Université Paris-Dauphine

Mémoire d'Habilitation à Diriger des Recherches

Spécialité : INFORMATIQUE

par

Pavlos MORAÏTIS

Decision Theoretic and Logic Based Agents for Multi-Agent Systems

Date de la soutenance: 6 Décembre 2002

Table of Contents

Introduction 3

1 Multi-Agent Decision Support Systems 5

1.1 The ARISTOT System…...6

1.2 The COSMIMA System…7

2 The Agent as an Individual Entity 11

2.1 Agent Architectures…..13

2.2 Argumentation…..15

2.2.1 Argumentation with Roles and Context…..16

2.2.2 A Computational Model for Agent Personality Modeling… 21

2.2.3 Capabilities and Personality…24

2.2.4 Future Work…28

2.3 Dynamic Planning…29

2.3.1 The Multi-Criteria Planning Model…30

2.3.2 Classification of Possible Changes and Reaction…31

2.3.3 Future Work…37

3 The Agent as a Social Entity 40

3.1 Agent Conversation…..41

3.1.1 Future Work…45

3.2 Distributed Planning…46

3.2.1 The Plan Merging Procedure…. 46

3.2.2 An Example…48

3.2.3 Conflicts Resolution and Negotiation…. 50

3.2.4 Future Work… 51

3.3 Automated Negotiation… 52

3.3.1 Negotiation Strategies…. 52

3.3.2 An Aggregation-Disaggregation Approach… 53

3.3.3 A Utility-Based Negotiation Model…56

3.3.4 An Argumentation-Based Negotiation Model…58

3.3.5 Future Work…60

4 Miscellaneous 61

4.1 Multiple Criteria Evaluation of Actions in Hierarchical Decomposition…61

4.2 Multi-Criteria based Task Allocation… 62

4.3 Why is Difficult to Make Decisions under Multiple Criteria… 62

5 **E-Commerce** **65**
5.1 Buying and Selling of Products………………………………………… 65
5.2 Buying and Selling of Services…………………………………………. 69

6 **Marketing** **73**
6.1 Consumer-Based Methodology for Products Penetration Strategy Selection..73
6.2 Agent's Types, Functionalities, Structure……………………………………74

7 **Information Services: The IMAGE System** **81**
7.1 The IMAGE Agent Scenario…………………………………………………81
7.2 General Description of IMAGE Modules……………………………………82
7.3 The Intelligent Module Architecture…………………………………………85

8 **Diagnosis** **88**
8.1 The Agents Architecture……………………………………………………...88
8.2 Agents Coordination…………………………………………………………90

9 **Agent Oriented Software Engineering (AOSE)** **93**

10 **Conclusion** **98**

References **101**
Personal References…………………………………………………………..101
References……………………………………………………………………105

Annex **111**
Encadrement d'Activités de Recherche………………………………………… 111
Collaborations liées à la Recherche……………………………………………...114

Introduction

My adventure in the field of agents and multi-agent systems started in the beginning of 90's. At that period I was a novice Ph.D. student, and the idea preoccupying my thoughts was "how to represent, model and solve complex, ill-structured (as Simon calls them), real problems". In particular, I had in mind the problem of "how to precisely represent a human organization (i.e. like a company) and especially how to model the underlying distributed decision making processes". What was interesting to me was not the specific problem, but the paradigm. This is certainly a complex problem, where several actors (i.e. decisions-makers) are involved, in different levels of responsibility (and therefore of autonomy), expressing different points of view, while these actors' decisions are often based on uncertain and imprecise data. Thus, the framework to work on was already there: distribution, interaction under different forms, such as coordination of actions expressing different points of view, cooperation among actors having common goals, centralized planning (or distributed in some situations) of actions at a superior decision level and distributed execution at inferior levels, communication among physically distributed actors, conflicts arising (and thus need of negotiation), will to impose one's own politic and thus need of argumentation.

What a scenario…but then, which concepts and what techniques can give us the necessary background in order to model such a situation? Searching in the literature is always a necessary first step and that is how I started reading about the concept of agent. Its definition it is not yet very precise. Carl Hewitt said in some article "the problem that Distributed Artificial Intelligence has to define what an agent is is the same that Artificial Intelligence has had to define what intelligence is". However, people already have spoken about an autonomous and intelligent entity, having its own goals, able to communicate and interact with other agents. These last properties give the social dimension of the agent and at the same time put the accent on the concept of multi-agent systems. Such systems correspond to agents' communities, where distribution and interaction are some of the fundamental generic concepts. Moreover, people talk about pro-activeness and mental states of the agent (i.e., beliefs, desires and intentions) and consider their link with commitments. In addition, researchers got interested in the emotions and the personalities of agents. The epistemological and philosophical implications of this scientific domain are evident. The motivation was there and I, a computer scientist but – at the same time – a votary of epistemology and philosophy, I found the right scientific space, where my intellectual and research interests were fulfilled; that is, agents and multi-agent systems.

The aim of this document is to give the trace of my modest itinerary of researcher in the above field, from the time I got my Ph.D. (in December 94) till today. This document starts with a short reference to my Ph.D. work and then is divided in two main parts. The first part is dedicated to my theoretical work on agents and multi-agent systems, which is mainly based on the use of decision theory and logic. This constitutes my will to contribute to the development of a formal multi-agent theory. As said before, an agent can be characterized by different properties. However, the characteristic that I personally find as the most exciting one is this duality between the individual and the social dimension of an agent. According to this principal characteristic, an agent can be considered as an individual entity (endowed with some more specific characteristics; also known as intrinsic in the literature) but simultaneously as a social entity (also endowed with more specific characteristics; known as extrinsic). I believe that the second dimension of the agent (i.e., the social) can be considered as fundamental for the concept of multi-agent systems.

My theoretical work concerns the agent both as an individual and a social entity. The first part of this document is therefore dedicated to these aspects. More precisely, section 2 presents the proposal of a new conception of an agent's architecture, along with the work concerning some capabilities of an agent as an individual entity, such as argumentation (and its use to different types of deliberation) and dynamic planning. Section 3 presents the work concerning the agent as a social entity, and more precisely the work done in agent conversation, distributed planning and negotiation. Section 4 presents some other aspects that attracted my interest.

The second part is dedicated to my application work. It concerns the application of several issues of my theoretical work in different application domains. More precisely, Section 5 presents the work done on e-commerce, Section 6 on marketing, Section 7 on information services, and Section 8 on diagnosis. Finally, section 9 presents some experimental work on agent-oriented software engineering.

References are divided in two parts. The first part presents my publications cited in this document. The second part presents the external citations constituting the theoretical framework of my research. Annex summarizes my research supervising activities (with the related list of works) and scientific collaborations.

1 Multi-Agent Decision Support Systems

As said before, my research in the multi-agent field has started with the use of this technology in order to model human organizations (more particularly, an enterprise), but also processes that are associated to it (such as distributed decision making).

The analysis of the distributed decision making process (i.e. strategic decision-making is a specific case) and of the problems that it deals with, has revealed the characteristics of the domain. It has allowed us to identify the nature of the problems (i.e. complex, ill-structured, irregular), the nature of the knowledge used (i.e. imprecise, qualitative, uncertain), the categories of involved actors who possessed it and the way that they interact in order to solve a mutual problem. More precisely, we have observed that such a process implies a decomposition of the problem to be solved in a multitude of sub-problems, an assignment of these sub-problems to different actors having a heterogeneous expertise to solve them, and finally a coordination process in order to achieve a coherent plan of actions along with a communication process. Actors consider these problems at different levels of abstraction, having a partial view of the overall problem and decisional functionalities of a variable complexity; thus, they need an organizational structure. The overall setting is associated to a distribution of roles and the existing data, which generate conflicts among actors. These conflicts are the results of decisions taken locally, which - in order to achieve local goals (eventually contradictory) - generate contradictory actions leading to the global goal failure. It is obvious that the "classic" decision support systems proposed in the literature (for a detailed analysis, the reader can see Moraitis 94) were not susceptible enough to take into account the distributed dimension of the situation, as well as the associated problems (i.e. organization, coordination, communication, etc.). The study of the multi-agent literature [Moraïtis, 95] has led me to consider that the problematic of this domain perfectly coincides with the characteristics of the context I had found.

Due to all the above, this new field started to fascinate me. My goal became therefore dual: a) how to use the agent concept and the proposed techniques in this field in order to conceive a new family of decision support systems, able to model distributed decision making process, and b) how to integrate concepts and techniques proposed in different fields of human sciences, such as decision theory, organization theory, cognitive sciences or control theory, in order to enrich the existing techniques in multi-agent field.

1.1 The ARISTOT System (A coopeRative Information STrategic Operation Tool)

The aim of this system was the modeling of the participating actors, the simulation of their interactions, the representation of the solution at different level of abstraction, the conception of a mechanism of conflicts detection and resolution and, finally, the proposal of several coherent alternative solutions, among whose a satisfactory but non necessarily optimal solution can be found. Different aspects of this work have been presented in [Moraitis, 93; 94; Pinson & Moraitis, 93; 95; 96; Moraitis & Pinson, 94a; 94b; Pinson, Moraitis & Louca, 96].

The approach on which this system is built on is called "Coherent Plan of Coordinated Actions" (CPCA). It aimed at modeling the hierarchical organizations type, which is very usual (i.e. car industries, governments, universities, army, etc.) in human organizations.

The CPCA approach proposes two types of agents: artificial agents and human agents. Three types of artificial agents are defined: strategic agents, decision center agents and specialist agents. These types model the actors at three levels of abstraction proposed in organization theory literature (i.e. strategic level, management level, operational level), while their roles depend of the specific type of organization.

The CPCA approach also proposes a conflicts detection mechanism and guaranties a feasible and coherent solution. In order to detect and solve the conflicts among the sub-goals of the same scenario of a complex goal resolution, the approach builds on the compatibility concept. This concept adopts the idea presented in [March & Simon, 77; Cyert & March; 63; Simon, 75], according to which, in order to ensure a coherent solution among several partial solutions, the organization must search for satisfactory solutions rather than optimal ones. This selection is based on the assignment of a relative priority order (similar to that proposed in [Saaty, 80]) on the different sub-goals related to the global goal achievement and on the actions proposed for the different sub-goals achievement. The priority order is represented by a coefficient taking values on the interval [0, 1].

The CPCA approach uses two types of communication among agents: direct communication (through messages passing) and indirect communication (through shared memory, such as a blackboard).

Four types of blackboards are considered:

-The Problem-Blackboard: it allows the control and the decision making distribution among the different types of the agents. It mainly allows human agents to follow the evolution of the global goal achievement.

-The Domain-Blackboard: it contains the plan of actions corresponding to an optimal solution for a specific sub-goal.

-The Compatibility-Blackboard: it contains the feasible plans of actions in the specific context for the set of sub-goals.

-The Strategy-Blackboard: it contains either a feasible and coherent solution or the sub-goals, which are incompatible in the current scenario.

Finally, the CPCA approach proposes two types of coordination between artificial and human agents: top-down coordination and bottom-up coordination. The top-down coordination concerns the way artificial agents will be organized in order to resolve a problem (who does what, how and when, who collaborates with whom). The bottom-up coordination involves artificial and human agents and performs the evaluation of the results proposed by the problem solving procedure coordinated by the top-down coordination process. This last process has two different effects: 1) the creation of a coherent global plan of actions, and 2) the dynamic re-organization of the agents' community in cases where either contradictions are generated or the solution is not satisfactory for the user.

Another (less automated) version of the ARISTOT system, where interactions between artificial and human agents at each level of abstraction take place, is presented in [Pinson, Louca & Moraitis, 97].

1.2 The COSMIMA Agent

COSMIMA [Moraitis, 94] is a generic agent model, cooperating with an artificial and/or human environment. This agent model has an individual control module, which is an extension of the blackboard model procedural control (see [Nii, 86]), a social control module allowing cooperative work, a conflict solving mechanism, and finally a suitable reasoning and decision making module, allowing it to play a double role according to the context (manager and/or contractor). The proposed agent model is able to cooperate into distributed decision making problems in other than hierarchical organizations (cyclic, linear, etc…) and therefore to be the base of enterprise modeling. Finally, the model is easily adaptable for solving any kind of complex problems (e.g. breakdown diagnosis or medical diagnosis problems).

PART 1

THEORETICAL WORK

L’ Agent comme une Entité Individuelle

Résumé

Dans cette section je présente mon travail considérant l’agent comme une entité individuelle. De ce point de vue l’attention est focalisée sur les propriétés de l’agent, telles que autonomie, proactivité, intelligence ou réactivité ainsi que sur ses différentes formes de raisonnement. Ce travail concerne la proposition de modèles formels pour différentes formes de raisonnement (p.ex., délibération pour résoudre un problème, choix d’un objectif en accord avec une politique décisionnelle ou pour la satisfaction de besoins et des motivations, choix d’un partenaire, d’une action communicative pendant un dialogue, ou délibération pour former un plan d’actions, etc.). Ces modèles sont liés aux différentes capacités possibles (p.ex ., résolution de problèmes, coopération, communication, etc.) qu’un agent déliberatif peut avoir et ils sont fondés sur la théorie de la décision, la logique ou les deux à la fois.

Plus précisément je présente mon travail sur les architectures des agents où je développe mon point de vue concernant une approche modulaire. Selon cette approche le comportement d’un agent est le résultat de l’interaction entre les différents modules qui composent son architecture, chaque module étant responsable d’un aspect particulier de son comportement global (c.a.d. résolution de problèmes, coopération, communication, etc.). Mon point de vue sur la modularité est completé par la considération que l’implementation de ses capacités implique un ou plusieurs processus décisionnels dans chaque module. La nature de ces processus n’est pas exactement la même. Néanmoins, nous pouvons considérer que certains d’entre eux ont comme caractéristique commune la prise de décisions, afin de choisir parmi différentes options (p.ex. choix d’un objectif à satisfaire pour un module de résolution de problèmes, d’un partenaire pour un module de coopération, etc.). Alors l’idée est d’avoir un modèle de déliberation qui pourrait nous donner la possibilité de représenter ces processus de manière uniforme. Pour cette raison nous avons proposé un modèle fondé sur l’argumentation.

A l’architecture modulaire d’un agent nous ajoutons une nouvelle dimension qui est celle de la personnalité. Ainsi son architecture est enrichie d’un module dédié à sa personnalité. A ce module nous associons aussi un processus décisionnel, l’idée étant que sa personnalité peut avoir une influence sur ses différentes capacités.

Dans cette section sont aussi présentés mes travaux liés à l’agent comme entité individuelle. Plus précisément je présente mon travail en argumentation. Ce travail concerne la proposition d’un modèle délibératif, son utilisation pour coder des personnalités d’agents par le biais de la modélisation de besoins et de motivations, son

utilisation pour représenter les processus déliberatifs associés aux capacités de l'agent et son implication à l'implementation de notre architecture modulaire.

Le cadre d'argumentation que nous proposons est une extension du cadre développé pendant la dernière décennie comme un résultat d'une série de travaux sur le lien de l'argumentation avec le raisonnement non monotone. Ce cadre s'appelle Programmation Logique sans Négation comme Echec. Notre modèle d'argumentation prend en compte des rôles et leurs relations, que les agents peuvent avoir dans un contexte précis où ces rôles sont définis (appellé contexte par défaut), ainsi que des contextes spécifiques qui peuvent renverser les relations (l'ordre de priorité) définies dans le contexte par défaut.

Ainsi, afin d'accommoder les rôles et le contexte dans le raisonnement argumentatif d'un agent, nous avons étendu le cadre de Programmation Logique sans Négation comme Echec, de telle façon que la relation de priorité entre les règles d'une théorie (la théorie d'un agent est répresentée sous forme de règles) ne soit plus statique mais dynamique. Ceci donne la possibilité de capter la nature non statique des préférences de l'agent associées aux rôles et au contexte.

Ensuite dans cette section je présente mon travail en planification dynamique. Ce travail concerne la proposition d'un modèle multi-critères, où les plans sont répresentés comme un graphe orienté. Dans notre approche nous tenons compte non seulement des changements provenant de l'environnement (ce qui est le cas des autres travaux dans le domaine) mais aussi des changements qui peuvent provenir de l'agent lui-même pendant le processus d'exécution, le poussant à changer ses préférences et par conséquent ses actions ou ses méthodes à évaluer ses actions.

Les changements sur les préférences de l'agent et à ses méthodes d'évaluation sont pris en compte comme une révision de trois structures spécifiques, nommées plans possibles, plans efficaces et meilleurs plans. Les préférences sont modélisées par des critères. Ainsi le formalisme que nous proposons nous permet de représenter ce problème de planification comme un problème de programmation dynamique multi-objectifs.

2 The Agent as an Individual Entity

The concept of agent is probably amongst the most exciting and promising ones in the recent history of computer science. An agent is endowed with a set of properties, such as autonomy, pro-activeness, intelligence, and reactivity, which make them indisputably unique compared to other innovative concepts (e.g. the concept of object). A basic property of an agent, which distinguishes him from the object-oriented case, is his capability to have his own goals and to refuse the execution of an action that somebody else (i.e. another agent or an human user) may propose to him (i.e., in case that he does not find a personal interest in these goals). This last property constitutes indeed a notable difference with the concept of object, since in the object-oriented case the decision about whether to execute an action lies inside the object that invokes the method. This distinction is nicely expressed in the slogan: “objects do it for free; agents do it for money”. For a deeper analysis on the agent’s properties and characteristics, the interested reader may see [Wooldridge & Jennings, 95; Jennings, Sycara & Wooldridge, 98; Weis, 99; Wooldridge, 01; Ferber, 99].

Personally speaking, what I find as the most attractive characteristic of an agent is this kind of duality between individual and social aspect. An agent, just like a human being, can indeed be considered under the prism of an individual dimension by focusing the interest particularly on his capabilities and his personality, but simultaneously under the prism of a social dimension, by focusing the interest on his role as a member of a society and, consequently, on his interactions with the other agents. It is therefore obvious that, like for humans, the individual dimension has a repercussion on the social dimension or, in other words, the social behavior of the agent depends on his individual profile. Simultaneously, his social environment, like also for humans, can have an influence on the formation of his personality (or character).

The abovementioned positions should give to the experienced reader the right impression that the author has adopted the deliberative (or cognitive) agent approach. I would like however to mention here that agent literature generally classifies agents in two categories, namely the deliberative (or cognitive) agents, which have reasoning capabilities, and the reactive agents, which act based on rules of the stimulus-action type. Currently, this distinction is less evident because the idea is to conceive agents, called hybrid, which, according to the circumstances, have the possibility to act as deliberative or reactive.

