline{S}$

# 4. La négociation

Il y a probablement autant de définitions similaires de la négociation que de chercheurs dans ce domaine. L'une des plus basiques et succinctes est formulée par Bussman et Muller [106] :

"Negotiation is the communication process of a group of agents in order to reach a mutually accepted agreement on some matter"

Tous les chercheurs s'accordent sur la finalité de la négociation, à savoir l'aboutissement à un accord commun satisfaisant.

Mais la négociation est elle-même définit comme un processus. Toute la diversité des recherches en négociation provient de ce mot : processus. La négociation peut donc être vue comme une boîte noire ayant en entrée un conflit et en sortie un accord, dans le meilleur des cas. La recherche sur la négociation consiste donc à étudier les mécanismes de cette boîte noire, pour la rendre transparente.

Cette définition de la négociation a été reprise par Jennings et al [107] :

"Negotiation is the process by which a group of agents come to a mutually acceptable agreement on some matter (. . .) to make proposals, trade options, offer concessions and (hopefully) come to a mutually acceptable agreement."

Ils y indiquent également que le minimum requis pour un agent négociant est la possibilité de faire et de répondre à des propositions et de pouvoir indiquer son insatisfaction avec les propositions qu'ils trouvent inacceptables. Les propositions peuvent être soit faites indépendamment des autres propositions, soit basées sur l'historique de la négociation.

Les auteurs clament également que si les agents peuvent seulement accepter ou refuser les propositions, alors la négociation peut être gourmande en temps et inefficace.

Pour améliorer l'efficacité du processus de négociation, le destinataire de l'offre doit être capable de fournir un feedback plus utile sur les propositions qu'il reçoit. Ce feedback peut prendre la forme d'une critique ou d'une contre-proposition. Grâce à de tels feedbacks, l'initiateur devrait être en position de générer une proposition qui est plus à même de conduire à un accord.

Müller [106] distingue trois éléments fondamentaux dans la négociation :

- **Protocoles de négociation** : c'est l'ensemble de règles qui dirigent l'interaction. Ceci inclut les types de participants permis (exemple : vendeurs et acheteurs), les états de la négociation (exemple : offres, fin de la négociation), les événements qui font passer d'un état à un autre (exemple : offre acceptée, abondons reçu) et les actions valides et acceptables de la part des participants étant donnés un état (exemple : quels messages peuvent êtres envoyés) ;
- **Objets de négociation** : ce sont les objets de la négociation qui, dans certains cas, se limitent au prix par exemple et/ou d'autres alternatives comme la qualité, la quantité, le temps, etc. Les agents négocient un accord satisfaisant selon le protocole de négociation défini à l'avance. Un tel protocole définit les actions possibles sur les objets de négociation (exemple : offres et contre offres) ;

- **Modèles de prise de décision des agents** : chaque agent possède son propre modèle de prise de décision qui peut être plus ou moins complexe selon le protocole, la nature des objets de négociation, et des opérations possibles lors du processus. C'est le modèle de décision qui définit le comportement de l'agent lors de la négociation et qui devrait lui permettre d'atteindre ses objectifs.

## 4.1 Les différentes approches de négociation

Les travaux dans négociation multi agent peuvent être divisés en gros en trois catégories d'approches :

- Approches à base de théorie de jeux.
- Approches à base d'heuristiques.
- Approches à base d'argumentation.

Tous les mécanismes pour négociation ont à leur coeur un échange d'offres. Les agents font offres qu'ils trouvent acceptables et répondent aux offres faites à eux. Par suite de cette exigence, il y a eu beaucoup de travail [107] [108] [109] [110] [111] [112] [113] en fournissant des agents la capacité de tenir de tels dialogues.

Les deux approches à savoir celle basé sur la théorie de jeux et heuristiques sont concentrées principalement sur l'estimation numérique des offres échanges en termes d'utilités.

Ce type d'approches utilise souvent des stratégies heuristiques et n'incorpore pas de mécanismes pour modeler des processus de la persuasion. En effet, ces approches, bien qu'efficace pour trouver des compromis, ne laissez pas beaucoup pièce pour l'échange des arguments et des informations.

Ils se sont concentrés sur l'envoyé des offres et la seule réaction qui peut être faites sur une offre est une contre offre.

Par exemple, ce peut être problématique dans situations où les agents ont limité de l'information au sujet de l'environnement, ou leurs choix rationnels dépendent de ceux d'autres agents.

Une autre limitation importante des deux catégories précitées est que les utilités ou préférences des agents sont assumées habituellement pour être complètement caractérisées avant l'interaction. Donc un agent est assumé pour avoir un mécanisme par lequel il peut répartir et comparer deux propositions.

Finalement dans ces approches, c'est dur de changer l'ensemble d'issues sous négociation, et les buts des agents sont supposés être fixes.

Ce qui distingue la négociation basée sur l'argumentation sur les autres approches est le fait que les offres peuvent être supportées par des arguments, d'une façon générale, sont des explications pourquoi l'offre a été faite.

Cela permet une plus grande flexibilité que dans autres schémas de négociation, par exemple, il le rend possible de persuader des agents, de changer leur vue d'une offre en introduisant de nouveaux facteurs dans la négociation (de même qu'un vendeur de voiture peut ajouter une assurance gratuite pour conclure une affaire).

En effet, une offre supportée par un bon argument a une meilleure chance d'être acceptée par un agent et aussi peuvent mener à réviser ses buts.

## 4.2 Système de négociation base sur l'argumentation

Pour construire de tels systèmes, on devrait spécifier les éléments suivants :

- Un ensemble d'agents à impliquer dans une négociation avec quelques capacités du raisonnement.
- Une langue de communication.
- Un protocole du dialogue.
- Un ensemble de stratégies.

Un des éléments de base dans un système de la négociation est l'ensemble d'agents impliqué dans la négociation. Les agents peuvent être humains ou systèmes informatiques et la négociation à lieu entre au moins deux agents.

Chacun des agents négociateurs a des états mentaux (croyances, buts, etc.) et quelques capacités du raisonnement. Les capacités de raisonnement, signifient qu'un agent peut :

- Produisez des arguments sur ces états mentaux et évaluer ces arguments. Ils seront connus sous le nom de règles de l'argumentation.
- Peut prendre des décisions : Pendant un processus négociation, un agent doit accomplir les trois étapes de décision suivantes :
- Sélectionnez le contenu d'une locution nécessaire.
- Décidez quand une locution donnée peut être avancée.
- Choisissez la locution suivante parmi toues possibles.

Donc, un agent devrait être équipé des règles de décision pour réviser ses croyances ou buts pendant le processus de négociation. C'est alors important de l'équiper de quelques règles de la révision.

Les agents hétérogènes engagent dans un dialogue de négociation en utilisant un langage de communication (ACL).

Un autre élément pertinent dans un système de négociation est le protocole de dialogue.

C'est l'ensemble de règles qui gouvernent le comportement de haut niveau les interactions entre les différents agents du système, il définit :

- Le type admissible de participants.
- L'état de dialogue.
- Les mouvements admissibles (actes de la parole).
- Les réponses admises à chaque acte, cette partie du protocole peut être définie comme une fonction quelles retourner de dernier acte avancer l'ensemble de prochains actes possibles.

Formellement, un protocole du dialogue spécifie les règles d'interactions, les différentes réponses possibles après un mouvement reçu par un autre agent, mais les réponses exactes qu'un agent propose sont le résultat de la stratégie. D'où, la stratégie dépend du protocole choisi.

# 5 Quelques systèmes de dialogue basé sur l'Argumentation

Les systèmes du dialogue sont destinés pour supporter ou produire des interactions entre deux ou plusieurs participants indépendants. Ces participants peuvent être humains ou des machines, bien que les dialogues puissent être entre un ou plusieurs humains et un ou plusieurs agents logiciels artificiels ou entre des agents logiciels.

Des disciplines différentes ont concentré sur différents types de dialogue. Ainsi, le dialogue de recherche d'information a été étudié par les linguistes [114].

Également, le dialogue de persuasion [115] a été étudié et modelé formellement en Philosophie ; Il a été aussi le centre de beaucoup d'attention dans l'IA et la loi, et l'apprentissage assisté par ordinateur.

La négociation a été longtemps étudiée par économistes et psychologues sociaux [116].

Plusieurs systèmes de dialogues ont été construits pour différents types de domaines, la plupart des attentions jusqu'à présent ont été prêtée aux systèmes pour supporter les dialogues de persuasion.

De plus, pendant que le dialogue de négociation est très important dans les sciences humaines, les économistes se sont intéressés seulement aux issues informatiques concernent les interactions économiques [117]. Quelle que soit la raison, la majorité de systèmes du dialogue s'adressent à supporter le dialogue de persuasion. Parmi ces systèmes, on trouve :

## 5.1 Le système Artikis, Sergot et Pitt

Permet la vérification de systèmes de dialogue gouvernés par les normes.

Artikis, avec Sergot et Pitt de Collège Impérial de Londres ont visé des Framework exécutables qui peuvent être utilisées dans le contexte des systèmes de dialogue [10, 118].

Artikis s'intéresse généralement aux systèmes multi-agents ouverts, signifiant qu'aucune supposition n'est faite au sujet des mécanismes internes des agents, ni sur le sujet de leurs comportement.

Le système comprend dans son ensemble certaines règles sociales et des notions comme : L'autorisation, l'obligation et le pouvoir institutionnel.

Il est possible que les agents violent les règles sociales après lesquelles des sanctions puissent le suivre.

Un système de dialogue peut être vu comme un cas particulier d'un système multi-agents ouvert ; Il spécifie quelles actions sont légales ou autorisées, sans imposer nécessairement aucune restriction sur les mécanismes internes des agents individuels.

En particulier, Artikis se concentre sur la spécification de l'interprétation de Brewka de la théorie de la discussion formelle de Rescher qui se réfère par RTFD *.

Dans RTFD*, trois parties sont actifs : un proposant, un opposant et un déterminant.

Le dernier a pour tâche de déterminer le gagnant de dialogue. Les actes disponibles sont :

- Réclame : Avec laquelle un parti introduit une nouvelle proposition qui doit être défendue.
- Concède : avec lequel un parti est d'accord avec une proposition de la contrepartie.
- Rétracte : avec lequel un parti déclare qu'il ne s'est plus engagé à une certaine proposition.
- Refuser : avec lequel un parti indique que ce n'est pas d'accord avec la déclaration l'autre partie s'est engagé.

- Désapprouve : avec lequel un parti désapprouve contre l'acte antérieur de l'autre partie qu'il voit comme illégal, en signifiant que l'autre partie n'ait pas le pouvoir institutionnel exigé pour ces locutions.
- Déclarer : avec lequel le déterminant déclare un vainqueur du dialogue. Remarquez que c'est le seul acte de la parole disponible au déterminant.

Une des contributions principales d'Artikis dans le domaine de systèmes du dialogue est qu'il exprime les principes de RTFD * dans action C+ [119] [120], qui avoir comme résultat un nombre de règles concernant par exemple, les conséquences des mouvements du dialogue en termes d'engagements, ou même la possibilité de sanctions quand un mouvement du dialogue n'a pas été permis.

La faiblesse principale du système est que le langage C+ exige que les actions exécutables puissent dépendre seulement de l'état courant et pas sur le chemin ou histoire par laquelle l'état a été atteint. Beaucoup de protocoles de l'argumentation ne satisfont pas cette limitation.

## 5.2 Le système Homey

HOMEY (une maison Dirige à travers un Système du Dialogue Intelligent, IST-2001-32434) a été développé dans le centre de recherche de cancer au Royaume-Uni (CRUK) comme une partie de projet de l'UE [9].

La première application du système est équipée d'un synthétiseur de voix, et il est destiné pour usage de dialogue d'un Médecin généraliste pour déterminer si un malade suspect avec le cancer du sein devrait être se reporter à un spécialiste.

Donc, le système du dialogue a besoin d'être intégré attentivement avec la connaissance de domaine médicale (cancer du sein) sur la forme d'une ontologie, et connaissance de tâches et processus cliniques, dans ce cas le langage de spécification du processus Proforma [121]

Pour autoriser l'intégration avec ces domaines de représentations, le modèle du dialogue est divisé en deux représentations de haute et de bas niveau.

La représentation de bas niveau définit un réseau d'état fini des actes de la communication représentée par une voiceXML spécification.

La représentation de haut niveau capture des informations concernant les structures intentionnelles et informationnelles sous le dialogue, avec son état attentionnel courant.

L'information dans la représentation de haut niveau est dérivée de la spécification du domaine sous-jacent, avec la structure intentionnelle qui dérive des décisions, des plans et

des autres tâches de voie de soin médical Proforma et la structure informationnelle qui dérive de l'ontologie médicale.

Pour utiliser ces représentations dans un système pratique, une architecture multi niveau similaire à une architecture d'agent hybride de 3 couches [122] doit être employé.

La couche délibérative est fournie par un directeur du domaine qui crée une spécification de la tâche abstraite (ATS) basé sur les productions de l'exécution du plan et moteurs de l'ontologie.

La couche de la mise en séquence inclut un moteur du jeu qui détermine le jeu de conversation [123] pour compléter les tâches dans l'ATS. Le moteur du jeu utilise la connaissance ontologique pour commander de nouveau le jeu.