In [Huhns & Singh, 98] one can find an interesting analysis on the abovementioned duality. As mentioned there, characteristics of agents are fundamentally separated into intrinsic (which are defined for an agent by itself) and extrinsic (which are defined for an agent in the context of other agents) properties. These characteristics are presented in the following tables:

Property	**Range of Values**
Lifespan	Transient to Long-Lived
Level of cognition	Reactive to Deliberative
Construction	Declarative to Procedural
Mobility	Stationary to itinerant
Adaptability	Fixed to Teachable to Autodidactic
Modeling	Of environment, themselves, or other agents

Table 1: Agent Characteristic: Intrinsic (source Huhns & Singh, 98)

Property	**Range of Values**
Locality	Local to Remote
Social Autonomy	Independent to Controlled
Sociability	Autistic, Aware, Responsible, Team Player
Friendliness	Cooperative to Competitive to Antagonistic
Interactions	Logistics: direct or via facilitators, mediators, or no-agents Style/Quality/Nature: with agents/world/both Semantic Level: declarative or procedural communications

Table 2: Agent Characteristic: Extrinsic (source Huhns & Singh, 98)

My research work and goals are expressed through this individual/social duality mentioned above. Thus, one part of my work is dedicated to the agent as individual entity and it will be presented first. This work concerns the proposal of formal models of different forms of reasoning, (e.g. deliberation to solve a problem, choose a goal according to a decision policy or for needs and motivations satisfaction, choose a partner, a communicative act during a dialogue, or deliberation to form a plan of actions, etc.), linked to different possible capabilities (e.g. problems resolution, cooperation, communication, etc.) a deliberative agent can have. These models are based on decision theory, logic, or a combination of both. Another goal is to capture the social dimension of the agent. For this reason, the abovementioned work is extended and adapted in order to propose formal models of interactions in multi-agent systems. As explained in the sequel, interactions can be of various kinds, including communication, negotiation (through argumentation or not), cooperation, coordination (through distributed planning or not), while agents are studied as social entities (members of multi-agent systems). This second facet of my work, related to the agent as a social entity, will be presented later in this document.

2.1 Agent Architectures

Specification of agents' architectures is an important aspect in multi-agent theory. As expected, different approaches have proposed different types of such architectures. An interesting discussion of the state of the art on the subject can be found in [Wooldridge 99]. Since the beginning of my work in MAS [Moraitis, 94], I opted for a modular agent architecture. I argue that the proposed types of architectures so far do not fully correspond to my own idea for a modular architecture. More specifically, according to my idea, all the necessary structures for the representation of a particular feature of the behavior of an agent are included within the same module. Therefore, an agent can be represented as a set of modules, each of them being responsible for a particular aspect of the behavior of the agent. Moreover, his overall behavior is the result of the interaction among these different modules [see for example Karacapilidis & Moraïtis, 01b; 02a]. The different aspects of a behavior correspond to the different capabilities an agent can have. For example, the role of a problem solving module can be the deliberation in order to choose a goal among several possible ones, of a planning module to generate a plan for a goal achievement, of a cooperation module to choose partners in the context of a collective work, while the role of a communication module is to ensure the communication of the agent with the external world (i.e. other agents or human users). As shown in the following, the structure of the modules can be the same (in Section 3.1 we present such a possible structure) or different (see for example Section 4.1 or in Sabater & al., 02). However, in the first case such modules are instantiated differently, according to the specific case (i.e. the type of the module, the type of the agent and the application domain). [Sabater & al, 02] present a modular approach, close to mine, which however has several differences concerning the adopted mechanism for communication among the modules, the granularity of the structures considered as modules, and the adopted reasoning mechanisms. My point of view on modularity could be considered together with the fact that the implementation of the agent's capabilities involves one or several deliberative processes within each module. The nature of these processes is not the same. However, we can consider that some of them have as a common characteristic decision-making, in order to choose among different possible options (e.g. choice of a goal for a problem solving module, choice of a partner for a cooperation module, etc.). So the idea is to have a deliberation model, something, which gives us the possibility to represent these deliberative processes in a uniform matter [Kakas & Moraïtis, 03]. To this end, we have proposed [Kakas & Moraïtis 02a; 02b] a model of argumentative deliberation, which will be presented in Section 2.2.

The above modular structure of an agent's architecture is enriched with the introduction of another dimension. This new dimension concerns the introduction of a module dedicated to an agent's personality. We also associate a deliberation process in this module. The idea is that an agent's personality can have an influence on his different capabilities. For example, his personality can have an influence on the choice of his goals and the possible solutions, the choice of his partners, the evolution of a dialogue with another agent, his negotiation policy with another agent, etc. Thus, we can consider that his professional policy could suggest him to choose goals and actions that they would characterize him as collaborative, while his own personality to choose goals and actions that would characterize him as selfish.

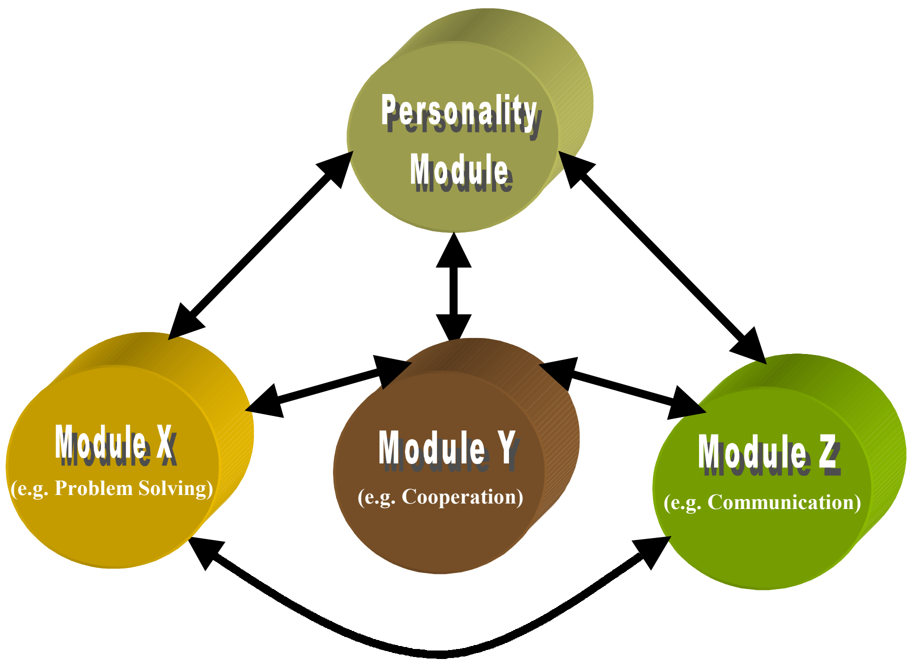


Figure 1: Modular architecture of agent

Similarly, his policy, upon which he chooses as collaborator being based on purely professional criteria, can suggest him to choose a particular collaborator, while his personality to choose the one which fits better with his personal preferences. Thus, the overall behavior of an agent is not only the result of the interaction among his modules, but also of the interaction of these modules and the module that implements his personality.

In the following, I present the argumentative deliberation model, its use to encode personalities through agent's needs and motivations modeling, its use to represent deliberation processes associated to the agents capabilities and its involvement in the implementation of our modular architecture. Then, I present my work that is associated to another aspect of an agent's behavior, namely the planning in a dynamic environment.

2.2 Argumentation

Autonomous agents need to make decisions under complex preferences policies that take into account different factors. These policies have a dynamic nature and are influenced by the particular state of the environment in which the agent finds himself. The argumentation framework we have proposed [Kakas & Moraïtis 02a; 02b; 03] intends to support the different agent's deliberation processes, as they have been presented above. It is an extension of an argumentation framework developed over the last decade as a result of a series of studies [e.g. Dung, 95; Kakas, Mancarella, & Dung 94] on the links of argumentation to non-monotonic reasoning. This framework, called Logic Programming without Negation as Failure (LPwNF), was proposed originally in [Kakas, Mancarella, & Dung 94] and can be seen as a realization of the more abstract frameworks of [Bodarenko & al, 97; Dung, 95]. The abstract attacking relation, i.e. the notion of argument and counter-argument, is realized through monotonic proofs of contrary conclusions and a priority relation on the sentences of the theory that make up these proofs. We have extended the framework, following the more recent approach of other works [Brewka, 01; Prakken & Sartor, 96] to allow this priority relation and thus attacking relation to be dynamic, making the framework more suitable for the application of the agent self deliberation. In LPwNF a non-monotonic argumentation theory is viewed as a pool of sentences (or rules) from which we must select a suitable subset, i.e. an argument, to reason with, e.g. to support a conclusion. Sentences in a LPwNF theory are written in the usual extended logic programming language with an explicit negation, but without the Negation as Failure (NAF) operator. We will often refer to the sentences of a theory as argument rules. In addition, these rules may be assigned locally a "relative strength" through a partial ordering relation. For example, we may have:

fly(X) ← bird (X) ¬fly(X) ← penguin (X)

bird(X) ← penguin (X) bird(tweety)

with an ordering relation between the rules that assigns the second rule higher than the first. This theory captures the usual example of "flying birds" with its exceptions, without the use of explicit qualifications of the default rules with abnormality conditions. We can conclude that tweety flies since we can derive this from the first rule and there is no way to derive ¬fly(tweety). We have an argument (i.e. a proof) for fly(tweety) but no argument for ¬fly(tweety). If we add to the theory penguin(tweety) then we can derive both fly(tweety) and ¬fly(tweety) - we have an argument for either conclusion. But in the non-monotonic argumentation semantics of the theory we can only conclude ¬fly(tweety). This overrides fly(tweety) since the argument that derives ¬fly(tweety) contains the second rule which is designated higher than the first rule which belongs to the argument that derives fly(tweety). We say that the argument for ¬fly(tweety) **attacks** the argument for fly(tweety) but not vise-versa.

The reader can find in [Dimopoulos & Kakas, 99; Kakas, Mancarella, & Dung 94] a full presentation of the argumentation framework of LPwNF. In this document, we will only present how this framework has been extended to allow dynamic priorities and formulate the general framework of argumentative agent deliberation.

2.2.1 Argumentation with Roles and Context

Agents, as it has been mentioned above, are always associated with a (social) environment of interaction. We call this the context of interaction. This determines the relationship between the possible roles the different agents can have within this environment. We consider, in line with much of the agent literature, (e.g [Panzarasa, Jennings & Norman, 02; Wooldridge, Jennings & Kinny, 00]), a role as a set of behavior obligations, rights and privileges determining its interaction with other roles.

Generally the substance of roles is associated to a default context that defines shared social relations of different forms (e.g. authority, friendship, relationship, etc.) and specifies the behavior of role between each other. Consequently it installs a partial order between roles that expresses preferences of behavior. For instance in the army context an officer gives orders that are obeyed by a soldier, or in a everyday context we respond in favor more easily to a friend than to a stranger. However, a default context that determines the basic roles filled by the agents is not the only environment where they could interact. For example, two friends can also be colleagues or an officer and a soldier can be family friends in civil life. Therefore we consider a second level of context, called specific context, which can overturn the pre-imposed, by the default context, ordering between roles and establish a different social relation between them. For instance, the authority relationship between an officer and a soldier would change under the specific context of a social meeting at home or the specific context of treason by the officer.

In order to accommodate in an agent's argumentative reasoning the roles and context as described above we have extended the framework of LPwNF so that the priority relation of a theory is not simply a static relation but a dynamic relation that captures the non-static preferences associated to roles and context. There is a natural way to do this. Following the same philosophy of approach as in [Prakken & Sartor, 96], the priority relation can be defined as part of the agent's theory $\mathcal{T}$ and then be given the same argumentation semantics along with the rest of the theory.

We distinguish the part of the theory that defines the priority relation by $\mathcal{P}$. Rules in $\mathcal{P}$ have the same form as any other rule, namely ground rules of the form $L \leftarrow L_1,\ldots,L_n$ where the head L refers to the (irreflexive) higher-priority relation, i.e. L has the general form $L = h\text{-}p(rule1, rule2)$. Also for any ground atom *h-p(rule1,rule2)* its negation is denoted by *h-p(rule2,rule1)* and vice-versa. For simplicity of presentation we will assume that the conditions of any rule in the theory do not refer to the predicate h-p thus avoiding self-reference problems. We now need to extend the semantic definitions of attack and admissibility.

Definition 1. Let $(\mathcal{T}, \mathcal{P})$ be a theory, T, T' $\subseteq \mathcal{T}$ and P, P' $\subseteq \mathcal{P}$. Then (T', P') **attacks** (T, P) iff there exists a literal L, $T_1 \subseteq T'$, $T_2 \subseteq T$, $P_1 \subseteq P'$ and $P_2 \subseteq P$ s.t.:

(i) $T_1 \cup P_1 \vdash_{min} L$ and $T_2 \cup P_2 \vdash_{min} \neg L$
(ii) $(\exists r' \in T_1 \cup P_1, r \in T_2 \cup P_2$ s.t. $T \cup P \vdash$ h-p(r, r'$)) \Rightarrow (\exists r' \in T_1 \cup P_1, r \in T_2 \cup P_2$ s.t. $T' \cup P' \vdash$ h-p(r', r))

Here, when L does not refer to h-p, $T \cup P \vdash_{min} L$ means that $T \vdash_{min} L$. This extended definition means that a composite argument (T', P') is a counter-argument to another such argument when they derive a contrary conclusion, L, and $(T' \cup P')$ makes the rules of its counter proof at least "as strong" as the rules for the proof by the argument that is under attack. Note that now the attack can occur on a contrary conclusion L that refers to the priority between rules.

Definition 2. Let $(\mathcal{T}, \mathcal{P})$ be a theory, $T \subseteq \mathcal{T}$ and $P \subseteq \mathcal{P}$. Then (T, P) is **admissible** iff $(T \cup P)$ is consistent and for any (T', P') if (T', P') attacks (T, P) then (T, P) attacks (T', P').

Hence when we have dynamic priorities, for an object-level argument (from $\mathcal{T}$) to be admissible it needs to take along with it priority arguments (from $\mathcal{P}$) to make itself at least "as strong" as the opposing counter-arguments. This need for priority rules can repeat itself when the initially chosen ones can themselves be attacked by opposing priority rules and again we would need to make now the priority rules themselves at least as strong as their opposing ones.

We can now define an agent's argumentation theory for describing his policy in an environment with roles and context as follows.

Definition 3. An agent's argumentative policy theory or theory, T, is a triple T=$(\mathcal{T}, \mathcal{P}_R, \mathcal{P}_C)$ where the rules in $\mathcal{T}$ do not refer to h-p, all the rules in $\mathcal{P}_R$ are priority rules with head h-p(r_1, r_2) s.t. $r_1, r_2 \in \mathcal{T}$ and all rules in $\mathcal{P}_R$ are priority rules with head h-p(R_1, R_2) s.t $R_1, R_2 \in \mathcal{P}_R \cup \mathcal{P}_C$.

We therefore have three levels in an agent's theory. In the first level we have the rules $\mathcal{T}$ that refer directly to the subject domain of the agent. We call these the **Object-level Decision Rules** of the agent. In the other two levels we have rules that relate to the policy under which the agent uses his object-level decision rules according to roles and context. We call the rules in $\mathcal{P}_R$ and $\mathcal{P}_C$, **Role (or Default Context) Priorities and (Specific) Context Priorities** respectively.

As an example, consider the following theory $\mathcal{T}$ representing (part of) the object-level decision rules of an employee in a company[1].

$r_1(A, Obj, A_1)$: $give(A, Obj, A_1) \leftarrow requests(A_1, Obj, A)$
$r_2(A, Obj, A_1)$: $\neg give(A, Obj, A_1) \leftarrow needs(A, Obj)$
$r_3(A, Obj, A_2, A_1)$: $\neg give(A, Obj, A_2) \leftarrow give(A, Obj, A_1), A_2 \neq A_1$

In addition, we have a theory $\mathcal{P}_R$ representing the general default behavior of the code of contact in the company relating to the roles of its employees: a request from a superior is in general stronger than an employee's own need; a request from another employee from a competitor department is in general weaker than its own need. Here and below we will use capitals to name the priority rules but these are not to be read as variables. Also for clarity of presentation we do not write explicitly the full name of a priority rule omitting in the name the ground terms of the rules.

R_1: $h\text{-}p\ (r_1(A, Obj, A_1), r_2(A, Obj, A_1)) \leftarrow higher\_rank(A_1, A)$
R_2: $h\text{-}p\ (r_2(A, Obj, A_1), r_1(A, Obj, A_1)) \leftarrow competitor(A, A_1)$
R_3: $h\text{-}p\ (r_1(A, Obj, A_1), r_1(A, Obj, A_2)) \leftarrow higher\_rank\ (A_1, A_2)$

Between the two alternatives to satisfy a request from a superior from a competing department or not, the first is stronger when these two departments are in the specific context of working together on a common project. On the other hand, if we are in a case where the employee who has an object and needs it, needs this urgently then he would prefer to keep it. Such policy is represented at the third level in $\mathcal{P}_C$.

C_1: $h\text{-}p\ (R_1(A, Obj, A_1), R_2(A, Obj, A_1)) \leftarrow common\_project(A, Obj, A_1)$
C_2: $h\text{-}p\ (R_2(, Obj, A_1)), R_1(A, Task1, A1)) \leftarrow urgent(A, Obj)$

Note the *modularity* of this representation. For example, if the company decides to change its policy "that employees should generally satisfy the requests of their superiors" to apply only to the direct manager of an employee we would simply replace R_1 by the new rule R'_1 without altering any other part of the theory:

R'_1: $h\text{-}p\ (r_1(A, Obj, A_1), r_2(A, Obj, A_1)) \leftarrow manager(A_1, A)$

Consider now a scenario where we have two agents ag_1 and ag_2 working in competing departments and that ag_2 requests an object from ag_1. This is represented by extra statements in the non-defeasible part, $\mathcal{T}_0$ of the theory, e.g. $competitor(ag_1, ag_2)$, $requests(ag_1, ag_2)$. So the question is "should ag_1 give the object to ag_2 or not?"

[1] Non-ground rules represent their instances in a given Herbrand universe

We e can easily see that if ag_1 does not need the object then, there are only admissible arguments for giving the object, e.g. Δ_1 = ($r_1(ag_1, obj, ag_2)$, {}) and supersets of this. This is because this does not have any counter-argument as there are no arguments for not giving the object since needs(ag_1, obj) does not hold. Suppose now that needs(ag_1,obj) does hold. In this case we do have an argument for not giving the object, namely Δ_2 = ($r_2(ag_1, obj, ag_2)$, {}). This is of the same strength as Δ_1 but the argument Δ'_2, formed by replacing in Δ_2 its empty set of rules of priority with {$R_2(r_2(ag_1, obj, ag_2), r_1(ag_1, obj, ag_2))$}, attacks Δ_1 and any of its supersets but not vice-versa: R_2 gives higher priority to the rules of Δ_2 and there is no set of priority rules with which we can extend Δ_1 to give its object-level rules equal priority as those of Δ_2. Hence we conclude skeptically that ag_1 will not give the object. This skeptical conclusion was based on the fact that the theory of ag_1 cannot prove that ag_2 is of higher rank than himself. If the agent learns that higher_rank(ag_2, ag_1) does hold then Δ'_2 and Δ'_1, obtained by adding to the priority rules of Δ'_1 the set {$R_1(r_1(ag_1, obj, ag_2), r_2(ag_1, obj, ag_2))$}, attack each other. Each one of these is an admissible argument for not giving or giving the object respectively and so we can draw both conclusions credulously.

Suppose that we also know that the requested object is for a common project of ag_1 and ag_2. The argument Δ'_2 is now not admissible since now it has another attack obtained by adding to the priority rule of Δ'_1 the extra priority rule $C_1(R_1(ag_1, obj, ag_2), R_2(ag_1, obj, ag_2))$ thus strengthening its derivation of h-p(r_1, r_2). The attack now is on the contrary conclusion h-p(r_1, r_2). In other words, the argumentative deliberation of the agent has moved one level up to examine what priority would the different roles have, within the specific context of a common project. Δ'_2 cannot attack back this attack and no extension of it exists that would strengthen its rules to do so. Hence there are no admissible arguments for not giving and ag_1 draws the skeptical conclusion to give the object.

We have seen in the above example that in several cases the admissibility of an argument depends on whether we have or not some background information about the specific case in which we are reasoning. For example, ag_1 may not have information on whether their two departments are in competition or not. This means that ag_1 cannot build an admissible argument for not giving the object, as he cannot use the priority rule R_2 that he might like to do. But this information maybe just unknown and if ag_1 wants to find a way to refuse the request he can reason further to find *assumptions* related to the unknown information under which he can build an admissible argument.

We can formalize this conditional form of argumentative reasoning by defining the notion of *supporting information* and extending argumentation with *abduction* on this missing information.

Definition 4. Let T= ($\mathcal{T}_0$, $\mathcal{T}$, $\mathcal{P}$) be a theory, and $\mathcal{A}$ a distinguished set of predicates in the language of the theory, called **abducible** predicates[2]. Given a goal G, a set S of abducible

[2] Typically, the theory $\mathcal{T}$ does not contain any rules for the abducible predicates.

literals consistent with the non-defeasible part $\mathcal{T}_0$ of $\mathcal{T}$, is called a **strong (resp. weak) supporting evidence** for G iff G is a skeptical (resp. credulous) consequence of ($\mathcal{T}_0 \cup$ S, $\mathcal{T}$, $\mathcal{P}$).

The structure of an argument can also be generalized as follows.

Definition 5. Let T = ($\mathcal{T}_0$, $\mathcal{T}$, $\mathcal{P}$) be a theory, and $\mathcal{A}$ its abducible predicates. A **supported argument** in T is a tuple (Δ, S), where S is a set of abducible literals consistent with $\mathcal{T}_0$ and Δ is a set of argument rules in T, which is not admissible in T, but is admissible in the theory ($\mathcal{T}_0 \cup$ S, $\mathcal{T}$, $\mathcal{P}$). We say that S supports the argument Δ.

The supporting information expressed through the abducibles predicates refers to the incomplete and evolving information of the external environment of interaction. Typically, this information pertains to the context of the environment, the roles between agents or any other aspect of the environment that is dynamic.

Given the above framework the **argumentative deliberation** of an agent can be formalized via the following basic reasoning functions.

Definition 6. Let A_g be an agent, T his argumentation theory, G a goal and S a set of supporting information consistent with $\mathcal{T}_0$. Then we say that A_g **deliberates** on G, supported by S, to produce s^{ag}, denoted by **deliberate**(A_g, G, S; s^{ag}), iff $s^{ag} \neq \{\}$ is a strong supporting evidence for G in the theory T∪S. If $s^{ag} = \{\}$ then we say that A_g accepts G under T∪S and is denoted by **accept(A_g, G, S)** . Furthermore, given an opposing goal $\hat{G}$ (e.g. ¬G) to G and s' produced by deliberation on $\hat{G}$, i.e. that deliberate(A_g, $\hat{G}$, S; s') holds, we say that s' is supporting evidence for agent A_g to refuse G in T∪S.

The presented approach has introduced, in the same spirit as [Sierra & al, 97; Amgoud & Parsons, 01], roles and context as a means to define non-static priorities between arguments of an agent. This helps to capture the social dimension of agents, as it incorporates in a natural way the influence of the environment of interaction (which includes other agents) on the agents "way of thinking and acting". The use of roles and dynamic context is a basic difference with most of other works [Sycara, 89; Parsons, Sierra & Jennings, 98; Kraus, Sycara & Evenchik, 98; Amgoud, Maudet & Parsons, 00] on agent argumentation. Our work complements and extends the approaches [Sierra & al, 97, Amgoud & Parsons, 01] with emphasis on enriching the self-argumentative deliberation of an agent. It complements these works by linking directly the preferences between different contexts, which these works propose, to a first level of roles that agents can have in a social context, called default context, showing how roles can be used to define in a natural way priorities between arguments of the agents filling these roles. It extends this previous work by incorporating reasoning on these preferences within the process of argumentative deliberation of an agent. This is done by introducing another dimension of context, called specific context, corresponding to a second level of deliberation for the agent. This allows a higher degree of flexibility in the adaptation of the agents' argumentative reasoning to a dynamically changing environment. In [Amgoud & Parsons, 01] the context preferences can also be dynamic but the account of this change is envisaged to occur outside the argumentative deliberation of the agent. An agent decides a-priori to change the context in which he is going to deliberate. In our case the change is integrated within the deliberation process of the agent.

This extra level of deliberation allows us to capture the fact that recognized roles in a context have their impact only within the default context where they are defined, although these roles always "follow" agents filling them, as a second identity in any other context they find themselves. Therefore agents who have some relationship, imposed by their respective roles, can be found in a specific context where the predefined (according to their relationship) order of importance between them has changed.

2.2.2 A Computational Model for Agent Personality Modelling

In this section, I present how our argumentation framework can help us model and encode an agent's needs corresponding to motivational factors, thus allowing us to express various personality profiles of an agent [Kakas & Moraïtis 02a; 03]. In particular, we examine the argumentative deliberation that an agent has to carry out in order to decide which needs to address at any current situation that he finds himself.

We have applied the same approach as we did when modeling a preference policy of an agent in a certain knowledge or problem domain, as it has been described in the previous section. We now simply consider the domain of an agent's needs and motivations where, according to the type or personality of an agent, the agent has a default (partial) preference amongst the different types of needs. Hence now the type of need, or the motivation that this need addresses, plays an analogous role to that of roles in the previous section. The motivations will then determine the basic behavior of the agent in choosing amongst his different needs and whenever we have some specific context this may overturn the default decision of the agent for a particular need.

We have adopted the work of Maslow [Maslow, 54] from Cognitive Psychology (see also Morignot & Hayes-Roth, 95; 96] where needs are categorized in five broad classes according to the motivation that they address. These are Physiological, Safety, Affiliation or Social, Achievement or Ego and Self-actualization or Learning. As the world changes a person is faced with a set of potential goals from which it selects to pursue those that are "most compatible with her/his (current) motivations". We choose to eat if we are hungry, we protect ourselves if we are in danger, we work hard to achieve a promotion etc. The theory states that in general there is an ordering amongst these five motivations that we follow in selecting the corresponding goals. But this ordering is only followed in general under the assumption of "other things being equal" and when special circumstances arise it does not apply. This is the first time that an argumentative deliberation framework is used to model an agent's needs following the Maslow's hierarchy of needs, in a way that, we believe, allows a natural expression of several behaviors. Therefore our aim here was to model and encode such motivating factors and their ordering in a natural way thus giving a computational model for agent behavior and personality.

Let us assume that an agent has a theory $\mathcal{T}$ describing the knowledge of the agent. Through this, together with his perception inputs, he generates a set of needs that he could possibly address at any particular situation that he finds himself. We will consider that these needs are associated to goals, G, e.g. to fill with petrol, to rest, to help someone, to promote himself, to help the community etc. For simplicity of presentation and without loss of generality we will assume that the agent can only carry out one goal at a time and thus any two goals activated by $\mathcal{T}$ oppose each other and a decision is needed to choose one. Again for simplicity we will assume that any one goal G is linked only to one of the five motivations above, m_j, and we will thus write G_j, j=1,...,5 to indicate this, with m_1=Physiological, m_2= Safety, m_3=Affiliation, m_4= Achievement, m_5=Self-actualization.

Given this theory $\mathcal{T}$ that generates potential goals an agent has a second level theory, $\mathcal{P}_M$, of priority rules on these goals according to their associated motivation. This theory helps the agent to choose amongst the potential goals that it has and forms part of his decision policy for this. It can be defined as follows.

Definition 1. Let A_g be an agent with knowledge theory $\mathcal{T}$. For each motivation, m_j, we denote by S_j the set of conditions, evaluated in $\mathcal{T}$, under which the agent considers that his needs pertaining to motivation m_j are **satisfied**. Let us also denote by N_j the set of conditions, evaluated in $\mathcal{T}$, under which the agent considers that his needs pertaining to motivation m_j are **critical**. We assume that S_j and N_j are disjoint and hence N_j corresponds to a subset of situations where $\neg S_j$ holds. Then the **default motivation preference theory** of A_g, denoted by $\mathcal{P}_M$, is a set of rules of the following form:

- R^1_{ij} : h-p(G_i, G_j) $\leftarrow N_i$
- R^2_{ij} : h-p(G_i, G_j) $\leftarrow \neg S_i, \neg N_j$

where G_i and G_j are any two potential goals, ($i \neq j$), of the agent associated to motivations m_i and m_j respectively.

The first rule refers to situations where we have a critical need to satisfy a goal G_i whereas the second rule refers to situations where the need G_j is not critical and so G_i can be preferred.

Hence when the conditions S_i hold an agent would not pursue goals of needs pertaining to this motivation m_i. In fact, we can assume that whenever a goal G_i is activated and is under consideration that $\neg S_i$ holds. On the other side of the spectrum when N_i holds the agent has an urgency to satisfy his needs under m_i and his behavior may change in order to do so. Situations where $\neg S_j$ and $\neg N_j$ both hold are in between cases where the decision of an agent to pursue a goal G_i will depend more strongly on the other simultaneous needs that he may have. These conditions S_i and N_i vary from agent to agent and their truth is evaluated by the agent using his knowledge theory.

For example, when a robotic agent has low_energy, that would make it non-functional, the condition N_1 is satisfied and a goal like G_1 = fill_up has, through the rules R^1_{1j} for $j \neq 1$, higher priority than any other goal. Similarly, when the energy level of the robotic agent is at some middle value, i.e. $\neg S_1$ and $\neg N_1$ hold, then the robot will again consider, through the rules R^2_{1j} for $j \neq 1$ the goal G_1 to fill up higher than other goals provided also that in such a situation there is no other goal whose need is critical. Hence if in addition the robotic agent is in great danger and hence N_2 holds then rule R^2_{12} does not apply and the robot will choose goal G_2= self_protect which gets a higher priority through R^1_{21}. In [Kakas & Moraïtis 02a; 03], we show that under some suitable conditions the agent can decide deterministically in any situation. However, in [Kakas & Moraïtis 02b] we show that there also exist situations where the agent can be in a dilemma as his theory can provide him with an admissible argument for each need. For example, a robotic agent may at the same time be low in energy and in danger. Similarly, the robotic agent may be in danger but also need to carry out an urgent task of helping someone.

According to Maslow's theory decisions are then taken following a basic hierarchy amongst needs. For humans this basic hierarchy puts the Physiological needs above all other needs, Safety as the second most important with Affiliation, Achievement and Self-Actualization following in this order. Under this hierarchy a robotic agent would choose to fill its battery despite the danger or avoid a danger rather than give help. One way to model in $\mathcal{P}_M$ such a hierarchy of needs that helps resolve the dilemmas is as follows. For each pair k, l s.t. $k \neq l$ the theory $\mathcal{P}_M$ contains only one of the rules R^1_{kl} or R^1_{lk}. Deciding in this way which priority rules, R^1, to include in the theory gives a basic profile to the agent.

But this would only give us a partial solution to the problem not resolving dilemmas that are not related to urgent needs and a similar decision needs to be taken with respect to the second category of rules, R^2, in $\mathcal{P}_M$. More importantly this approach is too rigid in the sense that the chosen hierarchy in this way can never be overturned under any circumstance. In other words, there may be special circumstances where the basic hierarchy in the profile of an agent should not be followed. This extra level of flexibility is needed to capture an adaptive dynamic behavior of an agent. For example, an agent may decide, despite his basic preference to avoid danger rather than help someone, to help when this is a close friend or a child.

We can solve these problems by extending the agent theory with a third level analogous to the specific context level presented in the previous sections.

Definition 2. An agent theory expressing his decision policy on needs is a theory T=($\mathcal{T}$, $\mathcal{P}_M$, $\mathcal{P}_C$) where $\mathcal{T}$ and $\mathcal{P}_M$ are defined as above and $\mathcal{P}_C$ contains the following types of rules. For each pair of rules R^k_{ij} , R^k_{ji} in $\mathcal{P}_M$ we have the following rules in $\mathcal{P}_C$:

- H^k_{ij} : h-p(R^k_{ij}, R^k_{ji})← true
- E^k_{ji} : h-p(R^k_{ji}, R^k_{ij})← sc^k_{ji}
- C^k_{ji} : h-p(E^k_{ji}, H^k_{ij})← true

where sc^k_{ji} are (special) conditions whose truth can be evaluated in $\mathcal{T}$. The rules H^k_{ij} are called the **basic hierarchy** of the theory T and the rules E^k_{ji} the **exception policy** of the theory T. The theory $\mathcal{P}_C$ contains exactly one of the basic hierarchy rules H^k_{ij} and H^k_{ji} for each k=1,2 and $i \neq j$.

Choosing which one of the basic hierarchy rules H^k_{ij} or H^k_{ji} to have determines the default preference of needs G_i over G_j or G_j over G_i respectively (for k=1 in critical situations and for k=2 in non-critical situations). The special conditions sc_{ji} define the specific contexts under which this preference is overturned. They are evaluated by the agent in his knowledge theory $\mathcal{T}$.

2.2.3 Capabilities and Personality

In this section, I explain our idea to integrate the personality of an agent in his architecture and I present how the personality of an agent can influence his decision making of his different capabilities (i.e. problem solving, cooperation, communication, etc.). As already said, we consider that the implementation of his capabilities as well as of his decision process on which needs to address at any situation, pertaining to his personality, involve several deliberation processes. We then show, through two examples (one with the problem solving module and one with the cooperation module) how we can represent this kind of deliberations processes by using our argumentative deliberation framework. This will also help the reader to understand better how the work presented in the previous sections can be applied.

Let's consider that an agent α is provided with the theories presented in the following. Theory T1 corresponds to a part of the knowledge of one of his modules related to his problem solving capability and expresses the "way of thinking" and the policy under which the agent chooses the goals he must achieve in his professional context. We suppose that the possible choice is between a goal perform(A, Task1, A1) and perform(A, Task2, self) which semantically mean to perform the Task1 for the agent A1 or to perform the Task2 for himself, respectively. According to Maslow's theory, goal perform(A, Task1, A1) belongs to the category of goals G_3 associated to the motivation m_3=affiliation, thus to the need of the agent to satisfy goals for the society. Goal perform(A, Task2, self) belongs to the category G_4 associated to the motivation m_4=achievement, thus his need to satisfy personal ambitions.

One of the problems we must take into account here is the “labeling” of the various goals according to the need or motivation each one is (primarily) related to. When a goal is generated (which is a problem to be considered in the future-see future work) the agent could give it a “label” or in other words categorize it in one of the specified, by the Maslow’s hierarchy, categories of motivations. In our present work we consider that the association of the agent’s possible goals with the defined motivations’ categories is part of his background knowledge and it is acquired during the agent’s design phase. However this could be the result of a learning procedure. The idea is that the agent could categorize a goal in one of the abovementioned motivations’ categories m_l, if he observes, that each time this goal has been achieved in the past, has had a positive repercussion on the given motivation m_l (i.e. the conditions S_l are satisfied). This idea will be exploited in the future.

Coming now back to our example we consider that the background knowledge of the problem solving of the agent α is: {competitor(α, α_1), higher_rank(α_1, α), common_project(α, ask1, α_1), category(perform(α, task1, α_1))=G_3, category(perform(α, task1, self))=G_4} for specific agents α, α_1 and task1.

Theory T1 is presented as follows:

r_1: perform(A, Task1, A1)←ask(A1, Task1, A)
r_2: ¬perform(A, Task1, A1) ←perform(A, Task2, self)

R1: h-p (r1(A, Task1, A1), r2(A, Task1, A1)) ←higher_rank(A1, A)
R2: h-p (r2(A, Task1, A1), r1(A, Task1, A1)) ←competitor(A, A1)

C1: h-p (R1(A, Task1, A1), R2(A, Task1, A1)) ←common_project(A, Task1, A1)
C2: h-p (R2(A, Task1, A1), R1(A, Task1, A1)) ←urgent(A, Task1)

According to T1 and his background theory, agent α will choose the achievement of the goal G_3. Thus, his policy in his professional context (environment) would characterize this agent as a collaborative and fully consistent with the philosophy of the company where he works.

Let’s also consider the theory T2 expressing “his way of thinking” (related to his personality). This theory, as presented in the following, would rather characterize the agent as selfish.

R^2_{43}: h-p (G_4, G_3) ← ¬S4 ∧ ¬N3
R^2_{34}: h-p (G_3, G_4) ← ¬S3 ∧ ¬N4

H^2_{43}: h-p (R^2_{43}, R^2_{34}) ← true
E^2_{34}: h-p (R^2_{34}, R^2_{43}) ← immediate_personal_profit (G_3)
C^2_{34}: h-p (E^2_{34}, H^2_{43}) ← true

We assume that the part of the background knowledge of the personality module of the agent concerning the problem solving module is such that all of: $\{\neg S4, \neg N4, \neg N3, \neg S3\}$ hold. Thus according to T2 and his background theory agent A will choose the goal G_4 because he has no information that Task1 could provide him with an immediate personal profit. According to our theory, this is a special condition that could overturn the default preference of needs of the agent, expressing his personality. Therefore, this agent will find himself in a dilemma because, reasoning as a professional, he will have to choose the goals G_3 while, reasoning as an individual, the goal G4.

Suppose now that the personality of the agent is given by the theory T3 instead of T2. This theory could rather characterize the agent as altruist or collaborative.

R^2_{34}: h-p $(G_3, G_4) \leftarrow \neg S3 \wedge \neg N4$
R^2_{43}: h-p $(G_4, G_3) \leftarrow \neg S4 \wedge \neg N3$

H^2_{34}: h-p $(R^2_{34}, R^2_{43}) \leftarrow$ true
E^2_{43}: h-p $(R^2_{43}, R^2_{34}) \leftarrow$ against_principle_reasons(G_3)
C^2_{43}: h-p $(E^2_{43}, H^2_{34}) \leftarrow$ true

According to T3, the agent will choose goal G_3 because he has no information (see above) that choosing to achieve Task1 could be against principle reasons (i.e. special conditions). Therefore, in this case, both T1 and T3, expressing his "way of thinking" as a professional and as an individual character, respectively, are consistent; therefore, the agent will not have any dilemma to choose the same goal G_3.

In order to illustrate further our approach and to show its generality in the deliberation of an agent for his different capabilities, let us consider another relationship between the personality and the capability of cooperation (i.e. the policy to find partners) of an agent.