La spécification du dialogue de haut niveau résultant (HLDS) est utilisée par un Moteur du Mouvement pour produire la séquence d'actes communicatifs de bas niveau (mouvements), cela peut être fait par l'un et l'autre participant au point courant dans le dialogue.

Les tâches d'enquête de Proforma engendrent le jeu de 'Questionner', tâches d'action pour 'Informer' ou 'Ordonner' le jeu, et les tâches de la décision pour 'Propose'.

On remarque que la décision dans le système est basée sur le modèle argumentative Proforma pour produire les actions et les arguments qu'elles justifient.

Cependant, l'argumentation est relativement superficielle, peu profonde, afin de faciliter une argumentation imbriquée, donc Proforma doit implémente l'argumentation de manière plus complète et conforme à la logique d'Argumentation définie dans [124].

## 5.3 Les systèmes Tutoring de Leeds Metropolitan

Un groupe de chercheurs à l'université métropolitaine de Leeds au Royaume-Uni mené par David Moore a implanté plusieurs systèmes de dialogue homme-mchine pour l'éducation intelligente et l'enseignement assisté par ordinateur.

La motivation pour la recherche a été le développement des systèmes informatiques pour supporter des méthodes d'enseignement à base de dialogue et de débat.

Ces systèmes sont développés à partir du jeu de MacKenzie DC [125] et puis un système alternatif (DE) pour améliorer DC et prévenir des arguments fallacieux et les erreurs communes [126]. Dans le modèle DE, la représentation de la connaissance sous-jacente est sous la forme des propositions orientées objet.

La structure des arguments résultants de l'usage des participants de module DE tant que l'usage est légal. Les arguments en concurrence sont modelés comme des propositions opposantes. Le modèle ne mesure pas la force des arguments.

Le but du dialogue est de résolu un conflit d'opinion, deux participants sont impliqués : l'ordinateur et l'utilisateur où l'ordinateur peut adopter soit un rôle de partisan ou d'un adversaire.

L'ordinateur et l'utilisateur adoptent des rôles opposés dans le dialogue et ils essayent de convertir la vue des unes aux autres. Les rôles n'influenceront pas le processus de l'argumentation parce que le dialogue est symétrique.

Le profil du système est modelé dans ce cas comme une configuration partielle d'un agent.

Le protocole implique les règles suivantes : les participants font des actes à tour de rôle, les questions doivent toujours être répondues, et les demandes doivent être défendues (ou retirées) s'ils ont défié légitimement.

Les réponses alternatives aux mêmes actes sont permises librement. Autoriser seulement un coup par tour peut diminuer l'exigibilité du dialogue ; cela peut augmenter cependant l'interactivité et l'équitabilité du jeu du dialogue.

Les participants ne sont pas autorisés à changer le protocole du jeu.

Un parti perdra le débat quand sa thèse sera enlevée de sa base et la thèse de l'adversaire est ajoutée à sa base. Il peut se terminer aussi si les deux parties sortir hors de stratégie et un vainqueur n'est pas encore décidé, l'arbitre terminera le jeu (de cet effet, le dialogue termine dans l'impasse).

Un prototype homme-mchine qui utilise le modèle de dialogue DE a été construit, et le prototype a été évalué par des techniques heuristiques.

## 5.4 Le système PARMA

Le modèle de PARMA a été développé à l'Université de Liverpool [127] [128], permet à un proposant de déclarer et justifie une proposition pour entreprendre une action, et pour un adversaire d'offrir des critiques pour telle proposition.

Le modèle adresse le dialogue de persuasion pour l'action, par identification des composants d'une proposition et ses justifications, et par l'élaboration de tous les chemins possibles dans lesquels une telle proposition peut être contestée.

Le modèle de PARMA donne un compte détaillé de raisonnement pratique comme justification présomptive avec un ensemble de questions critiques, un compte qui subsume le compte le plus limité de raisonnement pratique de Walton [129].

Le jeu de dialogue PARMA (PARMA DG) est implémenté comme un de jeu de dialogue basé sur ordinateur sur le modèle de PARMA [130]. PARMA DG autorise à deux participants humains de faire une proposition pour l'action ; conteste cette proposition ; et justifier ou se rétracter la proposition originale, tout conformément au modèle PARMA de persuasion sur action.

Le système lui-même jouer seulement un rôle limité dans le processus de dialogue, il assure que toutes les déclarations font par l'un ou l'autre participants sont légaux, stocké tous les engagements, et de traqués l'histoire de dialogue.

Le système n'agit pas comme médiateur dans une discussion, il ne produit pas aussi des déclarations ou déclarations possibles pour les participants. En conséquence, les utilisateurs humains ont besoin d'être familiers avec le modèle de PARMA.

Parce que le modèle de PARMA fournir un grand nombre d'attaques possibles sur une proposition pour action, les participants peuvent trouver qu'il est difficile de déterminer leurs stratégies dans un dialogue particulier.

# 6 Conclusion

Au long de ce chapitre, nous avons essayé d'éclaircir la notion de système de dialogue multi-agents en présentant certaines définitions. Nous avons montré les différents types de dialogue, les principales approches utilisées pour leurs constructions tout on focalise sur l'argumentation. Apres, nous avons illustré un type particulier de dialogue ; la négociation. Finalement, nous avons conclu en présentant quelques implémentations de systèmes de dialogue à base d'argumentation.

La conclusion principale de l'étude des systèmes de dialogues, c'est que l'attention jusqu'à maintenait s'est concentrée sur des systèmes pour supporter ou pour produire de dialogues de persuasion. Pour d'autres applications dans les domaines tels que le commerce électronique, l'attention aura besoin d'être donné aux systèmes pour supporter des autres types d'interaction.

# *Chapitre IV*
# *Architecture d'un Système de Négociation*

## 1 Introduction

Dans le chapitre précèdent, nous avons vu les systèmes de dialogue et l'apport de l'argumentation pour leurs constructions, avec quelques systèmes de dialogue construire pour des domaines différents.

Dans ce chapitre, en plus d'orientation de notre travail vers le dialogue de négociation dans le domaine de commerce électronique, nous discutons sur les caractéristiques et les inconvénients de ses systèmes ce qui nous permet de dégager les caractéristiques que doit possède notre système. Puis une architecture qu'il concrétise. Pour cela, ce chapitre est organisé comme suit :

La première partie présente une synthèse concerne les systèmes de dialogues existants, puis une illustration de quelques caractéristiques architecturales que doit posséder notre système, ce qui nous conduit à faire sortir les principales composantes et d'en déduire les fonctionnalités appropriées. Le tout donnera naissance à une nouvelle architecture d'un système de dialogue argumentative multi protocole pour la négociation automatique entre deux agents.

Nous concluons le chapitre par une deuxième partie où nous présentons une étude de cas.

## 2 Synthèses des systèmes de dialogue existants

Dans le chapitre précèdent, nous avons vu qu'au cours de dernières années un nombre croissant de systèmes de dialogue basés sur l'argumentation ont été élaborés et présentés pour différents types de disciplines à savoir le linguiste, AI, le domaine juridique, l'enseignement assisté par ordinateur, l'économie, le domaine médical, etc.

La première remarque est que ces systèmes sont centralisés dans le sens qu'ils adoptent un agent médiateur comme les systèmes Artikis [10], PARMA [130].

En plus, la plupart des systèmes de dialogue conçu jusqu'à présent montrent une prédominance d'un type particulier de dialogue qu'est la persuasion.

Par exemple le système TRAINS [131] a été conçu pour fournir à un homme décideur un support informatique homme-mchine intelligent pour la délibération sur la sélection d'un plan d'un train de transport. Le système PARMA est une implémentation d'une médiation de dialogue de persuasion entre deux personnes sur une série d'actions , le prototype

HOMEY [11] qui un système base sur la synthèse de parole dans laquelle des agents assistants s'engagent dans un dialogue avec une médecine générale pour décider sur une série d'action pour un patient.

On trouve aussi le système Artikis [10] un prototype d'un système de dialogue de persuasion dans laquelle un déterminant médiat l'échanger des locutions entre un agent proposant et un autre agent opposant pour de déterminer le gagnant le dialogue.

Vu l'émergence de l'Internet et ces applications comme le commerce électronique, l'attention doit être donnée pour d'autres types de dialogue mentionnent dans la topologie de Walton et Krabbe et surtout la négociation.

Dans l'autre côté, ces systèmes de dialogue supportent un seul protocole, plus précisément, le protocole est code implicitement dans l'agent et suppose connue à l'avance. Ce type de systèmes est fermé dans le sens que le protocole de dialogue et la stratégie sont prédéterminés.

Mais les avances en technologies ont poussé vers des systèmes de dialogues multi protocoles. Ces systèmes sont flexibles et dynamiques ; les règles de dialogue peuvent changer selon le type de protocole adapté.

Etant donnée ces exigences nous proposant un système de dialogue multi-agents pour la négociation, décentralisé, et qui soit multi protocoles.

Un agent n'est obligé d'engager dans un seul protocole de négociation, mais il est capable de choisir le protocole de négociation approprié au type d'interaction dont il participe.

Donc, dans notre système le protocole de la négociation n'est pas codé implicitement dans l'agent, mais les agents négociants choisissent le type de protocole qui règle leurs interactions. Ces protocoles de négociations sont décrits dans une ontologie partagée.

L'avantage est que les agents n'ont plus besoin d'aller hors linge pour reprogrammer pour supporter d'autres types de négociations.

Le tableau suivant résume les propriétés de ces systèmes :

| | Type de dialogue | Agent médiateur | Ontologie des protocoles |
|---|---|---|---|
| Artikis [10] | persuasion | oui | non |
| PARMA [128] | persuasion | oui | non |
| TRAINS [129] | persuasion | oui | non |
| **Notre système** | **négociation** | **non** | **oui** |
| HOMEY [11] | délibération | non | non |

**Tableau 4.1 : classification des systèmes de dialogue.**

# 3 Besoins pour une architecture multi agent

Dans cette section nous introduisons les caractéristiques que doit possède une architecture multi agents d'un système de dialogue basé sur l'argumentation.

## 3.1 Caractéristique architecturale

Le terme architecture est généralement utilisé pour d'écrire l'organisation interne d'un agent, par analogie avec la structure des ordinateurs [11].

Mais, cette comparaison n'en est pas de même pour les architectures multi-agents qui présentent un éventail très riche de solutions pour concevoir aussi bien l'organisation interne que celle externe des agents et d'en exhiber le comportement total de system.

Dans notre problématique qui est la proposition d'un système de dialogue multi agent pour la négociation, et le développement d'une architecture qui concrétisera notre système, il est nécessaire d'analyser et de considérer les caractéristiques suivantes.

### 3.1.1 Analyse structurale

Le besoin de l'analyse structurale surgit de fait de deux motivations :

1-quels sont les agents à considérer pour une situation de négociation, ce qui implique l'identification des rôles et des fonctions caractérisent ces agents. Il est tout à fait naturel que nos agents sont des agents coopératifs dotés de capacités cognitives relatives aux systèmes multi agents. Ajoutés à cela, leur particularité à traiter des problèmes conflictuels.

2-quels sont les types de relations entre les agents, dans notre cas c'est la coopération pour la résolution de conflits.

### 3.1.2 Structures de subordination

Nous envisageons une architecture distribuée selon un niveau de subordination égalitaire (les deux agents de notre architecture sont de même niveau).

### 3.1.3 Structures constitutionnelles

En découpant les différents aspects des agents en se basant sur l'approche basée sur les composantes pour leurs constructions.

Une fois nous avons identifié les composants du système multi agents, il s'avère utile de faire un zoom en profondeur pour désigner les différents composants des agents eux-mêmes. Sous cette optique, nous avons doté chaque agent d'un ensemble de composants.

Ces derniers assurent la fonctionnalité principale de l'agent dans l'architecture. Ce qui le rend similaire aux architectures modulaires horizontales qui sont considérées comme un assemblage de modules, chacun réalise une fonction horizontale particulière.

L'agent dans notre architecture doit comporte :

**La connaissance**

Les agents doivent avoir de la connaissance et des règles pertinentes pour échanger ces informations. Ces dernières sont stockées dans une base de données locale de l'agent. Ces connaissances sont accessibles seulement à l'agent qu'il possède et ne sera pas partagé entre les agents.

En plus, les agents doivent avoir de la connaissance des conséquences possibles de l'effet de leurs actes pour la réalisation de leurs objectifs.

Finalement, pour représenter la connaissance de l'agent et d'être capable du partage si nécessaire, l'agent doit avoir une ontologie du domaine.

**Le raisonnement :**

Comme plusieurs domaines d'application à savoir le commerce électronique, la plupart des connaissances disponibles sont annulables (defeasible), l'interaction entre agents implique souvent l'argumentation. Par conséquent, les agents devraient être capables de générer et évaluer des arguments pour et contre certaines demandes. Finalement, les agents doivent être capables de faire un raisonnement hypothétique pour générer des offres conditionnelles.

**Les objectifs de l'agent**

Comme décrits ci-dessus, les agents dans notre problème ont des objectifs individuels. Dans notre cas, l'agent a deux rôles :

Demandeur d'un service, et un Fournisseur de ce service.