Let us therefore suppose that the following theory is a part of the agent's cooperation module knowledge. This module, as we have said, is responsible for the decision of an agent of how to cooperate with other agents when this is considered necessary (i.e. when the agent is unable to solve a problem alone). Among other things, the role of this module is to find the appropriate collaborator for a specific task. The following theory T4 expresses the policy under which the agent selects his collaborators based on purely "professional" criteria. This theory, at the object level says that for any specific Task the agent needs only one collaborator and that any collaborator can be chosen according to his relevant expertise for the task at hand. In the roles level the theory says that if the task needs a manager he prefers to choose a manager agent A1 while if task needs a technical expert he prefers to choose an expert agent A2. When both apply (i.e. the task is both a management and technical one), the theory at the context level expresses a priority according to the current period. Thus if the current period imposes the need of a market share increase, a manager must be chosen (i.e. α_1), while if it imposes the image improvement, the quality eventually of the products, must be improved and therefore an expert must be chosen (i.e. α_2).

We assume that the background knowledge of the cooperation module of the agent α is: {management_task(task), manager(α_1), technical_task(task), expert(α_2), need_cooperation(a, task), relevant(task, α_1), relevant(task, α_2), market_share_increase_need_period(p_1), current_period(p_1), categoty(request_help(α, task, α_1))=G3, categoty(request_help(α, task, α_2))=G5}.

r1: request_help(A, Task, A1)←need_cooperation(A, Task), relevant(Task,A1)
r2: ¬ request_help(A, Task, A1)← request_help(A, Task, A2), A1≠A2

R1: h-p(r1(A, Task, A1), r1(A, Task, A2))←management_task(Task), manager(A1)
R2: h-p(r1(A, Task, A2)), r1(A, Task, A1)) ←technical_task(Task),expert(A2)

C1: h-p(R1(A, Task, A1), R2(A, Task, A2))←market_share_increase_need_period(P)
C1: h-p(R2(A, Task, A2), R1(A, Task, A1))← image_improvement_need_period(P)

According to T4 and his background theory, agent α will choose the achievement of the goal G_3, which corresponds to the choice of the agent α_1.With this choice agent shows his professionalism because he has made a choice, which is good for the company (i.e. satisfaction of the motivation m_3).

In order now to show the influence of the personality on the cooperation capability, we assume that agent α can also use the theory implementing his personality, in order to choose a partner. Evaluating now the choices under his personal criteria, we assume that the part of the background knowledge of the personality module, concerning the cooperation module is such that N5 and N3 hold.

Thus if we consider a rather ambitious agent he could be equipped with the following theory in the personality level:

R^1_{53}: h-p (G_5, G_3) ← N5
R^1_{35}: h-p (G_3, G_5) ← N3
H^1_{53}: h-p (R^1_{53}, R^1_{35}) ← true
E^1_{35}: h-p (R^1_{35}, R^1_{53}) ← job_loss(self, G_5)
C^1_{35}: h-p (E^1_{35}, H^1_{53}) ← true

According to the above theory and his background knowledge, agent α will prefer the achievement of the goal G_5, which corresponds to the choice of the agent α_2. This can be only overturned if the given choice G_5 will lead to the bankruptcy of the company and therefore to his loosing his job according to the following knowledge which could appear in the non-defeasible part of the agent's theory T4:

$$\text{job\_loss}(\text{Employe}, G) \leftarrow \text{company\_bankruptcy}(G), \forall \text{Employe}, \forall G$$

We can therefore imagine the case where, the choice of a manager α_1 is imposed by the current circumstances (because of the competition and the financial crisis) and thus the non-choice of the manager could lead even to the bankruptcy of the company having as consequence the loss of his job.

Therefore according to the background knowledge of the agent and taking into account that no such exceptional conditions appear in it, the agent following his personality, will choose an expert (i.e. α_2) to satisfy the motivation m_5 (i.e. agent α considers that the choice of an expert serves his own ambitions). Thus given that his professional policy had indicated to him the opposite, he will be found in a dilemma.

By concluding, we believe that the problem of dilemma, presented in situations where the decision policy and the personality give contradictory results, could be resolved following different approaches. The first one is to apply a multi-criteria method [Vincke, 92] that allows the evaluation of the two options according to several criteria. The second one is to apply an ad hoc method, which would give to the designer the possibility, according to the specific application, to decide whether the (professional) decision policy or the personality is stronger.

2.2.4. Future Work

Our future work on argumentation can be classified upon several dimensions. One of them concerns the use of our argumentation framework in order to model different types of dialogues presented in the literature [Walton & Krabbe, 95], such as negotiation, deliberation, persuasion, etc. (this is also related to our work on agent conversation, presented in Section 3.1). Relative to this issue is the evolution of the argumentation-based negotiation model we have proposed (our work on negotiation is presented in Section 3.3).

Another dimension is related to the different personalities of agents by modeling different agents personalities with respect to the way that they address their needs. In this direction, we need to extend the framework to allow an agent to decide amongst goals that concern more than one need simultaneously. A study of the formal properties of the labeling of goals according to motivations is needed together with a further study to specify a learning (or other) procedure for the labeling. It is also important to study, in relation with the social aspect of the agents that is inherent in our work, how these different personalities play a role in the interaction among agents in order to form heterogeneous societies.

2.3 Dynamic Planning

As mentioned before, in order to achieve a goal, planning is one of the capabilities an agent can have. The proposal of a formal model of dynamic planning is another feature of my research related to the effort of proposing a formal model of an agent. As explained, the adoption of the modularity for an agent's architecture allows us to apply formal models (based to decision theory, logic or combination of both) in order to implement the different capabilities of an agent without any problem of consistency. In this section, we discuss the formal model of dynamic planning presented in [Moraïtis & Tsoukiàs, 00; Moraïtis & Tsoukiàs, 99; Moraïtis & Tsoukiàs, 02a].

Dynamic planning concerns the planning and execution of actions in a dynamic, real world environment. Its goal is to take into account changes generated by unpredicted events occurred during the execution of actions. According to our approach, changes can come both from a dynamic environment and from the agent himself. Several works are proposed in the so-called "reactive planning" field in order to address planning in a dynamic environment under different approaches (e.g. [Firby, 94; Gat, 92]). Such works propose different techniques in order to react to environmental changes, which may occur during the execution process.

We adopt a more general approach since we consider that, in addition, any change may occur in agent's behavior (for any reason, i.e. according to a possible user suggestion) during the execution process, pushing him to change his preferences and consequently his actions or his method to evaluate these preferences. Changes on agent's preferences and on his evaluation methods, are taken into account as revision of three specific structures called possible plans, efficient plans and best plans. To model these structures, we have used graphs inspired by the ones we described in [Moraïtis & Tsoukiàs, 96]. Preferences are modeled as criteria in the multi-criteria planning problem we consider. This formalism allows us to present this planning problem as a multi-objective dynamic programming problem. Using dynamic programming in planning problems dates back to Bellman [Bellman, 57], but its use in agency theory has been limited in search algorithms, (see [Stenz, 95]) or in the frame of "universal planning" algorithms (see [Schoppers, 87]). Under such a perspective the model we propose allows an agent based on the set of possible actions to achieve a fixed goal, to express his preferences about the benefit he desires to take out (for example, profit, time, pleasure, etc.) by achieving this goal and consequently to define the efficient actions for this end.

Further on by introducing some additional information concerning his preferences, it is possible to define the best plan as the preferred compromise. During the execution of a single action the agent may modify his evaluations (a revision is necessary) or the world may be modified after an unanticipated event (an update is necessary). Such changes (how these are perceived is not considered yet in our work) may invalidate the plan under execution in the sense that it could be impossible to follow it or it could be no more convenient. So, the aim of our dynamic planning model is to take into account such changes and to decide what the agent should do.

2.3.1 The Multi-Criteria Planning Model

Let's consider that each agent A_g has to accomplish a set $\mathcal{T}$ of tasks in order to accomplish a fixed goal. Each task t_i can be decomposed in subtasks necessary to achieve t_i. We can consider that an agent has to go through a set of "states of the world" and more precisely from a state where no task is accomplished (the "nil" state of the world) to a state where all tasks are achieved and therefore his goal is achieved (the "final" state of world). We represent such a situation as an oriented graph. The agent has to execute some actions in order to accomplish his tasks. Each time an action is executed the agent perceives some consequences (for instance a resource is consumed, a distance is computed, a profit is reached etc.). Therefore each time a subtask is achieved the agent is able to register the level of associated consequences on a set of attributes on which he might be able to express his *preferences*.

The available information in this planning model consists in:

- a set $\mathcal{T}$ of tasks t_i necessary for a fixed goal achievement;
- a set $\mathcal{S}$ of possible states of accomplishment s_{li} for each task t_i;
- a set $\mathcal{A}$ of possible actions a_j
- a set $\mathcal{H}$ of partial orders $\succcurlyeq_q$ on the set $\mathcal{A}$ ($x \succcurlyeq_q y$: the action x is at least as good as the action y on the partial order $\succcurlyeq_q$); if some of such partial orders on the set $\mathcal{A}$ are at least weak orders, then there exist real values functions g_q, one for each such weak order. We represent with $g_q(a_j)$ the consequences of adopting action a_j under the *preference* g_q
- a set $\mathcal{P}$ of the possible sequences of actions (*plans*: denoted by ϕ, χ, ψ, etc);

Finally, we consider that it is possible to define a set $\mathcal{G}$ of binary relations $\sqsupseteq_r$ on the set $\mathcal{P}$ ($\chi \sqsupseteq_r \psi$: the plan χ is at least as good as the plan ψ on the relation $\sqsupseteq_r$). For the moment, the hypothesis made is that each such binary relation is reflexive ($\forall \chi \in \mathcal{P}$, $\chi \sqsupseteq \chi$). This model considers that is possible to establish the relations $\mathcal{G}$ on the set $\mathcal{P}$ from the partial orders $\mathcal{H}$ on the set $\mathcal{A}$.

A concept of "world state" is also introduced: a state w is a collection of propositions, predicates and/or functions $\langle \Psi, \lambda, \pi_r \rangle$ where: Ψ: is a set of descriptions (under form of propositions) specifying what is true in that state of the world; $\lambda \subseteq \mathcal{T} x \mathcal{S}$: is a binary relation associating a task t_i to an accomplishment state s_{li}; π_r: $\mathcal{P} \rightarrow \mathcal{R}$: are functions mapping the set $\mathcal{P}$ of possible sequences of actions to the reals, representing the binary relations $\sqsupseteq_r$. Of course, such functions exist iff the corresponding relations are at least weak orders (complete and transitive). Often, π_j are computed using the evaluations g_j (a typical case is $\pi_j(p)=\sum_{a_i \in p} g_j(a_i)$).

Definition 1- Possible paths graph: contains a start node corresponding to a nil state (none sub-goal is accomplished), an end-node corresponding to a given goal to achieve and a set of intermediate nodes corresponding to intermediate states of the world. Arcs correspond to the set of possible actions an agent can perform to achieve his goal through several sub-goals achievement. We denote the possible paths graph as $\Gamma_P = \langle \mathcal{W}_P, \mathcal{A}_P \rangle$. It should be noticed that Γ_P, operationally, is just a data-base describing the possible states of the world and the arcs among them. It consists in a $\mathcal{W}_P \times \mathcal{W}_P$ matrix with 0/1 entries, denoting the existence of an action between any two possible states of the world.

Definition 2-Efficient paths graph: represents the set of efficient paths among the possible paths, computed according to the agent's private goals. It represents all "efficient" (not dominated) ways to achieve the agent's work goals. Generally it is impossible to find a path that will be the best for all the agent's private goals, (this is an elementary notion in multi-criteria decision aid, see [Vincke, 92]). It is clear however, that there exist paths that are definitely dominated by other ones, in the sense that they are worse under all points of view (all preferences). Let's introduce a dominance relation $\gg$. Given any two possible paths p, p': $p \gg p' \Leftrightarrow \forall\ k\ p \sqsupseteq_k p'$ and $\exists k^*$: $p \sqsupset_{k^*} p'$. The set of efficient paths $\mathcal{D}$ will therefore be the set of paths which are not dominated: $\mathcal{D}=\{p: \neg\exists p' \in \mathcal{P}: p' \gg p\}$. We denote the efficient paths graph as $\Gamma_E = \langle \mathcal{W}_E, \mathcal{A}_E \rangle$. Clearly $\Gamma_E \subseteq \Gamma_P$.

Definition 3-Best paths graph: represents the best compromise solution among the efficient paths according to some further additional information (as for instance, an importance relation among his preferences). We make the hypothesis that the agent has such kind of information and therefore he is able to identify a plan p^* such that $\forall p \in \mathcal{D}$, $\Delta(p^*, p)$, Δ representing a weak order on the set $\mathcal{D}$. Under the hypotheses done in this paper, there exist a lot of procedures to identify the "best" compromise solution among the efficient ones [see Climaco & Martins, 82; Hansen, 80; Henig, 94]. We denote the best paths graph as $\Gamma_B = \langle \mathcal{W}_B, \mathcal{A}_B \rangle$. Clearly $\Gamma_B \subseteq \Gamma_E$.

2.3.2 Classification of Possible Changes and Reaction

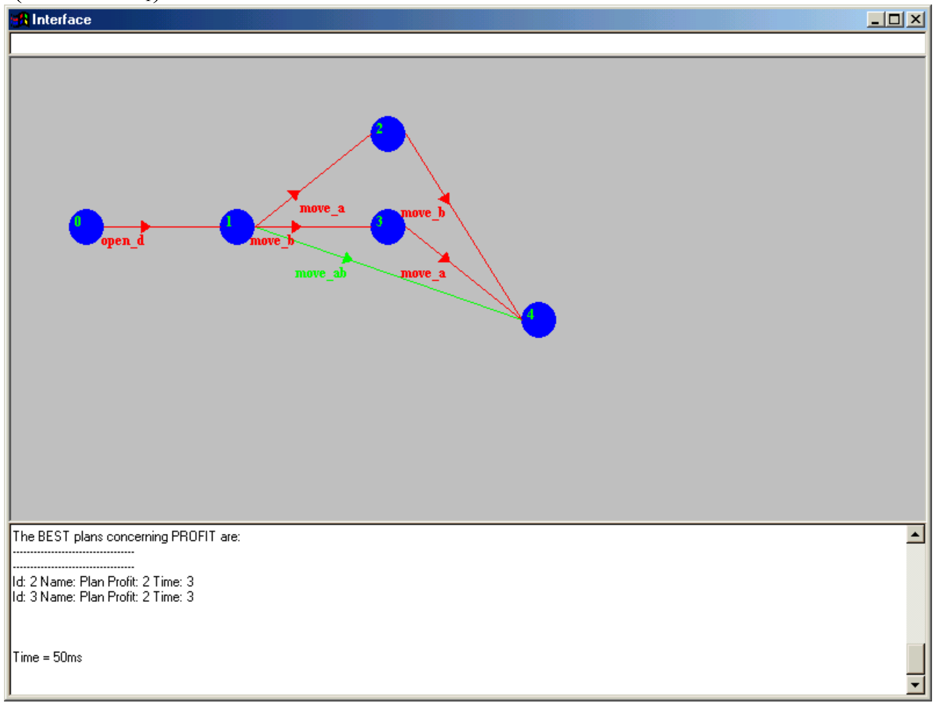
During a plan's execution different events may occur such that the agent may modify his evaluations (a revision is necessary) or such that the world is modified (an update is necessary). It is possible that such revisions or updates (hereafter called changes) may invalid the plan under execution. So, what should the agent do? When a change happens, the agent will recognize it and react either following an alternative best plan or constructing a new one. The new plan will have as initial state the state that the agent had reached before the interruption and as final state the same as before (i.e. where the fixed goal is achieved). Depending on the kind of changes, the agent will adopt the most suitable reaction.

In order to understand how our model works, let's consider the following example. An agent α_1 has the goal to move outside of a room two objects a and b. To do this, he must first open the door of the room, which is currently closed. The agent can move the objects outside either one by one or both of them at the same time. So, the actions he is able to perform are: move(a), move(b), move(ab), open(door). The preferences of the agent are min-time and max-profit. We assume that the first three actions leave a profit of 1 unit, while they generate a loss of 1 time unit. For the action open(door), we assume that it does not leave any profit, but it generates a loss of 1 time unit.

In the following we present the classification of possible changes we have considered namely "best" paths revision, "efficient" paths revision, "possible" paths revision. The algorithms implementing these revisions are presented in [Moraïtis & Tsoukiàs, 00; Moraïtis & Tsoukiàs, 02a].

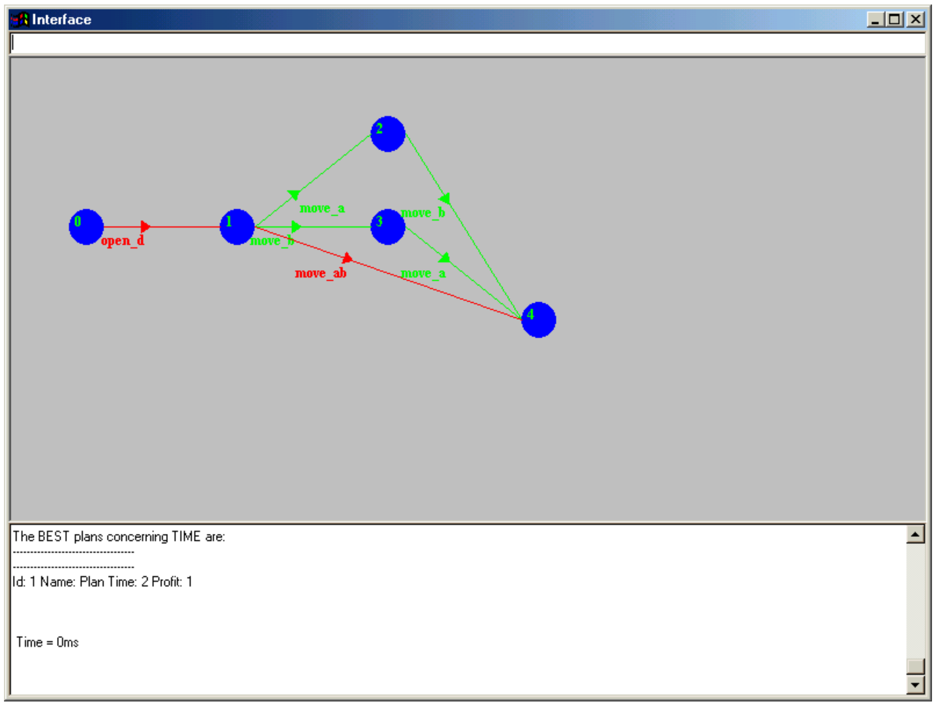
C_1: Best Paths Revision. It concerns the weak order Δ. For different reasons the agent may modify the weak order under which the specific best plan has been chosen among the efficient ones.