De plus les deux agents veulent contribuer aux objectifs globaux de système en termes d'échange optimal des informations. Les agents et leurs interactions devraient être conçus dans une telle manière que leur comportement est conformé avec leurs objectifs.

**La communication**

Les agents devraient être capables d'échanger des informations mais aussi d'autres types d'interactions devraient être aussi possibles.

Au-dessus, nous avons noté que l'objectif de l'agent qui possède le service est d'examiner les conditions sous lesquelles il peut donner les informations.

En plus, les agents doivent être capables de négocier avec d'autre. Par conséquent, les agents doivent être capables de prendre part aux dialogues de la négociation pour atteindre une meilleure offre. Pour valider de telles interactions, un protocole de dialogue convenable doit être implémenté.

Des politiques doivent aussi être donnés, aux agents, ou tactiques, pour régler leurs comportements dans les dialogues. Ces politiques devraient être conçues pour avancer les objectifs de l'agent. Depuis que ces objectifs incluent ceux-là de l'institution totale de système, les politiques des agents devraient induire un degré clair de coopération.

## 4. Un système de dialogue multi-agents pour la négociation

Nous avons développé une architecture qui est destinée à supporter notre système. Elle prend en compte les caractéristiques mentionnées précédemment. Dans le but de la faire projeter vers des fins d'implémentation d'une part et d'autre part pour la faire rapprocher de l'utilisateur, nous avons conçue chaque agent pour la présentation d'un utilisateur via un agent d'interface.

Donc, les utilisateurs sont délégués par les agents de l'architecture pour réaliser la négociation. Lors de l'apparition des conflits, ils seront seulement informés par le système via leurs agents d'interface, et n'interviennent qu'au moment où les agents leur demandent d'effectuer des choix sur des préférences ou pour faire entrer des nouvelles informations.

Nous classifions les constituants de notre système en trois éléments :

1. Les agents d'interface.
2. Les agents en situation de dialogue de négociation.
3. L'ontologie de protocoles de négociation argumentative.

Les différents éléments sont reliés par des flux d'informations et de contrôles. L'architecture du système et ces éléments sont présentés dans la figure 4.1.

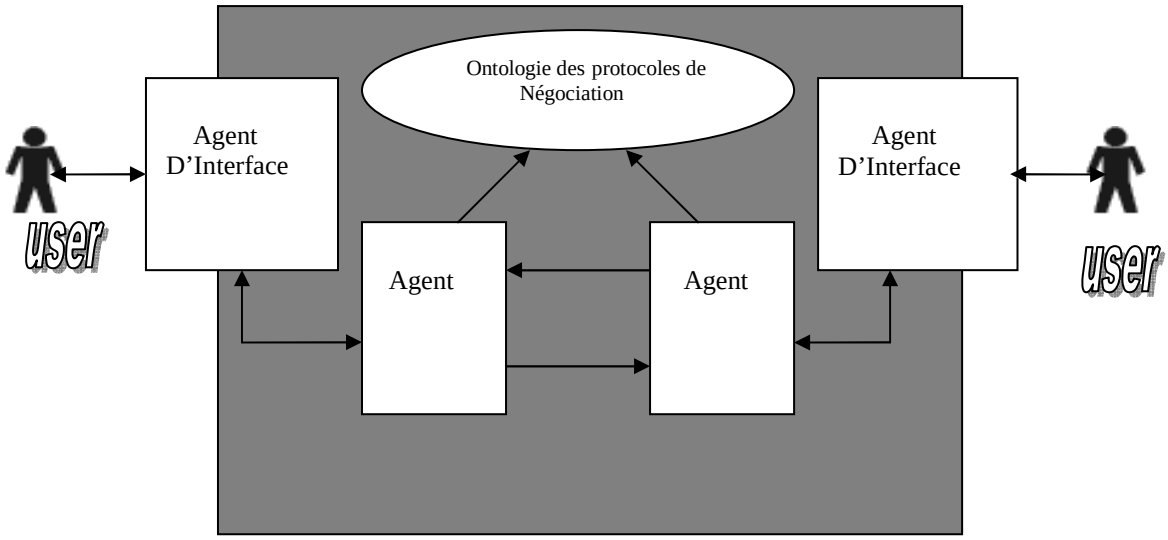


**Figure 4.1 : architecture d'un système de dialogue pour la négociation.**

## 4.1 L'agent d'interface

L'agent d'interface sert à représenter l'utilisateur que se soit un acheteur ou un vendeur. Il sert comme un médiateur entre l'utilisateur et l'agent.

L'utilisateur (par exemple un acheteur) communique avec l'agent d'interface via une interface graphique, par contre la communication entre l'agent d'interface et l'agent de système se fait par envoi de messages ACL.

Le rôle de l'agent d'interface est d'informer l'utilisateur de l'état de dialogue, et de transformer les actions de l'utilisateur vers des messages FIPA ACL et vice et versa.

Quand un agent d'interface reçoit un message FIPA de la part de l'agent de système, il converti ce message sous forme graphique pour informer l'utilisateur de changement dans l'état de négociation.

Aussi, un agent ne peut pas décider pour entreprendre certaines actions pour différentes raisons, et pour cela il demande à l'utilisateur via l'agent d'interface de prendre l'initiative pour exécuter ces actions ou d'introduire des nouvelles connaissances pour répondre à cette situation.

## 4.2 Les agents négociants

Nous supposons que les agents négociants sont des agents homogènes c'est-à-dire qu'ils comportent le même type de composants. Dans cette section nous schématisons l'architecture interne des agents.

Nos agents sont cognitifs dotés de croyances, de désires et d'intentions stockées dans la base de moteur d'inférence, ce qui constitue la partie interne responsable de la prise de décision. Quant à la partie responsable de la communication avec l'utilisateur, elle est réalisée par un composant d'interface utilisateur. L'interaction avec l'autre agent est réalisée par le composant de communication avec l'agent.

Vu le but principal du système, la négociation est assurée par le protocole argumentatif et le moteur d'inférence argumentatif et les protocoles de négociation décrits dans l'ontologie partagée et le module de protocole de négociation qui sert à héberger le protocole de négociation.

Tous ces modules ou composants sont contrôlés par un autre composant qui est le gestionnaire de dialogue. L'architecture d'un agent est présentée dans la figure 4.2.

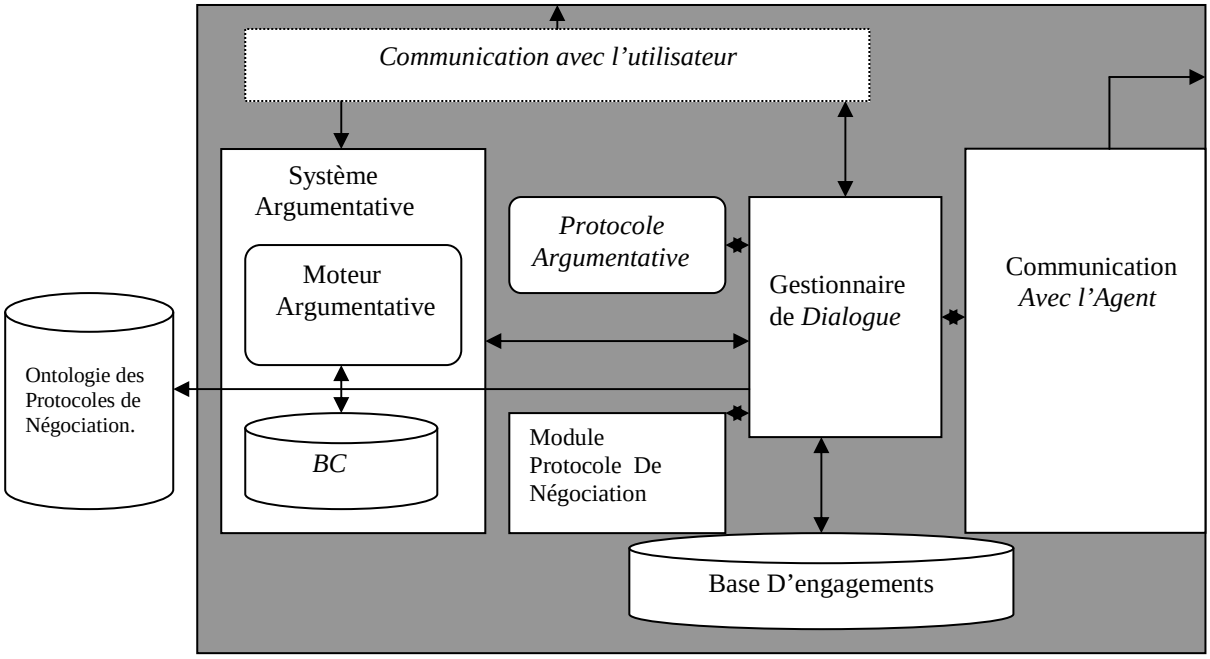


**Figure 4.2 : Description d'un agent négociant.**

### 4.2.1 Module de communication inter-agent

Ce composant est responsable de l'émission et la réception des messages entre les agents de système, ces deux agents communiquent par le langage FIPA ACL.

Quand un agent reçoit un message de l'autre agent, il le décompose et enlève l'en-tête de ce message et envoi le contenu vers le gestionnaire de dialogue et vise et versa, c'est-à-dire recevoir un message de gestionnaire de dialogue, compose et l'envoi vers l'autre agent.

### 4.2.2 Module d'interface avec l'utilisateur

Ce module a pour rôle de communiquer avec l'agent d'interface, il reçoit des messages de ce dernier, les décompose et les envoie vers le gestionnaire de dialogue. Dans l'autre cote, il reçoit les messages de gestionnaire de dialogue, les transforme et les envoie vers l'agent d'interface pour informer sur l'état de dialogue. Il permet aussi la consultation et la mise à jour de la base de connaissances de système argumentatif.

### 4.2.3 La base d'engagements

On appel tableaux d'engagements (commitment stores) les structures de données qui répertorient les engagements pris par les participants au cours du dialogue.
Les tableaux d'engagements sont mis à jour en fonction des locutions émises afin d'enregistrer méticuleusement toutes les déclarations.
Ces informations sont publiques et externes au dialogue, par exemple un engagement pour vendre une voiture.

### 4.2.4 Le module de protocole de négociation

Une structure qui sert d'acquérir le protocole de négociation à partir de l'ontologie des protocoles de négociation, cette dernière supporte plusieurs protocoles de négociation, mais un seul protocole est téléchargé de l'ontologie à la fois durant le processus de dialogue.

### 4.2.5 Le système argumentatif

Nos agents sont capables d'exécuter un raisonnement annulable basé sur l'argumentation avec leurs connaissances internes. En particulier, nous supposons qu'ils sont capables de construire des arguments pour certaines propositions et alors vérifier si ces arguments sont justifiés, défendable ou rejetable. Par conséquent, on a besoin d'un système d'argumentation.

Le système d'argumentation consiste à un moteur d'inférence argumentatif et d'une base de connaissance. Cette dernière contient des informations de domaine de l'agent, ses buts et leurs stratégies.

Ces informations sont exprimées dans une logique argumentative, cette base est locale et n'est pas accessible à l'autre agent du système.

La connaissance est représentée sous la forme des langages à base des règles et les arguments sont construits par le chaînage de ces règles en arbres.

Le système argumentatif est un moteur d'inférence basé sur la logique argumentative non monotone, nous avons présenté dans le chapitre précédent un système argumentatif abstrait, et nous verrons dans le chapitre suivant une implémentation d'un système argumentative (Gorgias).

### 4.2.6 Le protocole argumentatif

Nous avons dis dans les sections précédentes que notre système de négociation est basé sur l'argumentation, et pour cela nous adoptons un protocole d'interaction argumentative nommé Fatio défini par [132].

Nous trouvons le plus approprié pour un dialogue argumentatif, ce protocole : concis, générique permet de supporter plusieurs protocoles d'interaction. Dans notre cas, il permet d'incorporer plusieurs protocoles de négociation. Les actes dans un dialogue d'argumentation sont factuelles, déclaratives, expressives, déclarations du rapport sociales, commissives, directives, déductives et argumentatives.

La syntaxe des actes sera : $illocution(P_i, f)$ ou $illocution(P_i, P_j, f)$ où l'$illocution$ est un illocution, $P_i$ est un identificateur de l'agent qui fait la déclaration. $P_j$ dénote un agent vers lequel la déclaration est dirigée, et $f$ est le contenu de la locution.

Nous utilisons le terme obligations dialectiques pour faire référence aux engagements à l'intérieur du dialogue, par exemple, une obligation dans un dialogue de défendre une assertion contre une attaque d'un autre participant.

Nous définissons maintenant les cinq principales locutions légales dans le protocole Fatio :

F1 : $assert(P_i, \phi)$ : Un locateur $P_i$ affirme une proposition $\phi \in C$ (une croyance,une intention, un rapport social,une engagement externe, etc.). Par conséquence, $P_i$ crée une obligation dialectique pour fournir si besoin une justification pour $\phi$ si elle est demande par l'autre participant.

F2 : $question(P_j, P_i, \phi)$ : Un locateur $P_j$ questionne une proposition antérieure envoyée par un autre participant $P_i$ et cherche une justification $\phi$. $P_j$ ne crée pas une obligation dialectique sur lui-même.