The best plans of the agent considering that his preferred goal is max- profit appear in Figure 2, while in Figure 3 we have considered that the agent has modified the priorities or importance of his preferences, by choosing now to satisfy the min-time goal (i.e. case C_1).



```
The BEST plans concerning PROFIT are:
--------------------------------
--------------------------------
Id: 2 Name: Plan Profit: 2 Time: 3
Id: 3 Name: Plan Profit: 2 Time: 3


Time = 50ms
```

Figure 2: Best plans of agent for max-profit goal



```
The BEST plans concerning TIME are:
-----------------------------------
-----------------------------------
Id: 1 Name: Plan Time: 2 Profit: 1


 Time = 0ms
```

Figure 3: Best plan of agent for min-time goal

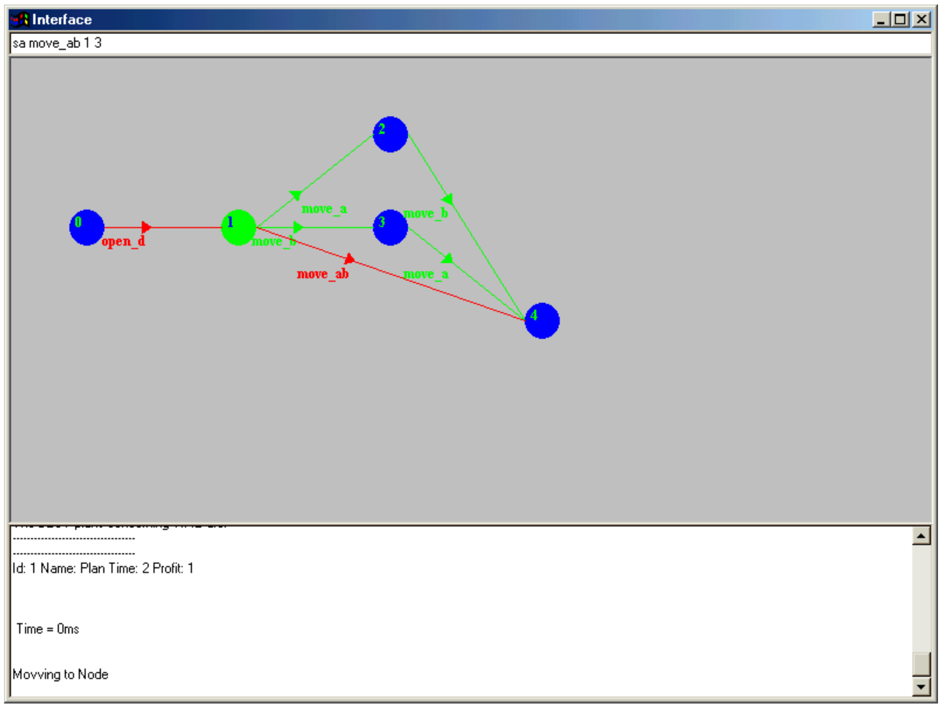
C_2. Efficient Paths Revision. It concerns the states of the world and particularly the functions π^k_j. Actually the way by which the agent evaluates the actions and therefore the plans, as far as his preferences are concerned, can change (for instance the agent may realize that some actions are "more expensive" from what has been considered at the beginning, (i.e., consider that the agent, having chosen the plan for satisfying the goal min-time and being on the node 1, notices that the action move(ab) generates a loss of 3 time units, while it has been considered before that it generates a loss of 1 time unit see Fig. 4). Under this point of view the efficient graph Γ_E could be modified (although not strictly necessary) since a path considered efficient may not be any more, while a dominated path may become efficient. From the execution point of view the following possibilities may occur:

C_{21}: the present best path is not more efficient and therefore is not any more the best compromise (see Fig 5).

C_{22}: the present best path is still efficient and but is no more the best compromise

C_{23}: the present best path is still efficient and the best compromise. Obviously only the first two cases may affect plan's execution.

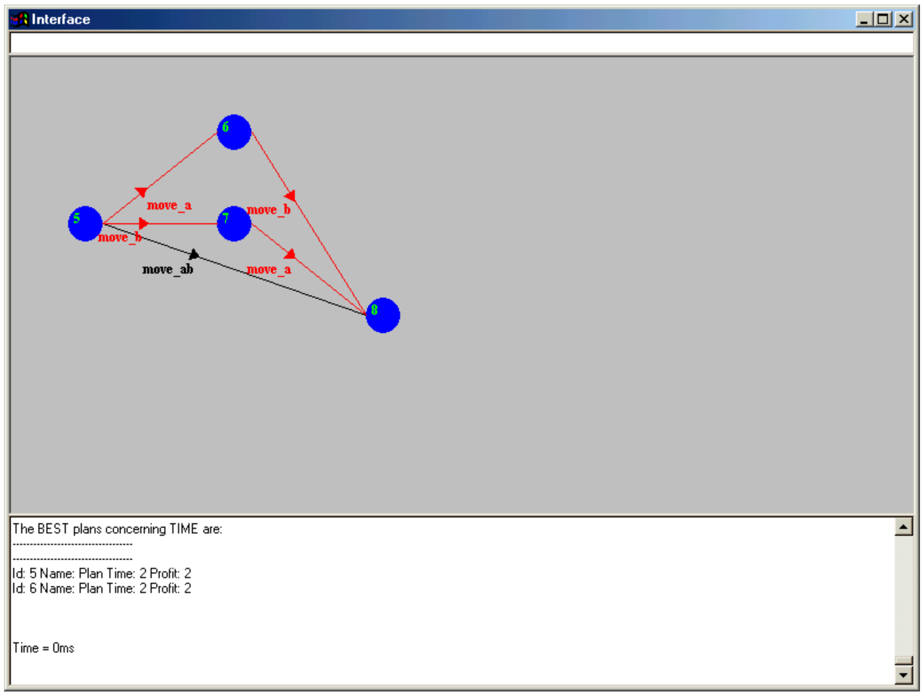
In Figure 4, we show how we can change the characteristics of the actions of the agent. More particularly, we modify the cost of the action move(ab) in time by simulating the fact that the agent can discover, during the execution of a plan (i.e., 0-1-4), that an action is more expensive from what has been considered before.



sa move_ab 1 3

```
-----------------------------------
-----------------------------------
Id: 1 Name: Plan Time: 2 Profit: 1


 Time = 0ms

Movving to Node
```

Figure 4: Agent on node 1



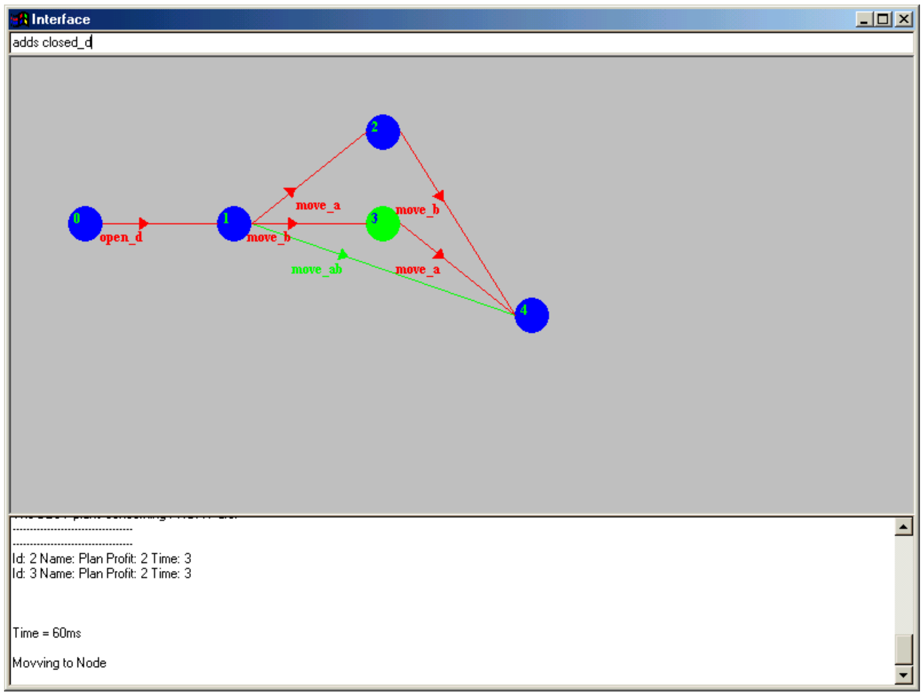
```
The BEST plans concerning TIME are:
-----------------------------------
-----------------------------------
Id: 5 Name: Plan Time: 2 Profit: 2
Id: 6 Name: Plan Time: 2 Profit: 2


Time = 0ms
```

Figure 5: New efficient and best paths

Figure 5 presents the situation (i.e. change C_{21}) where the plan considered before is not any more efficient and, therefore, is not the best any more. In this figure, the red arcs correspond to the new, simultaneously efficient and best, plans.

C_3: Possible Paths Revision 1. One or more possible actions from the set $\mathcal{A}$ can be eliminated. Such modification affects the possible paths graph Γ_E and therefore can affect the efficient paths graph Γ_E and the best paths graph Γ_B. The possible consequences of such a change are the following:
C_{31}: some states of the world are modified as far as the functions π^k_j are concerned (the sequences under which a state can be reached are now different; the values of some π^k_j can be modified). The considerations of point 2 apply here. For example, consider that agent discovers that he is unable to perform the action move(ab) (Fig. 4) because objects a and b are finally together too much heavy to be simultaneously moved. Under this possibility we include also the case where action(s) eliminated belong to the best plan. That means that some states of the world which have been foreseen to be reached under certain conditions remain reachable, but under new conditions.
C_{32}: a state of the world becomes unreachable because all the actions leading to such a state are eliminated. If such a state belongs to the best plan then the agent has to reconsider the ongoing execution, otherwise the change will not affect his behavior. We call such a state as "infeasible state" and we denote it as $w^{\perp}$ (the state of node 4 (Fig. 4) if agent is unable to perform the action move(ab)).
C_{33}: a state of the world becomes a "cul-de-sac" in the sense that all actions (arcs) leaving this state (node) are eliminated. Again a problem will arise only if such a state belongs to the best plan. We call such a state an "infeasible state" and we denote it as $w^{\perp}$ (i.e. the state of node 1 (Fig. 4) if agent is unable to perform the action move(ab)).

C_4: Possible paths revision 2. It concerns the availability or necessity of one or more actions, which before were impossible or unforeseen. Again such a modification affects Γ_P and therefore Γ_E and Γ_B. The possible consequences are the following:
C_{41}: some states of the world are modified as far as the functions π^k_j are concerned. A node that was reachable for a certain value of the function π^k_j is now reachable for new values (possibly better). Under such a perspective the new action will connect nodes that in the original possible paths graph were not adjacent. A problem will arise only if the modified states of the world belong to the efficient paths graph and can influence the best path graph.
C_{42}: the new actions(s) may create a state of the world, which was not considered in the set W (for instance the new action may correspond to the necessity to accomplish a new task or subtask, which was not considered before). For example, if an unpredicted event (the door is closed) occurs at the moment when the agent is in node 3 during the execution of the path 0-1-3-4 (Fig.6), the agent will have to open the door (see Fig. 7) before executing the remaining action of his current plan (i.e. move(a)) .

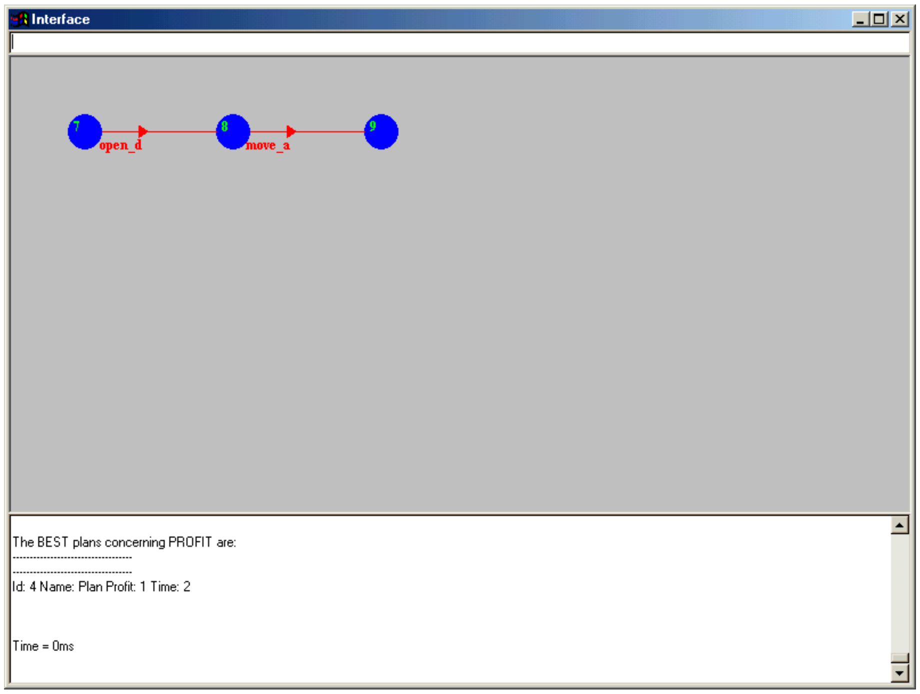


adds closed_d

```
-----------------------------------
-----------------------------------
Id: 2 Name: Plan Profit: 2 Time: 3
Id: 3 Name: Plan Profit: 2 Time: 3


Time = 60ms

Movving to Node
```

Figure 6: An unpredictable event arrives

.



```
The BEST plans concerning PROFIT are:
-----------------------------------
-----------------------------------
Id: 4 Name: Plan Profit: 1 Time: 2


Time = 0ms
```

Figure 7: New best plan

Figure 6 presents how we can generate an unpredictable event (i.e. the door is closed) that modifies the current state of world. The new state of the world, not considered in the beginning, is (OUT(b), IN(a), closed(door)). A problem will arise only if the new action(s) and the new state(s) of the world may belong to a path, which compared to the best path, can be considered as a better one. Figure 7 presents the new plan, generated after the arrival of an unpredictable event that leads to the execution of a new action (i.e. open(door)) and, therefore, to the generation of a new state of the world. This is necessary in order for the execution of the remaining action (i.e. move(a)) to be possible.

2.3.3 Future Work

We believe that this work highlights interesting issues by having proposed dynamic planning as a useful mean to reason about changes generated not only by the environment, but also by the agent himself. Our future work will concern the problem of how to detect the changes occurred and how to classify them according to the categories defined in our work, as well as its integration in the existing model.

Another direction will be to improve the planning model by studying different ways to reduce the search space (e.g., by considering maximum or minimum thresholds in the evaluation of the plans, which is based on the agent preferences)

L’ Agent comme une Entité Sociale

Résumé

Dans cette section je présente mon travail concernant l’agent comme une entité sociale. En accord avec cette dimension l’attention est focalisée sur les interactions qu’un agent peut avoir avec les autres agents comme un membre d’une societé qu’il partage avec eux. La littérature de l’Intelligence Artificielle Distribuée, considère la coordination comme la forme pricipale d’interaction. L’objectif de la coordination est de donner aux agents qui interagissent, la possibilité d’atteindre ou d’éviter des situations selon qu’elles sont désirables ou à éviter pour un ou plusieurs parmi eux.

La coordination des actions peut être assurée par différents moyens. Un moyen est celui de la planification distribuée. Dans ce cas les agents essayent de trouver un plan collectif cohérent (dans un contexte coopératif) ou de détecter des conflits possibles parmi les actions individuelles. Dans la dernière situation, le but est de résoudre ou d’ éviter les conflits (dans un contexte d’agents individualistes), de telle façon que les agents puissent exécuter leurs actions (afin d’achever leurs objectifs) sans interférences négatives sur les objectifs des autres.

Un autre moyen de coordination est celui de la négociation. Dans ce cas, les agents essayent d’habitude de trouver un compromis afin d’achever, le mieux possible, leurs objectifs respectifs (cas des agents individualistes) ou de trouver la meilleure façon d’achever ensemble un objectif commun (cas des agents coopératifs).

Cependant peu importe la procédure choisie, les agents doivent communiquer. Plus précisèment, ils ont besoin d’un protocole de conversation qui va supporter l’échange de la connaisance et des informations qui sont nécessaires dans le cadre du mécanisme de la coordination choisie (p.ex., des plans dans cadre de planification distribuée, offres et contre-offres dans un cadre de négociation, etc.).

Alors plus précisément, dans cette section je présente mon travail en planification distribuée, négociation et conversation.

Le travail en planification distribuée exploite notre modèle de planification multi-critère présenté dans la Section 2. Notre approche est fondée sur un cycle Planification-Négociation-Execution (PNE) où chaque agent planifie, négocie, et execute ses propres plans tout seul. La situation que nous considérons est celle où des agents essayent de satisfaire des objectifs de travail (nécessaires pour un objectif global) tout en essayant de satisfaire des objectifs personnels. Les agents développent leurs plans en utilisant le modèle de planification multi-critère, ils élaborent les plans possibles pour les objectifs de travail et ensuite ils évaluent les plans efficaces en utilisant les objectifs personnels comme des critères d’évaluation. Ensuite, les plans efficaces sont échangés entre les agents et une fusion de plans est effectuée. Cette fusion consiste en une recherche

combinée sur le comment ces plans pourraient coexister. La détection et la résolution des conflits est effectuée par un algorithme de fusion que nous avons proposé. Le résultat de cet algorithme est un ensemble des plans multi-agents possibles, libres de conflits où apparaisent les actions que les agents peuvent éxecuter en parallele et de manière asynchrone. Ces plans permettent tous la résolution des objectifs de travail de chaque agent et par conséquent de l'objectif global et ils sont répresentés sur un graphe orienté que nous appellons graphe de négociation. Cependant ces plans sont évalués différemment par les agents leur évaluation étant fondée sur leurs objectifs personnels. Ceci est une différence entre notre approche et celles qui apparaissent dans la littérature. Un autre type de conflits est ainsi generé. Ces conflits sont résolus maintenant par négociation cherchant un compromis entre les agents qui essayent simultanément de satisfaire au mieux leurs objectifs personnels.

La négociation est fondée sur les modèles de négociation que nous avons proposés et qui sont aussi présentés dans cette section. Tous ces modèles intégrent la dimension multi-critère et sont appliqués au cas de la planification distribuée (en utilisant comme base de négociation le graphe de négociation). Néanmoins ils peuvent être appliqués à des situations plus générales. Par exemple un de ces modèles est fondé sur une approche d'aggregation-desaggregation. Selon cette approche, les agents sont équipés d'un modèle de prise de décision multicritère, font des offres, des contre-offres en utilisant un modèle individuel de préférences, et essayent de trouver un accord, en incluant dans leur modèle à chaque tour de négociation dans leur modèle, une estimation du modèle de décision de leur contre-partie. L'estimation est fondé sur un modèle de régression linéaire multiple. La négociation prend fin lorsque les agents font la même offre.

Finalement dans cette section je présente mon travail en protocoles de conversation entre agents. Dans ce travail nous avons proposé un cadre fondé en logique pour modéliser des dialogues complexes entre des agents autonomes. Ce modèle s'appuie sur ma conception d'architecture modulaire pour les agents. Ainsi chaque module est responsable pour un certain type de dialogue (p.ex., négociation, persuasion, délibération, etc.), et le comportement global est le résultat de l'interaction entre les différents modules. Dans ce travail nous définissons un ensemble de performatifs (intégrant des actes illocutoires comme proposer, accepter, refuser, etc.) de communication, une base de connaissance qui implémente chaque module (et où la connaissance est répresentée en logique de prédicats de premier ordre), un ensemble de poliques de dialogue qui permettent la génération automatique de dialogues et une combinaison de raisonnements monotones, par chainage-arrière et par chainage-avant. Le raisonnement par chainage-arrière a comme but la satisfaction des objectifs générés de l'agent, alors que le raisonnement par chainage-avant a comme but de "batir" les réponses appropriées aux performatifs reçus durant un dialogue (en exploitant les politiques de dialogue).