F3 : $challenge(P_j, P_i, \phi)$ : $P_j$ défie une déclaration antérieure $assert(P_i, \phi)$ envoie par un autre participant $P_i$ et cherche une justification pour $\phi$.

Contrairement avec la locution précèdent, $P_j$ crée une obligation dialectique sur lui-même pour fournir une justification pour n'affirmer pas $\phi$ , par exemple un argument contre $\phi$.

F4 : $justify(P_i, \Phi \rightarrow^{+} \phi)$: Un locateur $P_i$ qui a envoie $assert(P_i, \phi)$ est questionner ou challenger par l'autre participant ; donc il est capable de fournir une justification $\Phi \in A$ pour la proposition initial $\phi$ . La locution $justify(P_i, \Phi \rightarrow^{-} \phi)$ est définie d'une manière similaire.

F5 : $retract(P_i, \phi)$ : Un locateur $P_i$ qui a envoie $assert(P_i, \phi)$ ou $justify(P_i, \Phi \rightarrow^{+} \phi)$ peut annuler cette locution par l'un des locution $retract(P_i, \phi)$ ou $retract(P_i, \Phi \rightarrow^{+} \phi)$ respectivement. Ceci retire l'obligation dialectique précédente de $P_i$ pour justifier $\phi$ où $\Phi$ est questionne ou challenge.

### 4.2.7 Gestionnaire de Dialogue

le composant central de système de dialogue est responsable de la coordination entre les différents modules de l'agent.

Lorsqu'un message arrive a ce module, que soit de module d'interface avec l'utilisateur ou le module de communication avec l'agent, il répond par l'activation de module approprie et d'envoyer les informations nécessaires.

Le gestionnaire de dialogue assure aussi la cohérence de dialogue, c'est lui qui responsable de commencement de dialogue, et leur terminaison.

Il a pour rôle de lancer le processus de négociation et téléchargement de protocole de négociation argumentative à partir de l'ontologie du système et le chargé au niveau de module protocole de négociation.

Le gestionnaire de dialogue est responsable de mise à jour des différentes bases de l'agent à savoir la base d'engagements et la base de connaissances.

### 4.2.8 Ontologie de protocoles de négociation

L'introduction d'une ontologie dans un système d'information vise à réduire, voire éliminer la confusion conceptuelle et terminologique et à étendre vers une compréhension partagée pour améliorer la communication.

Nous avons vu dans le chapitre deux que pour intégrer une ontologie des protocoles de négociation dans un système, il faut que le système permettre respectivement la spécification de protocoles de négociation, la sélection dynamique des protocoles, et l'instanciation de ces protocoles en vue de leur future exécution. Dans notre système, ces trois opérations sont réalisées par les agents du système.

La figure 3 montre comment les agents de sélection, de protocoles et de lancement de de ces derniers utilisent les ontologies OWL à l'aide de la librairie Jena.

Une fois instanciers, ces protocoles seront stockés dans un répertoire constituant une bibliothèque de protocoles en OWL. L'ontologie de protocoles est stockée dans deux fichiers OWL séparés : le schéma et l'instance.

Le schéma comprend la définition des concepts des relations entre concepts, les contraintes et les axiomes. Les protocoles sont définis d'une manière déclarative et leurs instances sont stockées dans un fichier d'instances accessible à l'ensemble des agents du système.

Lors de la création d'une nouvelle conversation, l'agent sélection de protocole doit rechercher le protocole approprié à cette conversation à partir du fichier des instances. Une fois le protocole est sélectionné, l'agent doit créer une instance de la nouvelle conversation conforme au protocole en question.

Au moment de l'exécution, ces deux fichiers sont traités par un Raisonneur (par exemple Racer ou Pellet). Ce dernier supporte l'ensemble des requêtes pour la sélection dynamique des protocoles de négociation. Ces requêtes peuvent être lancées pour interroger le fichier des instances pour la sélection et le choix dynamique des protocoles. Au niveau de l'exécution, la recherche et la sélection d'un protocole pour la négociation du système de dialogue peuvent être effectuée en prenant en compte l'ensemble des propriétés ou caractéristiques du protocole à mettre en oeuvre. Par conséquent, la recherche d'un protocole est non limitée au critère ; nom du protocole, mais peut prendre en compte toutes les propriétés permettant sa description.

Dans la suite, nous montrerons à travers un exemple l'instanciation d'un protocole à partir d'une ontologie. Ce protocole de négociation argumentative défins dans [109].

Une instance du protocole est définie comme l'ensemble d'instance des concepts permettant sa description.

La description des ontologies se fait d'un langage d'échange OWL. Ce dernier repose sur la logique des descriptions pour décrire les hiérarchies, les axiomes et les contraintes. OWL permet de représenter les classes (owl : class), les liens sémantiques (owl: objectProperty), les types de données (owl : dataTypeProperty) et aussi les cardinalités.

Il supporte aussi les hiérarchies de spécialisation, l'équivalence entre les concepts (owl equivalentClass), l'intersection et l'union entre classes ou concepts (owl : unionof, owl : intersectionof).

Pour cette représentation formelle de l'ontologie, nous avons utilisé le « plugin » OWL de Protégé-2000.

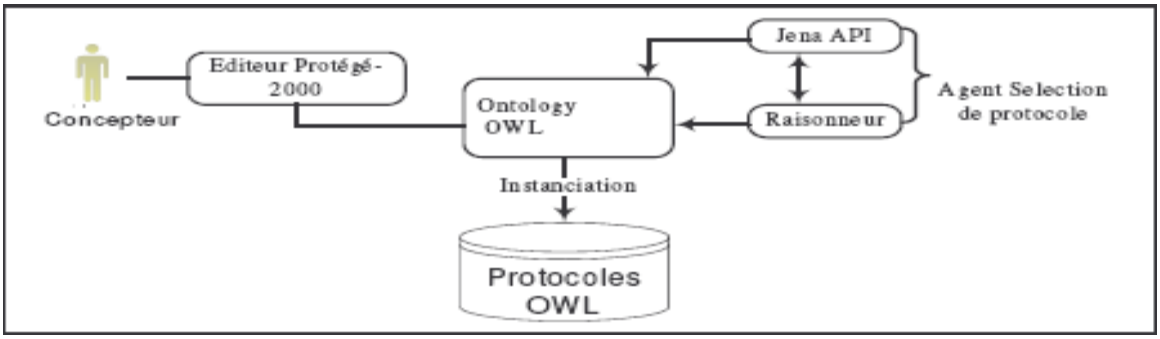


**Figure 4.3 : Architecture logicielle pour l'utilisation de l'ontologie [75].**

### 4.2.8.1 un protocole de négociation argumentative

Le protocole de négociation argumentative que nous adaptent dans notre système est un type particulier des protocole de négociation base sur l'argumentation nommé négociation base sur les intérêts (interest based negotiation, IBN) [109] ; une forme de NBA dans lesquels les agents explorer et discuter leurs intérêts sous-jacent.

Les informations sur les buts des autres agents peut être utilisé dans plusieurs chemin pour améliorer le processus de négociation est ceci par exemple, tel que découvrir et exploiter les buts communs.

Ce protocole est décrit comme suit :

L'agent commence avec une offre O au temps t = 0, à chaque temps t > 0 :

1. Propose (i, offre) : l'agent i propose à l'agent j de négocie l'offre O lequel n'a pas été proposé auparavant;

2. l'agent j soit :

A. Accept (j, O): j accepté l'offre et la négociation terminé.

B. Reject (j, O): l'offre est rejet et la négociation terminée sans accord.

C. Faite une contre-proposition en allant au pas l et en échanger les rôles d'agents i et j au temps suivant.

D. La commutation vers le dialogue basé sur les intérêts (argumentative). On suppose que chaque agent possède un ensemble de buts.

1. why (j, x): j demande de i le but sous-jacent pour l'allocation d'une ressource (offre).
2. i soit:

A. Assert (i, but): i répond en déclarant un but qui est ajouté à buts de l'agent i.

B. decline (i): refuse de donner l'information.

3. j soit :

A. Accept (j, O): j accepte O, si maintenant l'offre est plus favorable.

B. Un nouveau cycle de recherche d'informations en allant au pas D.1

C. j saute son tour, allé au pas 2 avec i qui prend le rôle de J.

la spécification de ce protocole par OWL est laissée au chapitre implémentation.

# 5 Étude de cas : agence de voyages

On suppose qu'une agence de voyages organise des voyages nationaux et internationaux [133].

Le système aide les clients via leur agent pour commander un billet d'avion, réserver un hôtel, allouer une voiture, etc. Donc, l'utilisateur via l'agent d'interface informe son agent au sujet de ce qu'il besoin, et ce dernier traduit cette interaction IHM dans un langage ACL et l'envoie à l'agent, celui-ci va l'envoyer vers l'agent de voyages qui représente l'agence.

Ensuite, l'agent de voyages analyse la demande, consulte les offres qu'il possède et envoie l'un d'entre eux à l'autre agent.

Puis, un dialogue de négociation se poursuit entre le client (leur agent) et l'agent de voyages pour trouver une offre intéressante qui satisfaisait les deux parties.

On suppose que le client et leur agent possèdent des informations incomplètes ou incorrectes.

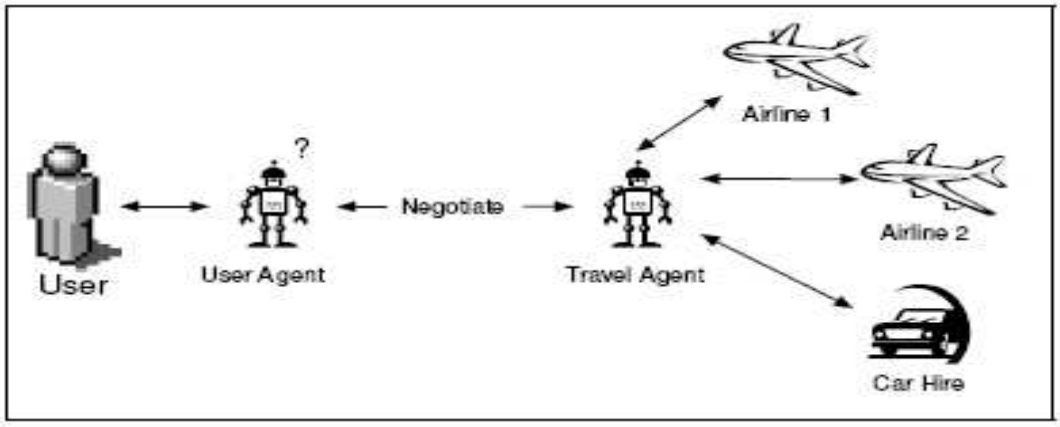


**Figure 4.4 : Un scénario d'un agent de voyages [133].**

## Le scénario

Considère un potentiel dialogue de négociation entre un client A et un agent de voyages V. Supposant que A désire de ne pas payer plus que 200$, et V ne peut pas vendre moins que 400$ pour un billet d'avion de malbourn à Sydney.

On suppose qu'un agent a l'ensemble ($B_b$,$B_d$,$B_p$,D,ID,I;R) tel que:

1. $B_b$ l'ensemble de croyances.
2. $B_d$ l'ensemble de règles pour la génération des désires.
3. $B_p$ l'ensemble de règles de palification.
4. D l'ensemble des désires initialises par $\varnothing$.
5. ID : l'ensemble de désires initialises futur par $\varnothing$.
6. I l'ensemble des intentions initialises de l'agent par $\varnothing$.
7. R $\in$ RES l'ensemble des ressources de l'agent.

On suppose que aussi les prédicats:

aic = « Assiste une conférence en AI à Sydney ».

syd = « voyager a Sydney ».

reg = « payer les frais de conférence ».

rentCar = « cherche a louer un voiture »

ford = « avoir une voiture de la marque Ford »

hire = « louer une voiture »

ticket = «fournir un billets pour Sydney »

book = « vendre un billets »

Déduire les désires **:** Suppose que l'agent client A génère un futur désire IDA = {aic}, et suppose que l'agent de voyages V a un désire non conditionnel pour conduire une transaction IDV = {biz}.

La formation des intentions : Après l'inférence d'un ensemble de désires a partir des croyances contextuels, les agents utiliser les règles de planification pour construire des plans pour achève ces désires.

Suppose que l'agent de voyages V possède la base de connaissances suivante :

1. $B_B^V = \varnothing$
2. $B_D^V = \{\rightarrow biz, 1\}$
3. $B_P^V = \begin{cases} hire \rightarrow biz \\ ticket \rightarrow biz \\ ford \wedge pay400\$ \rightarrow hire \\ book \wedge pay200\$ \rightarrow ticket \\ flysyd \rightarrow syd \\ book \wedge pay200\$ \end{cases}$

Aussi, supposons que l'agent de voyages $V$ possède les ressources $R_V = \{ford, book\}$ et le client possède les ressources $R_A = \{pay200\$, pay100\$, pay400\$\}$.

Dans ce cas, on a une situation de conflit d'intérêts, l'agent A est besoin d'un Ford pour 200$ seulement et l'agent V lui demande 400$ , donc ils s'engagent dans un dialogue de négociation argumentative et ceci par l'échange des propositions et des offres.

Pour raison de complexité de dialogue, la négociation commence par les actes non argumentatifs puis les actes argumentatifs.

Les deux fragments de dialogue montrent deux offres qui ne conduisent pas à un accord.

Supposons que le client via son agent, commence la négociation par la proposition suivante :

1. Client: PROPOSE(A, V, do(V,ford))

   Permettre d'ajouter int(do(V, rentCar) dans la base d'engagement de client.
2. Agent de voyages: PROPOSE(V,A, do(V, ford) ^ do(A, pay$400 ))
3. Client: REJECT(A, V, do(V, ford) ^ do(A, pay$400 ))
4. Agent de voyages : PASS(V)
5. Client: PROPOSE(A, V, do(V, ford)^ do(A, pay$200 ))
6. Agent de voyages: REJECT(V,A, do(V, ford) ^ do(A, pay$200 ))
7. Client : PASS(A)

On remarque que le client n'accepte pas l'offre de l'agent de voyages et l'agent de voyages n'accepte pas la contre-proposition de client.

Le dialogue terminé avec aucun accord parce que les positions des deux parties ne s'accordent pas (le prix d'allocation de la voiture).

Chacun des propositions précédentes crées des obligations dialectiques. Maintenant, nous utilisons les actes argumentatifs de notre protocole.

En utilisant l'argumentation, l'agent de voyages essayera d'influencer la négociation par le biais du plan de client. Dans ce cas, l'agent de voyage demandera au client la motivation derrière le besoin d'une voiture.

Le client répondu de fait qu'il est besoin d'une voiture pour assister une conférence à Sydney.

8. Agent de voyages : REQ-PURPOSE (V,A, ford)

9. Client : ASSERT (A,V, instr({ford}, syd))

Faire connaître cette information, le client donne l'opportunité pour l'agent de voyages pour présent une moyenne alternative et moins chère pour assister la conférence à Sydney, et ceci par l'avion.

10. Agent de voyages : ASSERT(V,A, prule(flysyd $\rightarrow$ syd))

11. Client : PASS(A)

12. Agent de voyages : ASSERT(V,A, prule(book ^ pay\$200 $\rightarrow$ flysyd))

13. Client : PASS(A)

Donc, le client construire un nouveau plan pour assister la conférence à Sydney par l'avion.

14. Agent de voyages : PROPOSE(V,A, do(V, book) ^ do(A, pay\$200 ))

15. Client : ACCEPT (A, V, do(S, book) ^ do(A, pay\$200 ))

Il est bien clair que le protocole de négociation argumentative permettre de trouver un accord en demandant des informations sur les plans de l'autre partie et en exploitant les informations publiques disponibles dans les bases d'engagements pour présenter des arguments qui influence l'autre partie.

Dans le dialogue précèdent, l'agent de voyage a réussi d'obliger le client d'adopter une nouvelle intention (acheter un billet) et qui permettre de trouver un accord alternatif.

# 6. Conclusion

Dans ce chapitre, nous avons construit une architecture multi-agents basée sur l'argumentation pour la négociation automatique.

Nous avons commencé par la spécification des agents de notre système où on a présenté leurs structures internes, ensuite nous avons spécifié l'ontologie des protocoles de négociation que intègre notre système, et finalement on a conclu avec une étude de cas.

Le chapitre suivant est consacré à la validation de notre architecture, avec une simulation dans l'environnement JADE.

# *Chapitre V*
# *L'Implémentation*

## 1 Introduction

Un système multi-agents (SMA) est un ensemble d'agents situé dans un environnement qui communiquent entre eux afin d'atteindre un but programmé.

Un agent est une entité programmée et donc définie par des caractéristiques afin de pouvoir évoluer de façon quasi autonome.

Cette entité peu être matérialisée par un programme, un robot, un processus, etc. Les SMA font l'objet de longues années de recherche en intelligence artificielle.

Les systèmes multi-agents sont conçus de manière générale à l'aide de plate-forme Multi-agents comme celle que nous allons voir : la plate-forme JADE (Java Agent Development framework).

Cette plate-forme doit être compatible avec la norme FIPA (1997) qui rassemble un ensemble de règles permettant à une société d'agents d'interagir entre eux.

Le but de notre travail est d'implémenter à l'aide de JADE, Gorgias, Protége 2000 et Jena une architecture d'un système de dialogue multi protocole pour la négociation entre deux agents, nos agents sont capables de s'appuyer sur un raisonnement argumentatif pour utiliser des théories de communication lors d'une négociation.

L'exemple que nous avons implémenté est celui de l'agent voyages décrits dans le chapitre précédent.

Premièrement, nous allons vous présenter les outils qui nous ont permis de réaliser notre architecture de notre système ce qui nous permettrons de vous donner dans un deuxième temps une explication plus claire du projet même. Les outils utilisés lors du développement sont le langage Java et la plate-forme multi-agents JADE, Gorgias et prolog, Protége 2000 et Jena (Protége et Jena sont décrit dans le chapitre concernant les ontologies).

## 2 Les outils utilisés

### 2.1 La plate-forme JADE

JADE (Java Agent Development Framework - Bellifemine, Poggi, Rimassa, 1999, http ://jade.tilab.com/) est une plate-forme multi-agents développée en Java par le CSELT (Groupe de recherche de Gruppo Telecom, Italie) qui a comme but la construction des

systèmes multi-agents et la réalisation d'applications conformes à la norme FIPA (FIPA, 1997).

JADE comprend deux composantes de base : une plate-forme agents compatible FIPA et un paquet logiciel pour le développement des agents Java. JADE a été présentée dans [134].

Une plate-forme ? Une plate-forme multi-agents est un ensemble d'outils nécessaire à la construction et à la mise en service d'agents au sein d'un environnement spécifique.

**La norme FIPA (trois rôles principaux + langageAgentCommunicationLanguage)**

La norme FIPA est une norme qui rassemble un ensemble de règles pour qu'une société d'agents puisse inter-opérer.

En voici le modèle de référence pour une plate-forme SMA :

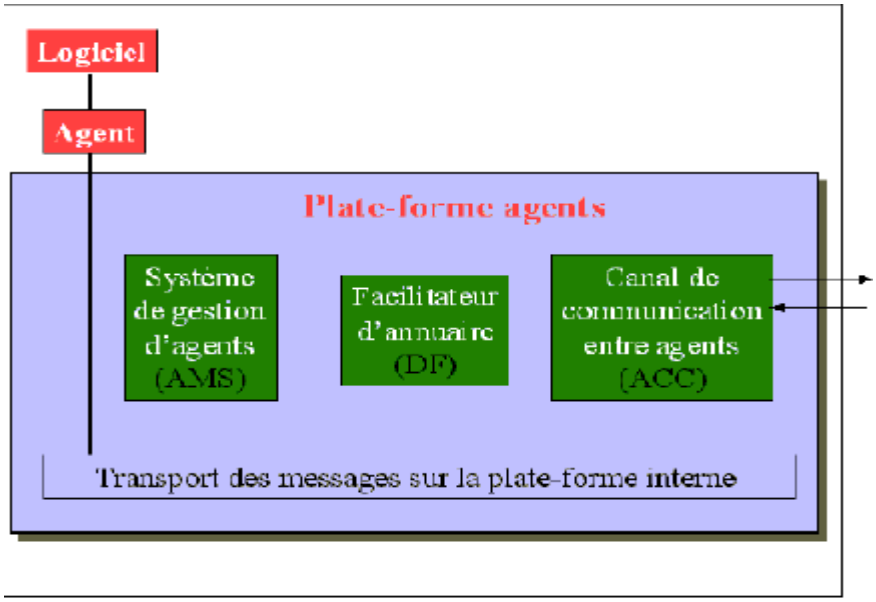


**Figure 5.1 : modelé de référence de plateforme JADE [132].**

Dans la figure précédente, on peut y voir 3 rôles principaux définissant une plate-forme multi-agents sous la norme FIPA. Ces rôles sont :

1. le système de gestion d'agents (AMS - Agent Management System) : il fournit le service de nommage, assure que chaque agent possède un nom unique et représente la fabrication des agents. En effet il peut créer et tuer un agent dans un conteneur de la plate-forme.
2. le facilitateur d'annuaire (DF - Director Facilitator) : il fournit le service dit de «page jaune». Il a pour but d'aider à la recherche d'un agent grâce à son nom ou encore à ses compétences par exemple.

3. le canal de communication entre agents (ACC – Agent Communication Channel) : il fournit la route pour les interactions de base entre les agents dans et hors de la plate-forme; c'est la méthode de communication implicite qui offre un service fiable et précis pour le routage des messages ; il doit aussi être compatible avec le protocole IIOP (Internet Inter-ORB Protocol) pour l'interopérabilité entre les différentes plates-formes multi-agents.

### 2.1.1 Le fonctionnement de l'environnement Jade

Un environnement d'exécution est appelé un conteneur. Chaque conteneur peut contenir plusieurs agents. L'ensemble des conteneurs actifs constitue une plate-forme. Une plate-forme peut s'exécuter sur plusieurs machines.

La plate-forme contient toujours un conteneur qui contient les agents fournissant les services de base.

Ce conteneur principal contient impérativement 2 agents (AMS et DF) qui sont créés au lancement de celle-ci.

La création d'un agent se faite par la programmation d'une classe qui hérite la classe **jade.core.Agent.**

Cette classe doit posséder la méthode **setup()** qui est appelée à l'initialisation de l'agent. A la création de l'agent, il va lui être attribué un identificateur à l'aide de la classe **jade.core.AID.**

Cet objet identificateur est de la forme : < nickname > @ < platform – name >. On peut accéder à cet identificateur grâce à la méthode **getAID()**.

Au moment de la création de l'agent, il est directement enregistré auprès du service de nommage et lorsque cet agent « meurt », le service de nommage l'enlève de sa base. L'AMS (Agent Management System) peut au niveau de la création intervenir afin de pouvoir faire des changements dans la description de l'agent ou encore en rechercher un en enregistrant l'agent auprès de l'AMS.

On peut aussi enregistrer l'agent auprès du DF (Directory Facilitator service) pour enrichir le service des pages jaunes afin de publier les services que peut rendre l'agent et pouvoir y faire appel par la suite pour effectuer des tâches par un agent compétant. Lorsqu'on crée un agent, il faut définir son comportement.

Un agent peut posséder plusieurs comportements à un moment donné. En général, on utilisera un comportement par tâche unitaire : recevoir un message, envoyer un message,

faire un traitement. Ces comportements sont appelés Behaviour dans la terminologie JADE.

Un exemple de la création basique d'un agent :

```
public class abstractAgentSimulation extends Agent{
protected void setup(){
String nickname = ''exemple'' ;
AID id = new AID(nickname, AID.ISLOCALNAME); }
}
```

Nous avons vu comment créer un agent, voyons maintenant le cycle de vie de l'un d'entre eu. Voici la description des différents états d'un agent en accord avec la spécification de la FIPA :

- Initiated : l'objet agent est créé mais n'est pas encore enregistré au près du service de nommage (AMS).
- Active : l'objet agent est enregistré au près du service de nommage (AMS), il possède désormais une adresse unique et peut donc communiquer avec les autres agents.
- Suspended : l'exécution de l'agent est suspendue.
- Waiting : l'agent est bloqué et doit attendre un événement comme un message par exemple.
- Transit : l'agent rentre dans cet état lorsqu'il migre dans un autre conteneur.
- Deleted : l'agent est détruit et supprimé du service de nommage (AMS).

Une fois l'agent activé, il lui faut pouvoir communiquer avec les autres agents. Pour cela, il y a le langage ACL (Acts Communication Language). Le format du message est défini dans la classe **jade.lang.acl.ACLMessage**.

La communication se faite de manière asynchrone, c'est à dire que l'agent peut bloquer un behaviour pour qu'il attend la réception d'un message, et pendant ce temps s'occupe d'autres opérations. De cette façon un agent peut avoir plusieurs behaviours responsables de plusieurs tâches distinctes en fonction de son comportement comme vu précédemment.

Lorsqu'un agent veut envoyer un message, il doit créer un nouvel objet **ACLMessage**, compléter ses champs avec des valeurs appropriées et enfin appeler la méthode **send().** Lorsqu'un agent veut recevoir un message, il doit utiliser la méthode **receive()** ou la méthode **blockingReceive().**

Voici quelques champs et leurs descriptions dont dispose un message ACL :

Sender : Expéditeur du message.
Receiver: Destinataires du message
Content : Contenu du message respectant
Ontology: Ontologie utilisée par le contenu
Protocol: Protocole utilisé

## 2.2 Gorgias

Gorgias (http ://www2.cs.ucy.ac.cy/ nkd/gorgias/) est un ensemble de règles qui combine l'idée de raisonner par relation de préférences sur une base de connaissances de manière argumentative et l'idée de se baser sur des données supposées vraies tant que rien ne dit le contraire donc incertaines, les abducibles. Gorgias s'appuie sur le framework LPwNF (logic programming without negation as failure). Le raisonnement argumentatif a été présenté dans [135].
Gorgias représente le système d'argumentation des agents du système, il contient un moteur d'inférence argumentative et qui nous permettre de modéliser une base de connaissances de notre système.