3 The Agent as a Social Entity

The social dimension of an agent is of equal importance to his individual one. According to this dimension, the basic focus of attention is not on the capabilities of the agent and his efforts to pursue his goals by executing actions that optimize some given preferences. On the contrary, the focus of attention is on the interactions that an agent can have with other agents, as a member of a social environment that he "shares" with them. However, "interacting" indicates that the agents may be affected by other agents or perhaps by human agents in pursuing their goals and executing their tasks. The Distributed Artificial Intelligence literature (e.g. [Weis, 99]) considers coordination as the main form of interaction that is particularly important with respect to goal achievement and task completion. The purpose of coordination is to give the possibility to interacting agents to achieve or avoid situations that are considered desirable or undesirable by one or several of them. To coordinate their goals and tasks, agents have to explicitly take into consideration the dependencies that may exist among their actions. Coordination may concern agents that may are cooperative, self-interested or bluntly competitive.

In the first case, several agents work together as a team while, by assembling their capabilities and their knowledge, they try to achieve a common goal. Agents are forced to cooperate because nobody has the possibility to achieve the given goal alone. Therefore, coordination is useful since, for instance, it may allow agents detect situations where working together is better than working individually or avoid the execution of the same actions by several agents (in the case that these actions are useful for all). However, there exist situations (as in many human organizations [March & Simon, 58]) where agents try to achieve a common goal by simultaneously trying to optimize individual goals.

In the case of self-interested agents, interacting agents have their own goals, which could be conflicting (e.g. their achievement presupposes the use of the same resources). In this case, coordination is useful because it allows agents to detect harmful interactions before the execution of their actions and search (for example through synchronization) a "modus-viventi", which would allow them to achieve their goals by avoiding or resolving conflicts.

Finally, we can have situations where agents are bluntly competitive. That means that an agent tries to maximize his own benefit (profit) at the expense of others, and therefore, the success of this one implies the failure of others.

Coordination of actions can be reached through different ways [see for example Weiss, 99] and therefore, it can encapsulate different forms of interaction among involved agents. One of them is through distributed planning [see for example Durfee, 01]. In this case, agents try to find a consistent collective plan (i.e. in cooperative context) or detect possible conflicts among individual actions. In the later case, the aim is to resolve or avoid the conflicts (i.e. self-interested context), in that agents can execute their actions (in order to achieve their goals) without causing negative interferences on the goals of the others. In the sequel, I will outline our model of distributed planning presented in [Moraïtis & Tsoukiàs, 96; Moraïtis & Tsoukiàs, 02b].

Another way to assure coordination is through negotiation. In this case, agents usually try to find a compromise in order to achieve as better as they can their respective goals (i.e. in a self-interested context [e.g. Zlotkin & Rosenchein, 91]) or to find the best way to achieve together a joint goal (i.e. in a cooperative context [e.g. Lander & Lesser, 92]). In the following, I will also present our work on negotiation presented in [Moraïtis & Tsoukiàs, 96; El Fallah-Seghrouchni, Moraïtis & Tsoukiàs, 99; El Fallah-Seghrouchni, Moraïtis & Tsoukiàs, 00; Moraïtis & Tsoukiàs, 02b; Kakas & Moraïtis, 03]. The distributed planning and a part of the proposed negotiation models are both based on multi-criteria decision theory (e.g. [Vincke, 92]).

Whatever the coordination procedure chosen is, agents must communicate. More precisely, they need a conversation protocol that will support the exchange of knowledge and information that is necessary in the framework of the chosen coordination mechanism (e.g. plans in a distributed planning framework, offers and counter-offers in a negotiation framework, etc.) In the next section, I will present the conversation protocol we have proposed in [Karacapilidis & Moraïtis, 02a; 02b].

It is obvious that the properties of the coordination mechanisms depend on the characteristics of the involved agents (i.e. cooperative, self-interested, competitive). At the same time, having defined a specific organizational structure with the ubiquitous coordination mechanisms, any new agent integrating this structure must have the appropriate profile (or acquire it) in order to be able to respect the behavioral rules characterizing the organization (e.g. an self-interested agent could not survive for a long time in a cooperative environment; if he wants to stay, he must change by adopting a more cooperative profile). This is the expression of the "individual/social" duality that is the property that, as already said, we consider as the most exciting one.

3.1 Agent Conversation

This work proposes a logical framework for modeling of complex dialogues between intelligent and autonomous agents [Karacapilidis & Moraïtis, 02a; 02b]. The overall approach builds on the general assumption that I have adopted in my work, which is that an agent is composed of a set of modules, each of them being responsible for a particular feature of the agent's overall behavior. Such features may concern, abilities, such as deliberation, negotiation, persuasion, etc. [Walton & Krabbe, 95]. The original idea here is that each such module performs the dialogue associated to the ability which this module is responsible for. Thus, embedded dialogues (involving different types of dialogue) are expressed as moves from one dialogue type to another and performed through the interaction of the different modules an agent consists of.

For instance, the role of an agent's deliberation module is to have a dialogue with its peers in order to decide a sequence of actions in some situations, while the role of its negotiation module is to negotiate with another agent about a specific goal. The above assumption makes our approach different compared to existing work on developing frameworks for conversational agents. It should be made clear here that one may build an agent according to his particular interests in a specific application; that is, an agent Ag_x may be composed of just a negotiation module, while another one Ag_y may also include a deliberation and an information seeking module. Note that each agent has only one instance of the module types mentioned above. The only constrain that we impose in the design of such kind of agents is that all messages exchanged between two conversational agents pass through their communication modules.

Each of the above modules is triggered whenever it is necessary to play the specific role it is conceived for, thus performing a dialogue corresponding to its "area of expertise". The idea is that the reaction of an agent to an input received, or his global action towards achieving a goal, is based on a sequence of actions performed by one or more of his modules. In other words, each agent's module is associated with a certain part of the overall dialogue.

This work falls in the area of definition of conversation protocols and formal frames supporting different dialogue types for agent communication (e.g. [Amgoud, Parsons & Maudet, 00; Parsons & Jennings, 96; Sadri, Toni, & Torroni, 01; Hitchcock, McBurney & Parsons, 01]). Much attention has been paid in keeping our framework as operational as possible. The architecture of our agents and their conversational protocol are fully implemented and thoroughly interrelated. Agents in our framework operate combining two types of reasoning to regulate the continuity of dialogues. More specifically, they perform backward reasoning aiming at satisfying the goal generated upon the reception of a communication performative, and forward reasoning to build the appropriate answers to the performatives received. The algorithm implementing the execution cycle of the agent is presented in [Karacapilidis & Moraïtis, 02a].

In the framework of this work, we have adopted the same structure for each module an agent can have. Thus in order to serve its role, we assume that each module δ_x is equipped with a knowledge base $K(\delta_x)$. The knowledge conveyed is expressed in a declarative way, as described below:

Definition 1. A knowledge base is a tuple <F, G, A, solver, DP, PR, messenger, RF, D>, where:

- F contains application-specific knowledge (facts) related to the role of the module and the specific topics.
- G is the goal to be achieved (represented by sentences).
- A is the set of possible actions (represented by if-then rules).
- solver is an application-independent inference engine that exploits facts and actions to reach a goal. It is activated whenever a new goal G′ replaces the existing goal G. It is based on a backward reasoning mechanism.

- DP is a set of application-independent knowledge, namely dialogue policies, which are represented by if-then rules (see below) and used by the messenger to regulate the dialogues.
- PR is a set of preference relations $>^{pr}$ on the set of F and on the set of A.
- messenger is an application-independent inference engine that filters the received messages and permanently consults the existing dialogue policies and preference relations by exploiting a forward reasoning mechanism.
- RF is a list of the reasons (facts or actions) leading to the failure of the current goal.
- D contains the messages exchanged during the current dialogue. It is implemented as a queue that is emptied after the end of each dialogue.

Conversation between agents is therefore based on dialogues. A dialogue between agents can be a complex process, taking place through the exchange of messages conveying communication performatives. In a dialogue context, each time an agent receives a message, it has to know immediately which module it must be activated in order to set up the appropriate dialogue.

Definition 2. A message is an instance of a schema of the form Msg=(id, P), where P declares the performative conveyed. In our framework, it is P=S (x, y, σ, T), where:
id is the message's identification number;

- S is an illocutionary act belonging to the set {propose, accept, request, assert, refuse, challenge, reject}. It should be noted here that this set corresponds to the current implementation of our framework and can be easily changed according to a particular setting;
- x and y are the sender and the receiver of the performative, respectively;
- σ is the subject (i.e., body) of the performative, which may take one of the following forms:
  - -a tuple σ = <sentence [support]> where support consists of elements (facts, actions, etc.) expressing arguments supporting sentences. When no support is available (or necessary to be explicitly mentioned), its value is ∅;
  - -a dialogue context structure DC (see below);
  - -∅, meaning *"nothing to say"* representing the silence in a dialogue
- T is the time when the performative is uttered (times are actually timestamps of the related transaction).

In fact, the first of the forms proposed for σ may express any message content, provided that it respects first-order logic representation.

One of the most important aspects in a conversation protocol is the automated generation of the appropriate illocutionary acts, as well as the initiation and termination of a dialogue. To this end, we have defined a set of application-independent rules, namely dialogue policies, which are exploited by the forward reasoning mechanism and serve this automated generation. The above concept is similar to that of dialogue constraints [Sadri, Toni, & Torroni, 01]; however, our dialogue policies are associated with the specific profile of an agent, thus characterizing his personality and behavior.

Definition 3. For an agent Ag_y, a dialogue policy is a if-then rule of the form P (x, y, σ, T) ∧ C ⇒ P' (y, x, σ', T+1), where:

- P (x, y, σ, T) is a performative uttered at time T, P' (y, x, σ', T+1) is a performative sent at time (T+1) from the receiver of P (x, y, σ, T) to its utterer, and σ (σ') is the subject of the performative, as described above (the subject of P' is not always the same with that of P). The above concept is similar to that of dialogue constraints [Sadri, Toni, & Torroni, 01], which however correspond to integrity constraints in an abductive logic programming framework.
- C, hereafter referred to as condition.

The idea is that when a module of an agent receives a message related to a subset of the defined dialogue policies (see DP1, DP2, DP3, DP4, DP5 below), its subject σ is considered as a goal G^{σ}. The operation of the solver (which uses backward reasoning exploiting the if-then rules of the set A), corresponds to the reduction of G to sub-goals, which in turn correspond to the if parts (or premises) of the triggered rules. The satisfaction (or not) of these rules defines what DP will be triggered and consequently what is the condition C to be checked in order to choose the message to be sent. Otherwise, C depends on the type of the received message (see DP6, DP7, DP8, DP9). In other words, the dialogue policy is a procedure of entailment that defines what is the next message to be sent by an agent Ag_y, after the reception of a specific message coming from another agent Ag_x (this task is performed by the messenger). In the framework of this work we have proposed a set of dialogue policies corresponding to a (rather usual) profile and are formally presented below (condition *C* appears within the square brackets):

- **DP1:** request (x, y, σ, T) ∧ [$K(\delta_y) \vdash G^{\sigma}$] ⇒ assert (y, x, σ, T+1)
- **DP2:** request (x, y, σ, T) ∧ [$K(\delta_y) \nvdash G^{\sigma}$] ⇒ refuse (y, x, σ, T+1)
- **DP3:** propose (x, y, σ, T) ∧ [$K(\delta_y) \vdash G^{\sigma}$] ⇒ accept (y, x, σ, T+1)
- **DP4:** propose (x, y, σ, T) ∧ [$K(\delta_y) \nvdash G^{\sigma} \wedge \{\exists\, \sigma' \in K(\delta_y) \mid (\sigma' >^{pr} \sigma) \wedge (K(\delta_y) \vdash G^{\sigma'})\}$] ⇒ propose (y, x, σ', T+1)

The first three policies above are rather straightforward. The condition imposed in DP4 means that if $K(\delta_y)$ does not entail G^{σ} (i.e., the goal associated to the subject σ of the received performative), it is checked whether there exists another σ' belonging to $K(\delta_y)$ along with a preference in PR stating that σ' is preferable than σ, such that $G^{\sigma'}$ can be entailed by $K(\delta_y)$.

- **DP5:** propose $(x, y, \sigma, T) \wedge [K(\delta_y) \nvdash G^{\sigma} \wedge \{\nexists \sigma' \in K(\delta_y) \mid (\sigma' >^{pr} \sigma) \wedge (K(\delta_y) \vdash G^{\sigma'})\}] \Rightarrow$ refuse $(y, x, \sigma, T+1)$
- **DP6:** refuse $(x, y, \sigma, T) \wedge [\text{support}(\sigma) = \varnothing] \Rightarrow$ challenge $(y, x, \sigma, T+1)$

The meaning of the above is that an unsupported refusal, sent from agent x to agent y, triggers a challenge act from y (which actually asks x to justify his decision).

- **DP7:** challenge $(x, y, \sigma, T) \wedge [\exists\, \text{reason} \in RF(y) \mid \text{reason} \vdash (\neg\sigma)] \Rightarrow$ assert $(y, x, \sigma, T+1)$

This condition actually checks RF to verify whether there exists a reason of failure of the goal associated to the subject σ (i.e. a fact or an action that contradicts σ); if yes, the reason found is sent back to the utterer of the challenge act (as a support of σ).

- **DP8:** assert $(x, y, \sigma, T) \wedge [\text{support}(\sigma) \neq \varnothing \wedge \{\nexists\, \text{support}' \in K(\delta_y) \mid \text{support}' \vdash (\neg\text{support})\}] \Rightarrow$ accept$(y, x, \sigma, T+1)$
- **DP9:** assert $(x, y, \sigma, T) \wedge [\text{support}(\sigma) \neq \varnothing \wedge \{\exists\, \text{support}' \in K(\delta_y) \mid \text{support}' \vdash (\neg\text{support})\}] \Rightarrow$ reject$(y, x, \sigma, T+1)$

The meaning of condition DP8 is that if a support', which contradicts support, cannot be found then y has to accept the assertion of x. Otherwise (in DP9) y will reject it.

Definition 4. Given that agents convey knowledge in their constituent modules, as described in the previous section, and inspired by the work presented in [Reed, 98; Walton & Krabbe, 95], we define a dialogue context as a tuple $DC = (t, (\tau, M))$, where:

- t is the type of the dialogue ($t \in$ {deliberation, negotiation, persuasion,...})
- τ is the topic of the dialogue, that is, what agents discuss about
- M is the medium used for the dialogue, (e.g. messages exchanged between agents x and y either directly (denoted by Direct) or through a mediator z (denoted by med(z)), etc.)

3.1.1 Future Work

Our primary future work direction concerns the automation of moves from one dialogue type to another. Moreover, we plan to integrate agents' mental attitudes (beliefs, desires, intentions) in our framework.

Another direction concerns the definition of more properties for our framework and its enrichment with more performatives and dialogue policies (including policies for nested dialogues).

Finally, we plan to use argumentation-based reasoning (by exploiting our framework) in order to make the reasoning mechanism of our agents more powerful and handle cases where non-monotonic reasoning is necessary.

3.2 Distributed Planning

Distributed planning is an important subfield of Distributed Artificial Intelligence. As Durfee [Durfee, 99; 01] says, "it is something of an ambiguous term, because it is unclear exactly what is *distributed*". By adopting the terminology proposed by [Durfee, 01], it may concern the case where the planning is centralized while the execution is distributed (centralized planning for distributed plans, see e.g., [Lansky, 90]), the case where planning is distributed (distributed planning for centralized plans, see e.g., [Wilkins & Myers, 95]) while the final plan is the result of a collaboration among a variety of cooperative planning agents or, finally, the case (distributed planning for distributed plans, see e.g., [Kabanza, 95; Ephrati, Pollack & Rosenschein, 95]) where both the planning process and the resulting plans for execution are distributed.

My work in distributed planning [Moraïtis & Tsoukiàs, 96, 02b] exploits our multi-criteria planning model I have already presented in Section 2.3. Our perspective for distributed planning integrates elements from the last two-abovementioned cases and corresponds, as said in Section 3, to situations that characterize many human organizations. More precisely, we consider that agents accomplish work goals necessary for a global goal achievement, while simultaneously trying to optimize private goals (interests or motivations). Our approach is based on a planning-negotiation-execution (PNE) cycle, similar to the coordination process of Martial [v. Martial, 92], but in our approach all agents have the same capabilities. In our current model, we take into account only the stages of planning and negotiation. As it is also assumed in [Durfee, 99], we consider multiple agents formulating plans for themselves as individuals (i.e. as they were alone) and then ensuring that their separate plans can be executed without conflict. The difference with the general setting presented in [Durfee, 99], is that in our case agents have simultaneously two types of goals, namely work and private goals. In our setting, each agent conceives, negotiates and executes its plans (for example there is no coordinator agent to which the agents can transfer their plans or an executor agent who is unable to negotiate about plans). The agents develop plans using a multi-criteria planning model, calculate the "possible" plans for work goals and evaluate the "efficient" ones (i.e. the non-dominated plans, computed according to the private goals which are used as evaluation criteria). Then, "efficient" plans are exchanged among agents and a plan merging is elaborated, which consists of a combination search of how these plans can fit together.

3.2.1 The Plan Merging Procedure

The detection and removal of conflicts are made by the merging algorithm we have proposed (the detailed algorithm is presented in Moraïtis & Tsoukiàs, 02b), which builds totally ordered multi-agent plans achieving the global goal; in these plans, there appear pairs of actions (an action per agent) that can be executed in a parallel and asynchronous way. The resulting graph is called negotiation graph. The merging algorithm examines all possible combinations of efficient plans of two agents. More precisely, it considers all the actions of the efficient plans by pairs, starting by the first actions of the plans until no more actions appear in both plans and checks whether they can fit together or not. To this aim, the merging algorithm has to consider three situations.

In the first situation, available action for checking appears only in one of the two plans. That means that an agent has already executed all the actions that he has to, while the other disposes some actions yet to execute. In this situation, the merging algorithm forms a pair with the available action (i.e. of one agent) and the synchronization action of "wait" (i.e. for the other agent). That means that the second agent will be waiting until the other executes the remaining action. We consider that agents cannot leave the space of work before the joint goal is successfully achieved (this can be useful in dynamic environments, where one agent could help the other in unpredictable situations).

In the second situation, available actions for checking appear in both plans. That means that both agents dispose actions for execution. If both actions can be executed (i.e. if their pre-conditions are satisfied in the current state of the multi-agent plan under consideration), our algorithm verifies whether there exists a temporal constraint between them (i.e. if an action must be executed before the other), which would prevent them to be executed in parallel. If such a constraint exists, the algorithm forms a pair with the action, which has the priority and the synchronization action of "wait". Thus, the action that has the priority will be executed while the other will be replaced by the synchronization action "wait" and it will be taken into account in the next step of the merging procedure (it will be checked along with the next action of the other plan). That means that while an agent is executing his action, the other is waiting until the first one is finished. Otherwise (i.e. if no constraint exists), the algorithm forms a pair with the two actions, which means that both actions will be executed in a parallel and asynchronous way.