### 2.2.1 Le fonctionnement

Pour pouvoir se servir de et compiler Gorgias, il est nécessaire d'installer le logiciel SWI-Prolog (http://swi-prolog.org), une bibliothèque pour utiliser prolog depuis java. Cela consiste en une API permettant de manipuler des objets prolog et de poser des questions au moteur SWI-Prolog.
Ensuite, il faut que le fichier à compiler contenant des liens vers les fichiers permettant au logiciel SWI-Prolog d'utiliser les prédicats spécifiques à Gorgias. Il y a 2 liens à coder qui sont :
:- compile ('../lib/gorgias.pl').
:- compile ('../ext/lpwnf.pl').
Avec la première ligne, on charge le système proprement dit. Dans la deuxième ligne, on charge un ensemble de règles qui définissent les relations (de qualification) entre les arguments qui sont exploités dans une relation d'attaques et leur force dans l'argumentation. Le code d'une argumentation en Gorgias est composé de plusieurs éléments importants : la base de connaissances (knowledge Base), les liens entre les

différentes connaissances qui permettent une graduation de l'enchaînement possible des arguments et une base d'hypothèses.

# 3 Simulation de négociation entre les agents

Nous allons maintenant simuler l'interaction entre les différents agents de notre architecture et les différentes étapes de processus de négociation.

## 3.1 Lancement de plateforme Jade avec nos agents

Pour lancer l'environnement JADE et exécuter les agents, on doit exécuter la classe : Lancer_Dialogue.

Cette classe lance la plateforme Jade avec les agents de système à savoir : l'agent client L'Agent de l'agence de voyages et Agent d'interface de client (figure 5.2).

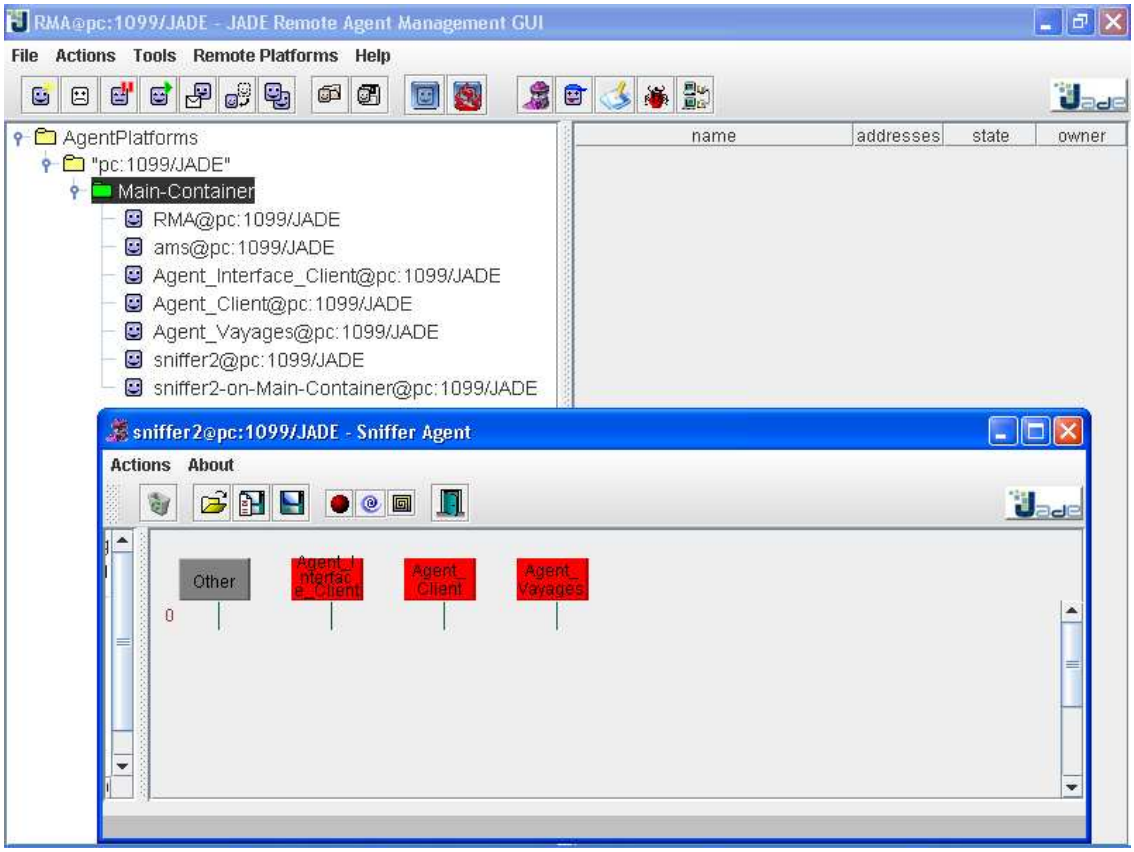


**Figure 5.2 : plateforme Jade.**

## 3.2 Agent d'interface de client

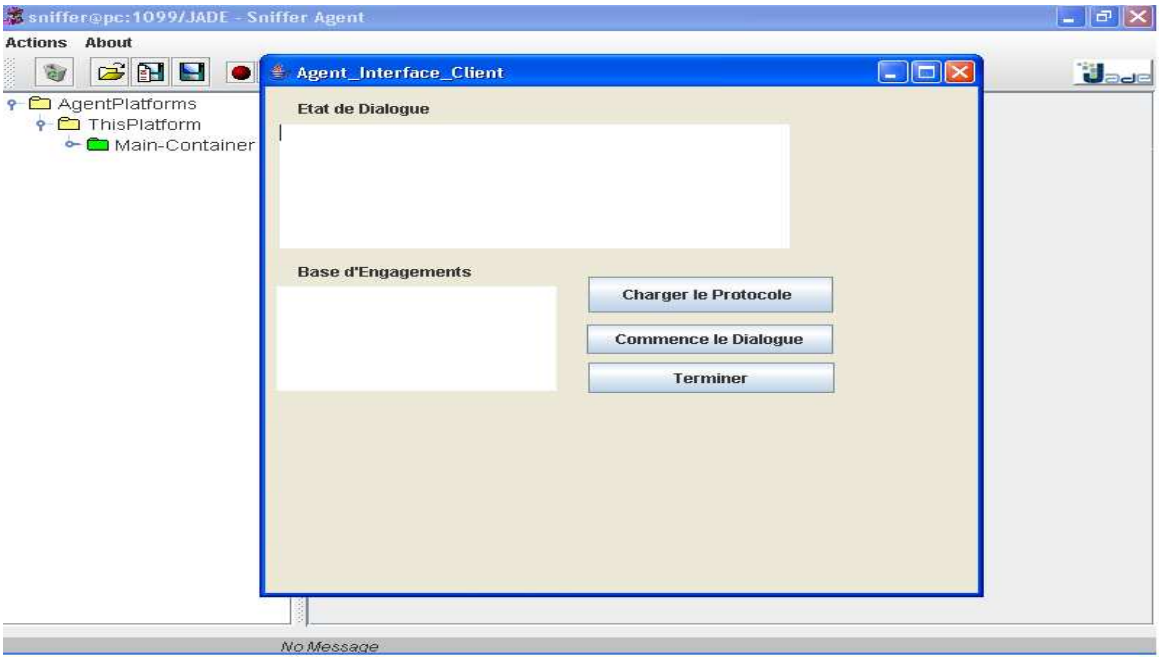


**Figure 5.3 : Agent d'interface de client.**

L'agent interface de client charge un protocole de négociation argumentative depuis l'ontologie des protocoles partagée via le bouton: *Charger-le-Protocole*.

Apres, l'agent commence le dialogue par l'envoyer d'un message à l'agent client.

Le champ de texte ; Etat de Dialogue affiche l'état de dialogue durant le processus de négociation.

Le champ de texte ; Base d'engagement affiche les engagements pris par l'autre agent.

Le client **e**nvoie à l'agent client un message ACL pour allouer une voiture, les messages ACL échanges entre les agents sont représentés sous la forme suivant :

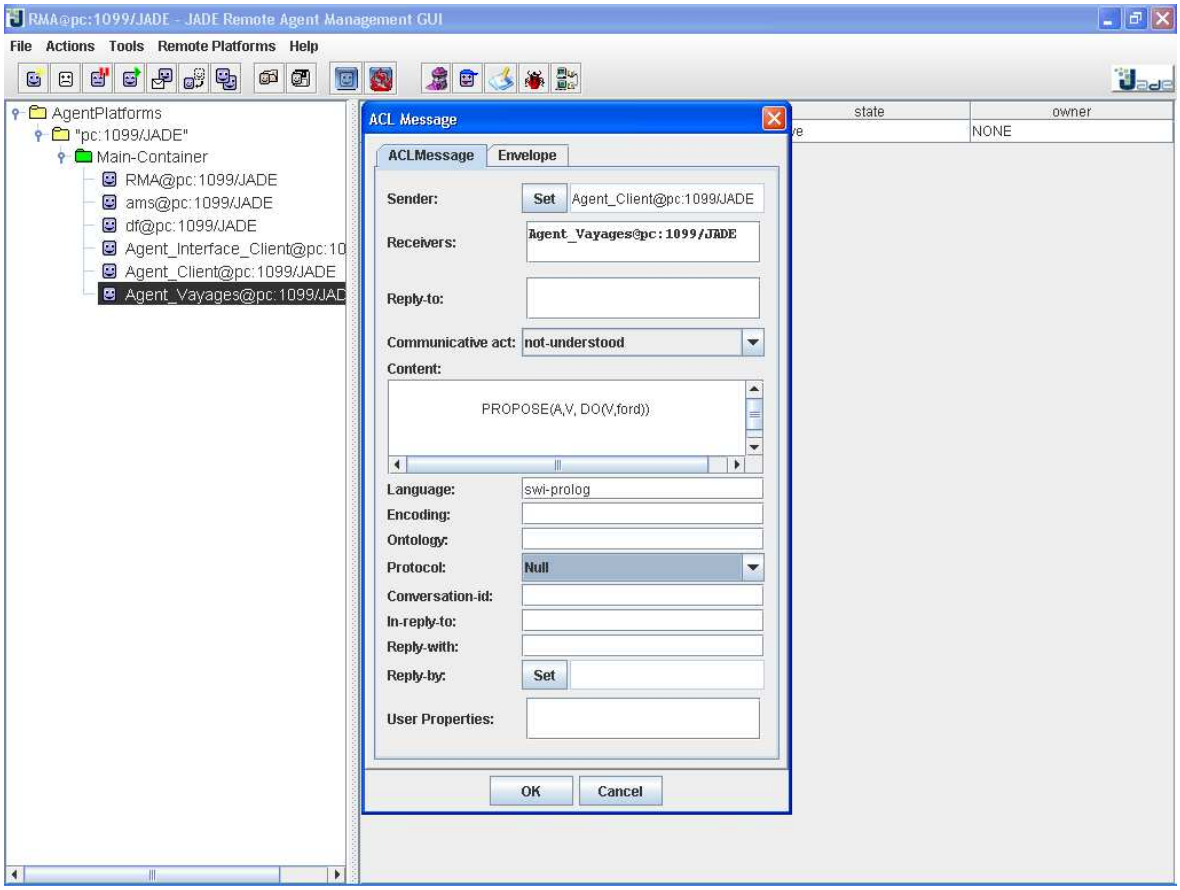


**Figure 5.4 : les messages ACL dans JADE.**

# 4 Conclusion

Dans ce chapitre, nous avons essayé de donner des éléments qui nous ont permet d'implanter notre architecture et les outils utilisés à savoir la plate-forme Jade et le système argumentatif Gorgias.

# Conclusion générale

Les systèmes de dialogue multi-agents suscitent de nos jours un intérêt croissant, avec en particulier la négociation automatique.

D'un part, l'émergence de technologie d'Internet a permis d'implémenter plusieurs systèmes de dialogue pour des domaines divergents avec une prédominance de persuasion. Mais, quelques applications, comme le commerce électronique, nécessitent d'autres types de dialogue à savoir la négociation.

D'autre part, la plupart des systèmes proposés supportent un seul protocole d'interaction fixé dans l'agent. Par conséquence, ces systèmes sont fermés; une propriété non souhaitée surtout avec l'apparition de la technologie de web services et les ontologies.

## Bilan

Dans notre démarche, nous avons commencé par la présentation du domaine des systèmes multi-agents dont lesquels nous avons présenté les définitions des agents informatiques, leurs caractéristiques et leurs différents types.

Nous avons vu aussi, les systèmes multi-agents, leurs domaines d'applications et leurs propriétés ; à savoir l'interaction, l'organisation et les mécanismes mis en œuvre pour la communication.

Ensuite, nous avons découvert les protocoles d'interaction et leur spécification par les ontologies, nous avons définit aussi la notion de protocole d'interaction et ses différents types. On a présenté les ontologies, leurs caractéristiques, leurs domaines d'application, et les outils nécessaires à leur développement à savoir : les langages de représentation, les outils d'édition et d'interrogation. Enfin, nous avons présenté les protocoles d'interaction et plus spécifiquement les protocoles de négociation par les ontologies.