In the third situation, available actions for checking appear in both plans. Thus, both agents dispose actions for execution. However, in this case one action can be executed (i.e. his pre-conditions are satisfied in the current state of the multi-agent plan under formation) while the other cannot (i.e. his pre-conditions are not satisfied). The algorithm forms also a pair with the executable action and the synchronization action of "wait". Thus, the first action will be executed while the other one will be replaced by a synchronization action "wait" and it will be taken into account in the next step of merging procedure. As in the previous situations, that means that while an agent is executing his action, the other is waiting until the first one is finished.

3.2.2 An Example

In order to better present our model, let's adapt the example presented in Section 2.3.2 in a distributed context. We therefore consider that one more object c and another agent α_2 exist in the room. We assume that this object is fragile and also that object b is now on object a. Agent α_2 is careful and that is why he is in charge to transfer fragile objects. Thus, he will move object c outside of the room. Agent α_2 is also in charge to put down objects being on other objects. So, he must put down b in order for agent α_1 to be able to move outside the objects a and b. In this new situation, agents α_1 and α_2 have a global goal, which is to empty the room, which means to move objects a, b and c outside of the room. This global goal will be achieved through the achievement of the work goals of the agents, which are to move outside the objects a and b (for α_1) and the object c (for α_2). Simultaneously, we consider that both agents have the private goals min-time and max-profit. We remind that the agents are able to perform the following actions: agent α_1: move(α_1, x) and agent α_2: putdown(α_2, y) and move(α_2, x). All these actions leave a profit of 2 units, while they generate a lost of a 1 time unit. We have also the relations: before(putdown(α_2, b), move(α_1, b)), before(putdown(α_2, b), move(α_1, a)).

Remark: the synchronization action "wait" has a cost of 1 profit unit while it generates a lost of 1 time unit. The lost of 1 profit unit is set because we want to push agents searching a compromise by avoiding (as much they can) the option of waiting.

Figure 8 represents the graph of efficient paths for agent α_2 while Table 3 presents how these paths are evaluated following the criteria profit and time modeling the private goals of the agents.

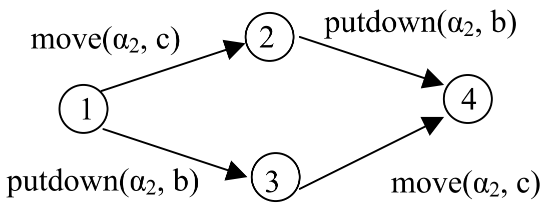


Figure 8: Efficient paths graph of agent α_2

Paths	Profit	Time
1-2-4	4	2
1-3-4	4	2

Table 3: Agent α_2 efficient plans' evaluation

Figure 9 represents the graph of efficient paths for agent α_1 while Table 4 presents how these paths are evaluated following the criteria profit and time modeling the private goals of the agents.

.

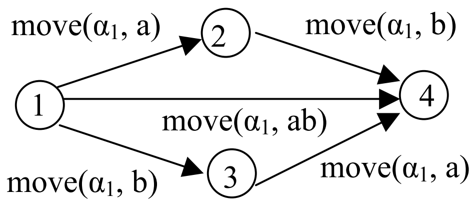


Figure 9: Efficient paths graph of agent α_1

Paths	Profit	Time
1-2-4	4	2
1-3-4	4	2
1-4	2	1

Table 4: Agent α_1 efficient plans' evaluation

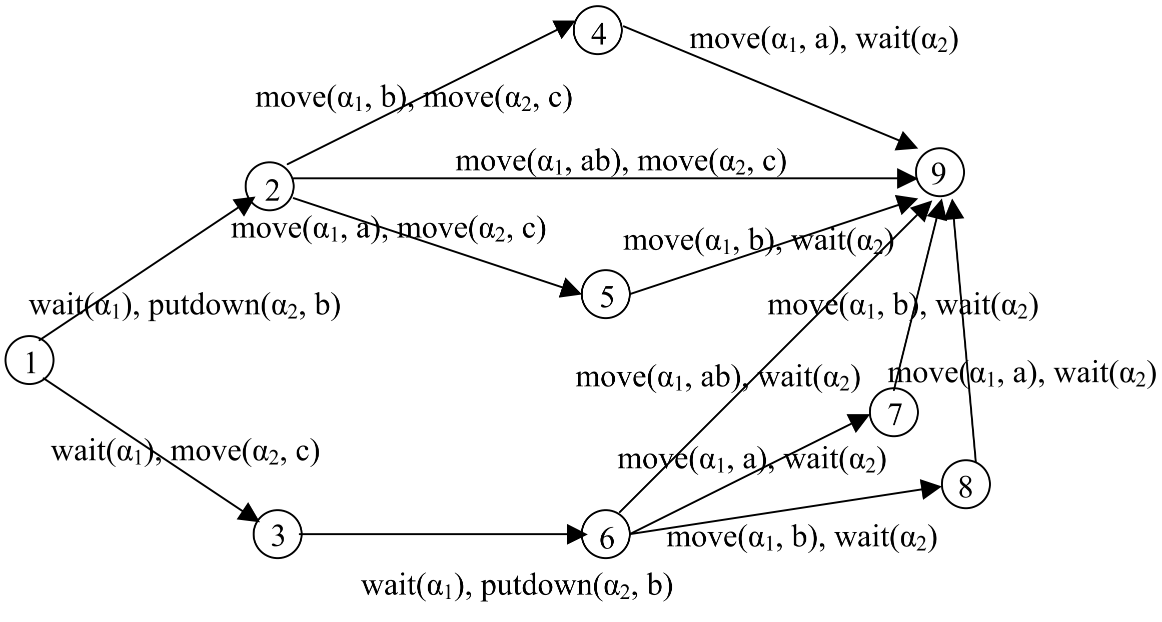


Figure 10: Negotiation graph for agents α_1, α_2

	Agent α_1		Agent α_2	
Paths	Profit	Time	Profit	Time
1-2-4-9	3	3	3	3
1-2-9	1	2	4	2
1-2-5-9	3	3	3	3
1-3-6-9	0	3	3	3
1-3-6-7-9	2	4	2	4
1-3-6-8-9	2	4	2	4

Table 5: Possible multi-agent plans evaluation for agents α_1, α_2

Figure 10 represents the graph of possible multi-agent plans (also called negotiation graph), which is the result of the merging procedure, while Table 5 presents how these paths are evaluated following the criteria profit and time modeling the private goals of the agents.

3.2.3 Conflicts Resolution and Negotiation

In our setting, each total ordered sequence of actions (appearing in the pairs labeling the arcs of the negotiation graph and forming each possible multi-agent plan) gives a different evaluation of the same plan for each agent. This is why each agent evaluates a multi-agent plan by using his own private goals as evaluation criteria. These private goals can be different, partially the same, or completely the same (as in our example), having in this case the same importance for each agent or not (in our example we can consider that for agent α_1 min-time is more important than max-profit, while for agent α_2 is the opposite). Therefore, even if all the possible multi-agent plans achieve the joint global goal, the payoff for each agent is different. Thus, there is a conflict between the agents of our example, since agents simultaneously want to optimize, as much as they can, their preferred private goals (or a trade off between all the private goals). This is a difference between our approach and the other approaches in the literature.

In fact, this conflict is due to two reasons. The first is that the private goals of an agent can be inconsistent with his work goal, in the sense that the best actions allowing the work goal optimization for the global good are not necessarily the ones, which optimize its motivations. In other words, the agent would not have done the same actions for its work goal if it was completely benevolent (devoted) to the global success (personal interest is sacrificed in front of the collective one's) and if it had no "back thoughts" on what it can win from its cooperation with the others. The second reason is that there may exist a conflict among the private goals of the agents. In any case, these reasons generate this conflict that concerns the decision about which of the possible multi-agent plans to choose for execution, taking into account that each agent would prefer the plan that contains the total ordered sequence of his actions that optimizes his preferred private goal.

In order to resolve this conflict we use negotiation. The base of the negotiation is the negotiation graph and the cycle of negotiation is initiated in order to decide which, among the possible multi-agent plans (i.e. plans free of conflicts where the actions to be taken by the agents in parallel and asynchronous way appear in a strict order), constitutes the best compromise for the involved agents. Negotiation is based on our work proposed in this field [Moraïtis & Tsoukiàs, 96; El Fallah-Seghrouchni, Moraïtis & Tsoukiàs, 99; El Fallah-Seghrouchni, Moraïtis & Tsoukiàs, 00; Moraïtis & Tsoukiàs, 02b], which is presented in the following section.

Let's now consider again our example. According to our planning model presented in section 2.3, in order to compute the best plans, agents must specify a relative importance between their private goals. Considering for example that for agent α_1 min-time is more important than max-profit, while for agent α_2 is the opposite, the reader can easily conclude that agents α_1, α_2 will have an agreement on the plan represented by the path 1-2-9 (Fig. 10). This is easily computed by applying our negotiation models presented in the next section. If now we consider the opposite situation (i.e. max-profit for α_1 and min-time for α_2 as more important), the result is less obvious but (by applying for example our utility functions based negotiation model) the candidate plans will be 1-2-4-9 or 1-2-5-9 (Fig. 10).

3.2.4 Future Work

Our future work in distributed planning is related to two aspects. The first one concerns the extension of the current model, by combining it with our dynamic planning model, in order to take into account the execution stage in dynamic environments. This will imply an enrichment of the model in order to allow a dynamic PNE cycle in case of unpredictable execution failures during the execution stage (e.g. an agent who cannot execute alone an action, despite his initial consideration and therefore he needs help). This also implies an update of the plan-merging algorithm we have proposed.

The second aspect concerns the proposal of efficient plan merging algorithms for distributed planning models in dynamic environments, based on HTN formalism (e.g. [Erol, Hendler & Nau, 94]). This work will be integrated in the context of a relatively new paradigm for planning in complex, dynamic environments, called distributed, continual planning (desJardins et al, 1999).

3.3 Automated Negotiation

As said above, one of the most important features of agents is their autonomy to carry out tasks, and make choices and decisions. Hence, it is necessary for agents to be able to develop their own strategy (i.e. no coordination mechanism can be imposed externally). Consequently, the diversity of strategies may raise conflicts that can be solved through negotiation among agents. Several definitions of negotiation have been proposed in the MAS literature. We adopt the one proposed in [Lomuscio, Wooldridge & Jennings, 00], that is "negotiation is defined as the process by which a group of agents communicate with one another to try and come to a mutually acceptable agreement on some matter". In fact, negotiation helps agents to modify their local plans and/or decisions in order to avoid negative (i.e. harmful) interactions and emphasize the situations where positive (i.e. helpful) interactions are possible.

Automated negotiation has long been studied in the MAS field. Different negotiation mechanisms have been proposed (e.g. [Jennings & al., 98; Sandholn & Lesser, 95; Sycara, 89; Zlotkin & Rosenschein, 96; Aknine, Pinson & Shakun, 02]). Most of these approaches are based on operational research techniques. For an interesting overview on negotiation in multi-agent environments, the reader may see [Jennings & al., 01; Kraus, 97; Műller, 96]. However, the multi-criteria dimension of the negotiation process is basically ignored in the different approaches presented in the literature; an exception to that is the work of Faratin, Sierra & Jennings (see for example [Faratin, Sierra & Jennings, 98; 00]) and Sycara [Sycara, 90] who combines case-based reasoning and optimization of multi-attribute utilities. Our work lies in this context and has been presented in [Moraïtis & Tsoukiàs, 96; El Fallah-Seghrouchni, Moraïtis & Tsoukiàs, 99; El Fallah-Seghrouchni, Moraïtis & Tsoukiàs, 00].

3.3.1 Negotiation Strategies

More precisely, in [Moraïtis & Tsoukiàs, 96; El Fallah-Seghrouchni, Moraïtis & Tsoukiàs, 99] we have presented a set of multi-criteria negotiation strategies where the negotiating agents are "sincerely cooperative", in the sense that they look for a compromise, while they may accept, if necessary, to have a loss in their private goals in order to reach the global goal. These negotiation procedures are used in the context of distributed planning and are applied on the negotiation graph representing the possible multi-agent plans and created by the merging of the individual efficient paths graphs of the involved agents (i.e. presented in the previous section). We remind that agents in this setting have two sort of goals (i.e. work and private) and that work goals are necessary for a global goal achievement. We also remind that the offer of an agent (in this setting) consists of the choice of a path of the negotiation graph (corresponding to a possible multi-agent plan) and we remark that each agent makes his choice by using his strictly individual preference model.

The negotiation procedures we have proposed may concern the settling of a new multi-criteria model using the union of the set of criteria of each negotiating agent. There exist two possibilities: the first one is to use a hierarchical model of preference aggregation (all criteria of an agent are aggregated to a single criterion and so on); the second one is to use a flat model considering all the criteria simultaneously. The choice depends on the nature of the criteria and the agent's preferences.

A first negotiation strategy corresponds to the definition of a compromise solution among the efficient paths of the negotiation graph. Different procedures can be used as establishing a global utility function (if the agents accept establish trade-offs among their criteria), go through direct pair wise comparisons (if the agents accept to simply compare their criteria) and so on. A second negotiation strategy, in case that the first one fails to find a compromise solution, is to re-discuss the model enhancing the criteria set. Under the new model, a negotiation of the first type can be set again. A third negotiation strategy, in case that the two previous ones fail, is to change again the model, by introducing a new common criterion, which could be more important than their previous criteria. A fourth negotiation strategy is to change the model introducing actions that were not previously considered. If such a situation occurs, the set of criteria has also to be redefined. The negotiation can go back to the first strategy and the whole process restarts.

3.3.2 An Aggregation-Disaggregation Approach

In [El Fallah-Seghrouchni, Moraïtis & Tsoukiàs, 00], we proposed an automated negotiation framework for a more general setting (always for agents using a multi-criteria decision model though), which is based on an aggregation-disaggregation procedure. This work is an extension of the ideas developed in [Moraïtis & Tsoukiàs, 96; El Fallah-Seghrouchni, Moraïtis & Tsoukiàs, 99]. Under the perspective proposed in [Moraïtis & Tsoukiàs, 96] and presented before in Sections 2.3 and 3.2, the agent's problem consists in solving a dynamic programming problem, that is to define the "best path" on a graph whose nodes are the states of the world, the arcs are the elementary actions, paths correspond to plans, and $\mathcal{H}_i$ and $\mathcal{G}_i$ represent the agent's preferences in order to define what is "best" (see Section 2.3).

Moving up an abstraction level, the decision model presented in section 2.3 may be extended to a community of agents *Ag* as follows:

Ag: $\langle \mathcal{T}, \Delta, \mathrm{H}, \mathrm{P}, \Gamma, \Re\ \mathcal{S} \rangle$ where:

- $\mathcal{T}$: a set of tasks to be accomplished by the community (different levels of accomplishment may be considered);
- Δ: a set of elementary actions available to the community of agents;

- H: a collection of $\mathcal{H}_j \subseteq \Delta\times\Delta$ binary preference relations on the set Δ of the type $\forall$ x, y $\in$ Δ: $\mathcal{H}_j$(x,y): the community, on dimension j, considers action x at least as good as action y;
- P: a set of plans (ordered sets of actions) the community may perform in order to accomplish the tasks belonging to $\mathcal{T}$;
- Γ: is a collection of $\mathcal{G}_l \subseteq \mathrm{P}\times\mathrm{P}$ binary preference relations on the set P of the form: $\forall\chi$, $\psi \in \mathrm{P}$, $\mathcal{G}_l(\chi,\psi)$ means that the community, on dimension l, considers plan χ at least as good as plan ψ;
- $\Re$: is an aggregation procedure enabling to establish a global relation H and Γ (if it is the case) and to connect the relations $\mathcal{H}_j$ to the relations $\mathcal{G}_l$;
- $\mathcal{S}$: is a set of states of the world representing the consequences of each elementary action the community may perform.

Under a conventional negotiation scheme the only object on which the negotiation may hold are the parameters defining $\Re$. In such a case it is necessary to consider:

$$\mathcal{T} = \cup_i \mathcal{T}_i \text{ and } \Delta = \cup_i \mathcal{A}_i$$

It is clear that such a perspective is very reductive with respect to the negotiation requirements of a multi-agent system. Moreover the existence of a multi-agent system level may enable actions not foreseen on a single agent level and modify the way by which plans are evaluated (i.e. each agent $\mathcal{G}_l$).

Under such a perspective we claim that the negotiation objects in a multi-agent system include:
- the establishment of $\Re$ and its parameters, considering $\mathcal{T}$, and Δ fixed;
- the establishment of Γ possibly modifying each agent $\mathcal{G}_i$;
- the establishment of P possibly modifying each agent $\mathcal{A}_i$, $\mathcal{T}_i$ and $\mathcal{H}_i$.

In our work presented in [El Fallah-Seghrouchni, Moraïtis & Tsoukiàs, 00] we have proposed a procedure concerning the first among the above negotiation objects. In fact, although it concerns the most commonly explored problem (at least in decision theory), it turns out that the extension of conventional negotiation models in the context of multi-agent systems is far than trivial.

In this setting, an agent can make an offer using his strictly individual preference model (called multiple criteria aggregation based decision step), while the same agent can receive the counter-offer of a counterpart. Such a model takes the form of a multiple-attribute additive value function. Using a multiple linear regression model (disaggregation step), agents are able to estimate the parameters of the preference model of their counterpart (that is the trade-offs and the shape of the value functions). Based on this estimation, agents are able to create an enhanced preference model by including the estimated preference model of their counterpart in their own model and compute a new offer on the basis of the enhanced model. The procedure loops until a consensus is reached; that is, all the negotiating agents make the same offer.

The key idea above is that during a negotiation process, each participant making an offer (that is, making a choice) tries to take into account the preferences of his/her counterpart. However, such preferences are initially unknown and are revealed gradually during the negotiation process through the counter-offers of the counterpart.

The negotiating agents, α_1 and α_2, use the same reference set of possible actions $\mathcal{A}$. There is a common family of criteria $\mathcal{G}$. Each of the two negotiators uses a preference model:

$$\sum p_i v_i = \sum u_i$$

The evaluation scheme $u_i = g_i(a_j)$ is common, and the different elements are the trades-off p_i providing so two different rankings R_1, R_2 of the alternatives belonging in set $\mathcal{A}$.

The linear multiple regression method used in order to estimate the utility functions is the one adopted in the UTA approach [Jacquet-Lagrèze & Siskos, 82]. We use the weighting sum of criteria values:

$$u(g) = \sum_{i=1}^{n} p_i g_i = \sum_{i=1}^{n} u_i(g_i)$$

Let us call P the strict preference relation and I the indifference relation. If $\underline{g} = g(a_i) = [g_1(a_i), g_2(a_i), ..., g_m(a_i)]$ is the multi-criteria evaluation of an action α_i, then the following properties generally hold for the utility function U:

$$U[g(\alpha)] > U[g(b)] \Leftrightarrow \alpha P b \qquad U[g(\alpha)] > U[g(b)] \Leftrightarrow \alpha > b$$

or

$$U[g(\alpha)] = U[g(b)] \Leftrightarrow \alpha I b \qquad U[g(\alpha)] = U[g(b)] \Leftrightarrow \alpha = b$$

It is possible then to use the relation $R=P \cup I$ as a weak ordering depicting the subjective preferences of an agent. In order to assess parameters p_i of the utility function, the UTA method uses special linear programming techniques. As inputs to the method we use the common evaluations $g_i(a_i)$ and the weak ordering of the actions.