Après, nous nous sommes attachés à la présentation de système de dialogue. Nous avons définit la notion de ce dernier, les différents types de dialogue, et les principales approches utilisées pour leurs constructions tout en focalisant sur l'argumentation. Après, On a présenté la négociation. Finalement, nous avons montré quelques aspects d'implémentation de systèmes de dialogue.

Pour concrétiser notre système nous avons conçu une architecture qui supporte ce système dont lequel un tel dialogue peut avoir lieu et qui s'intéresse à l'automatisation de la négociation entre agents interagissant dans le cadre du commerce électronique.

En plus, l'architecture de notre système a également pour but de fournir un ensemble de composants génériques pour intégrer différents types de protocoles.

Donc notre contribution a été :

- La construction d'une architecture d'un système de dialogue multi-agents basé sur l'argumentation pour la négociation automatique entre deux agents.
- L'intégration d'une ontologie de protocoles de négociation argumentative dans notre système de dialogue. Cette ontologie joue le rôle d'un fournisseur de protocoles de négociation.
- L'instanciation de l'architecture de système dans le cadre du commerce électronique, avec une étude de cas qui comprenait un scénario d'une agence de voyages.

## Perspectives

Plusieurs voies de recherche ou de développement méritent d'être poursuivies. Nous identifions plusieurs perspectives :

- Améliorer le protocole et le moteur de raisonnement argumentatif. Ces améliorations permettent de supporter d'autres framework argumentatives.
- Implémentation d'autres protocoles de négociations argumentatives.
- Essayer d'incorporer la stratégie de négociation dans notre système.
- Mise en œuvre d'une ontologie spécifique au domaine de connaissance.
- Poursuivre davantage l'implémentation de notre architecture, son évolution et son expérimentation, afin de déterminer les difficultés rencontrées et éventuellement, envisager plusieurs améliorations ou extensions possibles au système : Par exemple pour supporter d'autres types de dialogue.

# Bibliographie

[1] T. Berners-Lee, J. Hendler, O. Lassila. The Semantic Web. Scientific America.Available, 2001.

[2] M. Wooldridge. An Introduction to Multi Agent Systems. Chichester, John Wiley & Sons Ltd, 2002.

[3] D. Walton , E. Krabbe. Commitment in Dialogue. Basic Concepts of Interpersonal Reasoning. SUNY Series in Logic and Language. State University of New York Press, Albany, NY, USA, 1995.

[4] F. H. Van Eemeren, R. F. Grootendorst, F. S. Henkemans. Fundamentals of Argumentation Theory: A Handbook of Historical Backgrounds and Contemporary Applications. Lawrence Erlbaum Associates, Hillsdale NJ, USA, 1996.

[5] C. I. Chesñevar, A. Maguitman,R. P. Loui. Logical models of argument. ACM Computing Surveys 32(4), 337–383, 2000.

[6] C. Reed, T. J. Norman. Argumentation Machines: New Frontiers in Argument and Computation. Argumentation Library, vol. 9. Kluwer Academic Publishers, Dordrecht, the Netherlands , 2004.

[7] J. F. Allen, L. K. Schubert, G. Ferguson, P. Heeman, C. H. Hwang, T. Kato, M. Light, N. G. Martin, B. W. Miller, M. Poesio, and D. R. Traum. The TRAINS Project: a case study in building a conversational planning agent. Journal of Experimental and Theoretical Artificial Intelligence, p.7-48, 1995.

[8] K. Greenwood, T. Bench-Capon, P. McBurney. Towards a computational account of persuasion in law. In G. Sartor, editor, Proceedings of the Ninth International Conference on AI and Law (ICAIL-03), p.22-31, New York, NY, USA, 2003. ACM Press.

[9] M. Beveridge ,D. Milward. Definition of the high-level task specification language. Technical report, Deliverable D11, EU HOMEY Project, IST-2001-3243, http://www.acl.icnet.uk/lab/homey, 2003.

[10] A. Artikis. Executable Specification of Open Norm-Governed Computational Systems. PhD thesis, Department of Electrical and Electronic Engineering, Imperial College, London, 2003.

[11] J. Ferber. Les systèmes multi-agents, Inter Éditions, Paris, 1995.

[12] K. P. Sycara. The Many Faces of Agents, AI Magazine, 19(2), p.11-12, summer 1998.

[13] M. Bratman. Intention, plans, and practical reason. Harvard University Press, 1987.

[14] A. S. Rao and M. P. Georgeff. An abstract architecture for rational agents. In William Nebel, Bernhard ; Rich, Charles ; Swartout, editor, Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning, p.439-449, Cambridge, MA, 1992. Morgan Kaufmann.

[15] G. Weiss. Multi Agent Systems, A Modern Approach to Distributed Artificial Intelligence. The MIT Press, 2000.

[16] B. Chaib-draa and P. Levesque. Hierarchical models and communication in multi-agent environments. In Proceedings of the Sixth European Workshop on Modelling Autonomous Agents and Multi-Agent Worlds (MAAMAW-94), pages 119-134, Odense, Denmark, August 1994.

[17] B. Chaib-draa. Distributed Artificial Intelligence: An overview. In A. Ken, J. G. Williams, C. M. Hall, and R.Kent, editors, Encyclopedia Of Computer Science And Technology, volume 31, pages 215-243. Marcel Dekker, Inc,1994.

[18] B. Chaib-draa. Interaction between agents in routine, familiar and unfamiliar situations. International Journal of Intelligent and Cooperative Information Systems, 1(5):7-20, 1996.

[19] B. Chaib-draa ,P. Levesque. Hierarchical model and communication by signs, signals and symbols in multi-agent environments. Journal of Experimental and Theoretical AI (JETAI), 8:7-20, 1996.

[20] I. A. Ferguson. TouringMachines: An Architecture for Dynamic, Rational, Mobile Agents. PhD thesis, Clare Hall, University of Cambridge, UK, November 1992. (Also available as Technical Report No. 273, University of Cambridge Computer Laboratory).

[21] M. P. Georgeff and A. L. Lansky. Reactive reasoning and planning. In The Proceedings of AAAI-87, pages 677-682, Seattle, 1987.

[22] L. Gasser ,M. Huhns. Distributed Artificial Intelligence 2, San Mateo, CA: Morgan Kaufmann, 1989.

[23] E. H. Durfee ,V. Lesser. Negotiating task decomposition and allocation using partial global planning. In L. Gasser and M. Huhns, editors, Distributed Artificial Intelligence Volume II, pages 229-244. Pitman Publishing: London and Morgan Kaufmann: San Mateo, CA, 1989.

[24] K. P. Sycara. Multi-agents Systems, AI Magazine, 19(2): Summer 1998, p.79-92.

[25] F. Gandon: Ontology and Multi-Agent Systems for a Corporate Semantic Web.

[26] J. Ferber. Les systèmes multi-agents:un aperçu général. Techniques et sciences informatiques. Vol. 16, No. 8, pp. 979-1012, 1997.

[27] B. Moulin ,B. Chaib-draa. An overview of distributed artificial intelligence. In G. M. P. O'Hare and N. R. Jennings, editors, Foundations of Distributed AI, pages 3-54. John Wiley & Sons: Chichester, England, 1996.

[28] A. Vailly et M. A. Simon. Des systèmes experts coopérants, pourquoi, comment ?. Dans Cognitiva 87. pp. 183-188. AFCET. Paris. Mai 1987.

[29] E. Werner. Cooperating agents: A unified theory of communication and social structure. In L. Gasser and M. Huhns, editors, Distributed Artificial Intelligence Volume II, pages 3-36. Pitman Publishing: London and Morgan Kaufmann: San Mateo, CA, 1989.

[30] M. P. Papazoglou ,W. Heuvel. From Business Process to Coopeartive Information Systems: An Information Agents Perspective, p. 10-36, In the book Intelligent Information Agent: Agent-Based Information Discovery and Management on the Internet", Matthias Klusch, Springer, 1999.

[31] C. Shannon. A mathematical theory of communication. Bell System Technical Journal, 1948.

[32] T. Finin and R. Fritzson. KQML: a language and protocol for knowledge and information exchange. In Proceedings of the Thirteenth International Workshop on Distributed Artificial Intelligence, pages 126-136, Lake Quinalt, WA, July 1994.

[33] FIPA. Communicative Act Library Specification. Standard SC00037J, Foundation for Intelligent Physical Agents, 2002.

[34] J. Searle. Speech Acts: An Essay in the Philosophy of Language. Cambridge University Press, UK, 1969.

[35] J. L. Austin. How To Do Things with Words. Oxford University Press, Oxford, 1962.

[36] D. Vanderveken. Analyse et Simulation de Conversations. De la théorie des actes du discours aux systémes multiagents, chapter 2 : La structure logique des dialogues intelligents, pages p61–100. L'Interdisciplinaire, 1999.

[37] D. Vanderveken ,J. R. Searle. Foundations of Illocutionary Logic. Cambridge University Press, 1985.

[38] J. Habermas. The Theory of Communicative Action: Volume 1: Reason and the Rationalization of Society. Heinemann, London, 1984. Translation by T. McCarthy of: Theorie desKommunikativen Handelns, Band I, Handlungsrationalitat und gesellschaftliche Rationalisierung. Suhrkamp, Frankfurt, Germany. 1981.

[39] P. McBurney ,S. Parsons [2004]: Locutions for argumentation in agent interaction protocols. pp. 209—225 in: R. M. van Eijk, M-P. Huget and F. Dignum (Editors): Agent Communication. Revised Proceedings of the International Workshop on Agent Communication (AC2004), New York, NY, USA, July 2004. Lecture Notes in Artificial Intelligence 3396. Berlin, Germany: Springer.

[40] F. Dignum, B. Dunin-Keplicz, R. Verbrugge. Dialogue in team formation. In Issues in Agent Communication, pages 264–280, London, UK, 2000. Springer-Verlag.

[41] FIPA TC B. Fipa acl message structure specification. Component, Foundation for Intelligent Physical Agents, 6-12 2002. http ://fipa.org/specs/fipa00061/.

[42] FIPA TC C. Fipa contract net interaction protocol specification. Component, Foundation for Intelligent Physical Agents, 12-03 2003. http ://fipa.org/specs/fipa00029/.

[43] FIPA TC C. Fipa acl communicative act library specification. Component, Foundation for Intelligent Physical Agents, 6-12 2002. http ://fipa.org/specs/fipa00037/.

[44] J. Charlet , P. Laublet ,C. Reynaud. «Web sémantique», Rapport final : action scientifique 32 CNRS/STIC.http://doc.enssib.fr/IMG/pdf/ASWebSemantique2003.pdf

[45] J. S.Rosenchein ,G. Zlotkin. Designing Conventions for Automated Negotiation. AI Magazine,pages, p.29-46,1994.

[46] E. H. Durfee. Corrdination of Distributed Problem Solvers. Kluwer,1998.

[47] R. G. Smith. The Contract Net Protocole:High Level Communication and Control in a Distributed Problem Solver. IEEE Transactions on Computers,Vol.C-29,No.1,p.61-70,December 1980.

[48] R. Davis ,R. G. Smith. Negotiation as a Metaphor For Distributed Problem Solving.Artificiel Intelligence,Vol.20,No.1, p.63-109,January 1983.

[49] H. Wache,T. Vogele, U.Visser, H. Stuckenschmidt, G. Schuster, H Neumann et al. Ontology based integration of information - a survey of existing approaches. Paper presented at the IJCAI Workshop on Ontologies and Information Sharing ,2001.

[50] M. Uschold, M. Gruninger. Creating semantically integrated communities on the World Wide Web. Honolulu: Semantic Web Workhop,2002.

[51] R. Struder,V. Benjamin ,D. Fensel. «Knowledge Engineering, Principle and Methods». Data and Knowledge Enginnering, Vol. 25, p.161-197.

[52] J. Hendler. IEEE Intelligent systems. Agents and the Semantic Web, 2001.

[53] R. orazzon. Descriptiveand Formal Ontology. http://www.formalontology.it/,2003.

[54] N. Natalya, F. McGuinness, L. Deborah. “Ontology Development 101: A Guide to Creating Your First Ontology”, Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March

[55] What is an Ontology? http://mgeds.sourceforge.net/ontologies/index.php.

[56] T. R. Gruber . «Toward principle for the design of ontologies used for knowledge sharing», Technical report KSL n° 93-04, 1993, Standford University.

[57] R. Neches. “Acquisition of Knowledge for Sharing and Reuse,” Proceedings of the Annual Workshop on Knowledge Acquisition (edited by B.R. Gaines), Banff, Canada, October 6-11, 1991.

[58] A. G. Pérez, V. R. Benjamins. "Overview of Knowledge Sharing and Reuse Components : Ontologies and problem-Solving Methods". Proceeding the IJCAI-99 workshop on Ontologies and problem-Solving Methods (KRR5), Stockholm (Suède), Pp. 1.1-1.15, 1999.

[59] N. Guarino. « Formal Ontology and Information Systems», Formal Ontology in Information Systems. IOS Press, 1998.

[60] N. Hernandez. Ontologies de domaine pour la modélisation du contexte en recherche d’inrofmation. Thèse de doctorat, Université de Toulouse, 2005.

[61] M. Uschold , M. Grüninger, “ONTOLOGIES: Principles, Methods and Applications”. Knowledge Engineering Review. 1996.

[62] Resource Description Framework (RDF). http://www.w3.org/RDF/.

[63] RDF-Schema. http://www.w3.org/TR/rdf-schema/

[64] The DARPA Agent Markup Language. http://www.daml.org/.

[65] I. Horrocks. DAML+OIL: a Reason-able Web Ontology Language.

[66] Ontology Web Language (OWL). http://www.w3.org/OWL/

[67] S. Bechhofer, I. Horrocks, C. Goble, R. Stevens. "OILEd: a Reason-able Ontology Editor for the Semantic Web", In Proceedings of KI2001, Joint German/Austrian conference on Artificial Intelligence, September 19-21, Vienna. Springer-Verlag LNAI Vol. 2174, p.396-408 , 2001.

[68] Y. Sure, M. Erdmann, J. Angele, S. Staab, R. Studer,D. Wenke. "OntoEdit: Collaborative Ontology Engineering for the Semantic Web", In Proceedings of the International Semantic Web Conference 2002 (ISWC 2002) , Sardinia, Italia, June 9- 12 2002.

[69] Farquhar, R. Fikes, J. Rice. "The Ontolingua Server: A Tool for Collaborative Ontology Construction", In Proceedings of the 10th Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Alberta, Canada, p.44.1-44.19, 1996, http://www.ksl. stanford.edu/software/ontolingua/.

[70] http://www.ics.forth.gr/proj/isst/RDF/

[71] N. Natalya, F. McGuinness, L. Deborah. "Ontology Development 101: A Guide to Creating Your First Ontology", Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March

[72] http://www.hpl.hp.com/semweb/index.html.

[73] R. Studer, V. R. Benjamins, D. Fensel. Knowledge engineering, principles and methods. Data and Knowledge Engineering 25 (1–2), p.161–197, 1998.

[74] C. Hanachi , C. Sibertin-Blanc. «Protocol Moderators as active Middle-Agents in Multi-Agent Systems», Autonomous Agents and Multi-Agent Systems (AAMAS), Vol.8, n°3, p.131-164, Avril 2004.

[75] B. Benatallah, F. Casati, F. Toumani. «Representing, Analyzing and Managing Web Service Protocols». Data and Knowledge Engineering, Vol. 58, n° 3, p.327-357, September 2006.

[76] V. Tamma, S. Phelps, I. Dickinson, M. Wooldridge. «Ontologies for supporting negotiation in e-commerce», Engineering applications of artificial intelligence, Vol.18, p.223-236, March 2005.

[77] K. Amit, D. Nirmit, M. Ashok, W. Leena, P. Munindar. « A Semantic Protocol-Based Approach for Developing Business Processes », In The Second International Conference on Service-Oriented Computing (ICSOC), New York City, USA, p.99-107,November 2004.

[78]L. Charles Hamblin. Fallacies. Methuen, 1970.

[79] L. Amgoud, S. Parsons et N. Maudet. Arguments, dialogue, and negotiation. In Proceedings of the Fourteenth European Conference on Artificial Intelligence, pages 338–342, Berlin, Germany, 2000.

[80] J. MacKenzie. Question-begging in non-cumulative systems. Journal of Philosophical Logic, 8:117–133, 1979.

[81] R. McConachy , I. Zukerman. Dialogue requirements for argumentation systems. In Proceedings of IJCAI'99 Workshop on Knowledge and Reasoning in Practical Dialogue Systems, 1999.

[82] H. Prakken. On dialogue systems with speech acts, arguments, and counterarguments. In 7th European Workshop on Logic for Artificial Intelligence, Malaga, 2000.

[83] J. Barwise, L. Moss. Vicious Circles. Number 60 in CSLI Lecture Notes. CSLI Publications, Stanford, CA, 1996.

[84] P. Dunne, P. McBurney. Optimal utterances in dialogue protocols. In Proceedings of the Second International Conference on Autonomous Agents and Multiagent Systems (AAMAS-03), pages 608–615, 2003.

[85] N. Maudet ,F. Evrard. A generic framework for dialogue game implementation. In Proceedings of the Second Workshop on Formal Semantics and Pragmatics of Dialogue, Enschede, The Netherlands, 1998. University of Twente.

[86] S. Parsons, P. McBurney. Argumentation-based communication between agents. In Communications in Multiagent Systems, number 2650 in Springer Lecture Notes in AI, pages 164–178, Berlin, 2003. Springer Verlag.

[87] H. Prakken. A formal study of two-person dialogue games for argumentation. Technical report, Institute of Information and Computing Sciences, Utrecht University, 2004.

[88] M. R. Genesereth, N. Nilso. Logical Foundations of Artificial Intelligence. Morgan Kaufmann, USA, 1987.

[89] M.E. .What is intention? In Cohen editors, intentions in Communication, The MIT Press.15-32, 1990.

[90] M. P. Georgeff, B. Pell, M. Pollack, M.Tambe, M. Wooldridge. The belief-desire-intention model of agency. In Intelligent Agents, V, LNAI Volume 1555, Springer, Berlin, 1-10, 1999.

[91] ASPIC Consortium (2004) Theoretical framework for argumentation. Classified Technical report, 2004.

[92] F. H. van Eemeren, R. F. Grootendorst, F. S. Henkemans. Fundamentals of Argumentation Theory: A Handbook of Historical Backgrounds and Contemporary Applications. Lawrence Erlbaum Associates, Hillsdale NJ, USA, 1996.

[93] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial Intelligence 77(2), 321–358, 1995.

[94] M. Elhadad. Using argumentation in text generation. Journal of Pragmatics 24, 189-220, 1995.

[95] T. J. M Bench-Capon. Argument in artificial intelligence and law. Artificial Intelligence and Law 5(4), 249–261, 1997.

[96] T. F. Gordon, N. Karacapilidis. The Zeno argumentation framework. In: Proceedings of the Sixth International Conference on AI and Law, pp. 10–18. ACM Press, New York, 1997.

[97] K. Sycara. The PERSUADER. In: Shapiro, D. (ed.) The Encyclopedia of Artificial Intelligence, John Wiley & Sons, Chichester, 1992.

[98] S. Toulmin. The uses of argument. Cambridge University Press, 1956.

[99] G. Vreeswijk. The feasibility of Defeat in Defeasible Reasoning. Actes de 2nd International Conference on Principles of Knowledge Representation and Reasoning, KR'91. pp. 526-534, 1991.

[100] J. L. Pollock. How to reason defeasibly. Artificial Intelligence, vol. 57. pp. 1-42, 1992.

[101] F. Lin, Y. Shoham. Argument systems- An uniform basis for nonmonotonic reasoning. Actes de1st International Conference on Principles of Knowledge Representation and Reasoning, KR'89. pp. 245 - 255, 1989.

[102] G.R. Simari, R.P. Loui. A mathematical treatment of defeasible reasoning and its implementation. *Artificial Intelligence*, Vol. 53. pp. 125 - 157, 1992.

[103] H. Prakken, G. Sartor. On the relation between legal language and legal argument: assumptions, applicability and dynamic priorities. Actes de *5th International Conference in Artificial Intelligence and Law, ICAIL'95*, 1995.

[104] H. Prakken, G. Sartor. A Dialectical Model of Assessing Conflicting Arguments in Legal Reasoning. *Artificial Intelligence and Law*. pp. 331-368, 1996.

[105] L. Amgoud, S. Parsons et L. Perrussel. An argumentation framework based on contextual preferences. In Proceedings of the International Conference on Formal and Applied and Practical Reasoning, pages 59– 67, 2000.

[106] J. H. Müller. "Negotiation Principles" *Foundations of Distributed Artificial Intelligence*, John Wiley & Sons, pp. 211-229, 1996.

[107] N. R. Jennings, P. Faratin, A.R. Lomuscio, S. Parsons, C. Sierra, M. Wooldridge "Automated Negotiation: prospects, methods and challenge" *International Journal of Group Decision and Negotiation* (GDN), Vol 10, pp. 99-215, 2001.

[108] I. Rahwan, S. D. Ramchurn, N. R. Jennings, P. McBurney, S. Parsons, L. Sonenberg. Argumentation-based negotiation, 2004.

[109] I. Rahwan, L. Sonenberg, and F. Dignum. Towards interest-based negotiation. In AAMAS, 2003.

[110] S. D. Ramchurn, N. Jennings, and C. Sierra. Persuasive negotiation for autonomous agents: a rhetorical approach. In IJCAI Workshop on Computational Models of Natural Arguments, 2003.

[111] F. Sadri, F. Toni, and P. Torroni. Dialogues for negotiation: Agent varieties and dialogue sequences. In Proceedings of the International Workshop on Agent Theories, Architectures and Languages, ATAL, 2001.

[112] L. Amgoud, H. Prade. A possibilistic logic modeling of autonomous agents negotiation. In 11th Portuguese Conference on Artificial Intelligence, EPIA’2003, 2003.

[113] L. Amgoud, H. Prade. Reaching agreement through argumentation: A possibilistic approach. In 9 th International Conference on the Principles of Knowledge Representation and Reasoning, KR'2004, 2004.

[114] L. Carlson. Dialogue Games: an Approach to Discourse Analysis. Reidel Publishing Company, Dordrecht, 1983.

[115] Aristotle. Topics. Clarendon Press, Oxford, UK, 1928. (W. D. Ross, Editor).

[116] F. Zeuthen. Problems of Monopoly and Economic Warfare. Routledge and Sons, London, UK, 1930.

[117] P. Mirowski. Machine Dreams: Economics Becomes a Cyborg Science. Cambridge University Press, Cambridge, UK, 2002.

[118] A. Artikis, M. J. Sergot, J. Pitt. An executable specification of an argumentation protocol. In Proceedings of the Ninth International Conference on Artificial Intelligence and Law (ICAIL-03), pages 1{11, New York, 2003. ACM Press.

[119] E. Giunchiglia ,V. Lifschitz. An action language based on causal explanation: preliminary report. In Proceedings of Conference of the American Association for Artificial Intelligence (AAAI), pages 623{630. AAAI Press / The MIT Press, 1998.

[120] M. Beveridge , D. Milward. Combining task descriptions and ontological knowledge for adaptive dialogue. In Proc. TSD 2003, International Conference on Text Speech and Dialogue, Ceske Budejovice, Czech Republic, Sept. 2003.

[121] J. Fox , S. Das. Safe and Sound. Artificial Intelligence in Hazardous Applications. AAAI Press, The MIT Press, 2000.

[122] J. von Neumann, O. Morgenstern. Theory of Games and Economic Behavior. Princeton University Press, 1944.

[123] I. Lewin. A formal model of conversational game theory. In Proc. Gotalog-00, 4th Workshop on the Semantics and Pragmatics of Dialogue, Gothenburg, Sweden, 2000.

[124] R. A. Kowalski, F. Toni. Abstract argumentation. Artificial Intelligence and Law, 4(3):275{296, 1996.

[125] J. D. MacKenzie. Question-begging in non-cumulative systems. Journal of Philosophical Logic, 8:117{133, 1979.

[126] T. Yuan, D. Moore, A. Grierson. Computational Agents as a Test-Bed to Study Philosophical Model "DE", A Development of MacKenzie's "DC". Informal Logic, 2003. In press.

[127] K. Greenwood, T. Bench-Capon, P. McBurney. Structuring dialogue between the People and their representatives. In R. Traunmuller, editor, Electronic Government: Proceedings of the Second International Conference (EGOV03), Prague, Czech Republic, Lecture Notes in Computer Science 2739, pages 55{62, Berlin, Germany, 2003. Springer.

[128] K. Greenwood, T. Bench-Capon, P. McBurney. Towards a computational account of persuasion in law. In G. Sartor, editor, Proceedings of the Ninth International Conference on AI and Law (ICAIL-03), pages 22{31, New York, NY, USA, 2003. ACM Press.

[129] D. N. Walton. Argument Schemes for Presumptive Reasoning. Lawrence Erlbaum Associates, Mahwah, NJ, USA, 1996.

[130] K. Atkinson, T. J. M. Bench-Capon, P. McBurney. Implementation of a Dialogue Game for Persuasion over Action. Technical Report ULCS-04-005, University of Liverpool, 2004.

[131] J. F. Allen, L. K. Schubert, G. Ferguson, P. Heeman, C. H. Hwang, T. Kato, M. Light, N. G. Martin, B. W. Miller, M. Poesio, and D. R. Traum. The TRAINS Project: a case study in building a conversational planning agent. Journal of Experimental and Theoretical Artificial Intelligence, 7:7{48, 1995

[132] P. McBurney, S. Parsons. Locutions for argumentation in agent interaction Protocols.

[133] I. Rahwan, L. Sonenberg, sssF. Dignum. On interest-based negotiation. In F. Dignum, editor, *Advances in Agent Communication*, volume 2922 of *Lecture Notes in Artificial Intelligence*, pages 383.401. Springer Verlag, Berlin, Germany, 2004c.

[134] F. Bellifemine, A. Poggi. Giovanni Rimassa, JADE - A FIPA-compliant agent framework. CSELT, 1999.

[135] A. Kakas, P. Moraïtis. Argumentation Based Decision Making for Autonomous Agents. Dept. of Computer Science, University of Cyprus, 2003.