We suppose that possible actions have the same value for both agents. The procedure of the negotiation is presented as following:

```
begin
Each agent evaluates the possible actions based on his individual preference model and computes an initial ranking of these actions expressing his preferences.
loop
    1. agent $\alpha_1$ makes an offer $x_1^*$ such that: $x_1^*=\max_{x\in\Delta} \sum_j p^1_j g^1_j(x)$
    2. agent $\alpha_2$ makes an offer $x_2^*$ such that: $x_2^*=\max_{x\in\Delta}\sum_j p^2_j g^2_j(x)$
    3. knowing agent's $\alpha_2$ counter-offer, agent $\alpha_1$ can establish that: $\forall x \in A$, $\sum_j p^2_j g^2_j(x_2^*)>\sum_j p^2_j g^2_j(x)$;
    4. on these grounds and using a multiple linear regression model agent $\alpha_1$ is able to make a first estimation of the parameters of the preference model of agent $\alpha_2$ (that is the trade-offs and the shape of the value functions);
    5. using such estimation, agent $\alpha_1$ creates an enhanced preference model including the estimated preference model of agent $\alpha_2$ in its own model (actually agent $\alpha_2$ estimated value function will become a criterion to add to agent's $\alpha_1$ preference model) and computes a new ranking;
    6. agent $\alpha_1$ goes back to the first step and choose a new offer on the basis of the enhanced model; the procedure loops until a consensus is reached (agent $\alpha_2$ makes the same offer as agent $\alpha_1$);
  end loop
  end
```

3.3.3 A Utility-Based Negotiation Model

In [Moraïtis & Tsoukiàs, 02b] we propose a new negotiation model inspired by the work proposed in [Faratin, Sierra & Jennings, 98]. The negotiation process ensures an easy way to find an agreement. We have applied it in a distributed planning context, where the negotiating agents are "sincerely cooperative", in the sense that they look for a compromise. However, the above model can be also used in a more general setting.

Let's consider the following setting:
-*m* economically rational agents (α_i, $i=1,\dots m$) each of which using *n* criteria (always the same) in order to evaluate alternative options;
-to each criterion *j* ($j=1,\dots,n$) we associate an utility function g_j: $A\mapsto[0, 1]$, A being the set of alternative options; we assume $A=[0, 1]^n$;

-each agent considers a vector of trade-offs $<w^i_1,\ldots,w^i_n>$, such that is possible to obtain for each agent his linear additive comprehensive utility function $V_i(x)=\sum_j w^i_j g^i_j(x)$; without loss of generality we assume $\forall i \ \sum_j w^i_j =1$;
-without loss of generality we consider that an agent l establishes his own aspiration level $\theta(\alpha_l)=V_l(\alpha)$ where α is an alternative for which $g_k(\alpha)=1$, $g_{j\neq k}(\alpha)=0$, k being the criterion for which $w^l_k=max_j(w^l_j)$; in other terms $\theta(\alpha_l)= max_j(w^l_j)$.

The negotiation process can be as follows: Any among the agents makes an offer. The agent being economically rational the offer will be any point satisfying the equation $\sum_j w^i_j g^i_j(x)=max_j(w^l_j)$. Then the next agent will make a new offer which, besides satisfying its own comprehensive utility function, it will be as near as possible (in terms of Euclidean distance) to the previous offer and so on. It is obvious that after $n \times m$ offers each agent perfectly knows the hyperplanes containing all the possible offers of all the other agents. Then we just have to compute the intersection of such hyperplanes (and any agent is able to do it) in order to find the set of offers which satisfy contemporaneously the aspiration level of all agents. Rationally, any such offer can be an agreement ending the negotiation process.

In order to better understand our model let's consider the following example. Consider the situation where two agents 1,2 have the following comprehensive utility functions:

$$V_1(x)=0.25x_1+0.75x_2$$

$$V_2(x)=0.6x_1+0.4x_2$$

Clearly $\theta(1)=0.75$ and $\theta(2)=0.6$ and the hyperplanes (in this case the lines) containing the possible offers of the two agents will be:

$$\text{agent 1: } 0.25x_1+0.75x_2=0.75$$

$$\text{agent 2: } 0.6x_1+0.4x_2=0.6$$

Agent 1 will make the first offer and we assume without loss of generality that he will propose the extreme point most favorable to him: <0, 1>. Agent 2 can compute which is the point of his hyperplane nearest to this offer: <0.33, 1>. Agent 1 can compute on its turn the nearest offer satisfying his hyperplane, which is: <0.3, 0.9>. Agent 2 is now able to infer exactly the utility function of agent 1 (the two points exactly describe it) and therefore is able to compute the intersection with his own utility function, therefore suggesting: <0.42, 0.85>. Agent 1 does the same calculation and will reach the same point, therefore will agree to this offer. The agreement is reached (see Fig. 11).

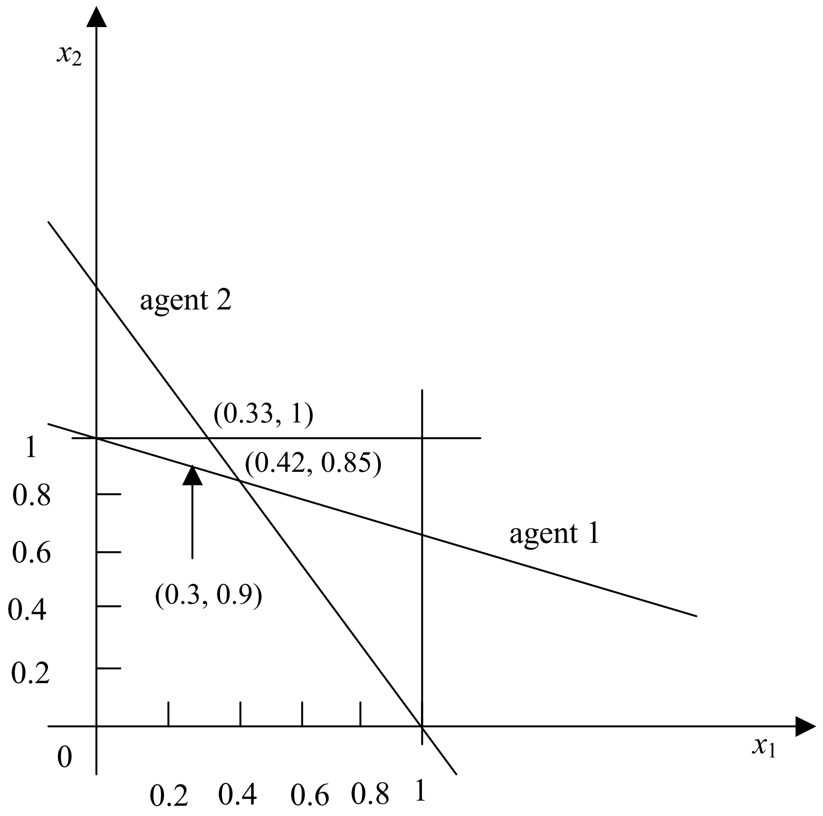


Figure 11: Graphical representation of the utility functions

3.3.4 An Argumentation-Based Negotiation Model

In [Kakas & Moraïtis, 03] we propose an argumentation-based approach for negotiation (an important issue for automated negotiation [Jennings et al., 01]), based on our argumentation framework. Negotiation is built on the exploitation of the internal argumentative policy theory of the agents, their current goals and other supporting information about the external environment that each agent has accumulated from the other agent. This extra supporting information is build gradually during the negotiation and it allows an incremental deliberation of the agents as they acquire more information. Here we must make clear, that for the moment, we are not interested in the conversation protocol supporting the negotiation.

The negotiation protocol applied by the agents could be described as follows:

- Each agent insists in making proposals and counter-proposals as long as his deliberation with his theory and the accumulated supporting information (agreed by the two agents so far) produces new supporting evidence (see Definition 4, Section 2.2.1) for his goal, suitable to convince the other agent. A goal contains several offers and each of them is supported by its supporting information (e.g. a goal to buy at a low price may contain several prices each of them being supported by the appropriate information).

- The first of the two negotiating agents, who is unable to produce a new such supporting evidence, abandons his proposal (i.e. his goal) and searches for supporting information, if any, under which he can accept the counter-proposal (i.e. the goal) of the other agent (e.g. a seller agent unable to find another way to support his high price considers selling at a cheap price, provided that the buyer has a regular account and pays cash).

- In such a case, if the receiver agent can endorse the proposed supporting information the interaction ends with the agreement on this goal and the supporting information accumulated so far.

- Otherwise, if the receiver refuses some of the proposed supporting information the sender takes this into account and tries again to find another way to support the goal of the other agent. If this is not possible then the interaction ends in failure to agree on a common goal.

Besides the argumentative functions deliberate and accept (see Definition 6, Section 2.2.1), in our model, we use three more auxiliary functions, which are external to the argumentative reasoning of an agent and relate to other functions of the agent in the negotiation procedure.

- The function propose(Goal, e_j, s_i) is used by a sender agent to determine what information to send to the other agent: Goal is a goal proposed, e_j is the evaluation by the sender of the supporting information s_j sent to him in the previous step by the other agent and s_i is a new supporting evidence produced by the deliberation function of the sender.

- The function evaluate(Ag, s_i) produces e_i where each (abducible) literal in the supporting information s_i may remain as it is or negated according to some external process of evaluation of this by an agent Ag. This function used by an agent within the negotiation process in order to decide if he can accept a proposed supporting information s_i, can vary in complexity from a simple check in the agent's database on the other hand to a new (subsidiary) argumentative deliberation on s_i according to a related argumentative policy theory that the agent may have.

- The function update(S,e) updates, through an external mechanism, the accumulated supporting information S with the new information e consisting of the agent's evaluation of the supporting evidence sent by the other agent and the evaluation of his own supporting information by the other agent.

The negotiation protocol described above can be represented, more formally, by the following algorithm:

```
begin
1. agent X receives a proposal $O_i$ from an agent Y,
2. if the proposal is of the form ($G^Y$, $e^{Y\to X}$, $s_n$Y) then
3.        $e_n^{X\to Y} \leftarrow$evaluate(X, $s_n^Y$);
4.        S$\leftarrow$update(S, $e_{n-1}^{Y\to X} \cup e_n^{X\to Y}$)
5.        if X accept(X, $G^Y$, S) then End (agreement, $G^Y$)
          else
6.                n $\leftarrow$n+1; deliberate(X, $G^X$, S; $s_n^X$)
7.                if $s_n^X$ exists then
8.                        propose($G^X$, $e_{n-1}^{X\to Y}$, $s_n^X$) to Y
                  else
9.                        S$\leftarrow$update(S, $e_n^{Y\to X}$)
10.                       n $\leftarrow$n+1; deliberate(X, $G^Y$, S; $s_n^X$ )
11.                       if $s_n^X$ exists then
12.                             propose($G^Y$, $\emptyset$, $s_n^X$) to Y
13.                       else end(Failure)
14.                       endif
15.               endif
16.       endif
17. endif
18. if the proposal is of the form ($G^Y$, $e_n^{Y\to X}$,$\emptyset$) then goto 9; endif
19. if the proposal is of the form ($G^X$, $\emptyset$, $s_n^Y$ ) then
20.       $e_n^{X\to Y} \leftarrow$evaluate(X, $s_n^Y$)
21.       if $e_n^{X\to Y}$ = $s_n^Y$ then End (agreement, $G^X$)
22.       else propose($G^X$, $e_n^{X\to Y}$, $\emptyset$)
23.       endif
24.endif
end
```

where e.g., $e_n^{X\to Y}$ stands for agent X's evaluation of the supporting evidence sent by agent Y.

3.3.5 Future Work

My future work directions on negotiation concern different aspects. One aspect is related to the multi-criteria negotiation per se. The objective here is to extent the negotiation objects by considering i) the establishment of $\mathcal{G}$ (collection of binary preference relations for the community of agents), possibly modifying each agent's $\mathcal{G}_i$

(binary preferences relations on the set of his plans), and ii) the establishment of P (the set of plans the community may perform), possibly modifying each agent $\mathcal{A}_i$ (set of elementary actions), $\mathcal{T}_i$ (set of tasks) and $\mathcal{H}_i$ (set of binary preferences on his actions). This can be viewed as negotiating the negotiation model itself.

Another aspect concerns a more precise consideration of the aggregation-disaggregation approach, where work must be done in order to efficiently define how an agent can automatically decide which is the weight he would assign to the estimated value function of the other agent (which he uses as an additional criterion). A related issue concerns the decision policy the agent uses in order to update the weights of the other criteria.

Concerning the proposed argumentation-based model, the aim is to integrate in it, our work on the influence of personality in an agent's decision making policies related to his capabilities (see Section 2.2.3), in order to study how this dimension can influence the behavior of an agent during a negotiation process (i.e. negotiation between a selfish and an altruist agent, or between two selfish agents, etc.). We will, finally, also explore the combination with the multi-criteria aspects already proposed.

4 Miscellaneous

In this section, I present some ideas concerning a new work on automatic decision-making and, more particularly, on multi-criteria evaluation of abstract operators in hierarchical planning. Moreover, I discuss issues concerning two other problems that attracted my interest. These concern the task allocation problem and the inherent difficulty in making decisions under multi-criteria (e.g. in the planning and scheduling context).

4.1 Multiple Criteria Evaluation of Actions in Hierarchical Decomposition

This is a common project with Alexis Tsoukiàs. Hierarchical decomposition in planning appears useful when some operators can be decomposed in more than one way [Russel & Norving, 95]. The problem we address in this work is exactly this one and the originality is situated into the introduction of a multi-criteria evaluation of actions in the hierarchical decomposition algorithms. So, we consider that when an agent has to undertake a task he has to decide the action (or plan) to follow for its accomplishment. Therefore the agent has to compare the different actions (plans) and make a choice. Then he may execute the decided action (plan) and possibly may reconsider the evaluations under which that precise decision has been taken. In any case the agent, in order to be able to decide, has to elaborate preferences. We consider two types of preferences. One, which is independent from the precise decision process for which the planning is undertaken and we call them generic preferences. The other which depends from the decision process. In this case we distinguish again two types of preferences. One where only the labels of the operator(s) of actions (plans) are considered and we call them contextual preferences and the other where besides the label of the operator(s) also the instances of their variables are considered and we call them structural preferences.

It is obvious that hierarchical decomposition is just a specific application case. This work could be applied in a more general setting of automated decision making for autonomous agents, where agents have to choose among several alternatives, based on various criteria.

4.2 Multi-Criteria based Task Allocation

In the multi-agent systems domain, cooperation among agents is a fundamental process, in that it aids them to resolve a global problem taking into account that agents have only a partial view of the overall problem. In this work [Balbo, Moraïtis & Pinson, 96a; 96b], we have presented a multi-criteria based cooperation approach. According to this approach, tasks and agents are characterized by a (separate) set of criteria. Then, our approach exploits agents' preferences. More precisely, it tries (through preferences' aggregation) to find a satisfactory combination for the task allocation problem, according to the current situation of each time instance.

4.3 Why is Difficult to Make Decisions under Multiple Criteria

This work [Della Croce, Tsoukiàs & Moraïtis, 02] makes a survey of the principal difficulties the multi-criteria decision making introduces with a particular emphasis on scheduling and planning problems. Two types of difficulties are considered. The first is of conceptual nature and has to do with the difficulty of defining the concept of optimality in presence of multiple criteria and the impossibility to define universal preference aggregation procedures. The second difficulty is of more technical nature and concerns the increasing computational complexity of multiple criteria decision making problems.

PART 2

APPLICATION AND EXPERIMENTAL WORK

Commerce Electronique

Résumé

Dans cette section je présente mon travail en commerce électronique. Ce travail concerne la proposition d'un système de marché électronique artificiel qui peut être utilisé dans deux situations différentes. La première concerne l'achat et la vente des produits, la deuxième l'achat et la vente des services.

Dans la première situation, en utilisant ce système, des acteurs (c.a.d. des clients et des marchands) peuvent déléguer une varieté de tâches à des agents intelligents personnels, qui agissent comme leurs employés artificiels. Plus particulièrement, l'originalité de ce travail est fondée sur la personnalisation des agents qui "vivent" en permanence dans le marché (contrairement aux autres travaux), en représentant les intérêts de leurs acteurs. Elle réside aussi dans les points suivants: a) les agents peuvent prendre l'initiative de demander la permission à leurs utilisateurs d' initier une transaction b) le système possède un outil hautement interactif de prise de décision multicritère, qui est capable de prendre en compte des informations incomplètes durant une transaction d'achat et de réaliser une synthèse progressive et une évaluation comparative des propositions existantes.

Dans la seconde situation, nous validons l'applicabilité de notre protocole de conversation présenté auparavant entre agents. Plus précisément dans cette situation, des utilisateurs ayant besoin d'un service (p.ex., reserver une table dans un restaurant et/ou des billets de cinéma), interagissent avec leurs agents artificiels afin de décrire le service qu'ils souhaitent. Ensuite l'agent artificiel, en utilisant ses acquointances dans le marché (c.a.d les fournisseurs de services), cherche d'abord le fournisseur le plus approprié afin d'établir un dialogue avec lui avant de lui soumettre une demande avec les caractèristiques du service dont il a besoin. Si le fournisseur est capable de satisfaire la demande dans sa totalité il informe le demandeur de la solution trouvée et le dialogue s'arrête. Autrement (si seuleument une partie de la demande est satisfaite), un dialogue secondaire est initié par le fournisseur afin de trouver des solutions alternatives qui pourraient satisfaire le demandeur du service. Si les deux agents arrivent à un accord le dialogue est terminé et la transaction est validée, autrement la transaction échoue et le dialogue s'arrête là.

5 E-Commerce

My work on e-commerce concerns the design and implementation of an agent-mediated artificial market system with advanced features. In this framework, agents collaborate in real-time mode. Using the system, actors (i.e., customers and providers of services and products) delegate various tasks to their personal intelligent agents, which act as artificial employees. In the proposed system, we have considered the case of buying and selling of products [Karacapilidis & Moraïtis, 01a; Karacapilidis & Moraïtis, 01b; Karacapilidis & Moraïtis, 00a; Karacapilidis & Moraïtis, 00b] and the case of buying and selling of services [Karacapilidis & Moraïtis, 04]. Our approach highly builds on the feature of pro-activeness and semi-autonomy of all software agents involved.

5.1 Buying and Selling of Products

Contrary to the majority of the already implemented systems, our system addresses efficiently many important issues. In the case of buying and selling of products, agents (due to the properties of pro-activeness and semi-autonomy) can take the initiative to contact their actors in order to start a transaction that seems "interesting" to them (e.g., when a new product, which matches one's profile, appears in the market), or trigger an actor's action (e.g., they can inform their merchant that a specific offer is of no interest in the market for the last month). We argue that semi-autonomy of agents assures the right level of control for the actions they could take; a fully autonomous agent could cause problems in such environments.

Second, our framework is based on a long (or even permanent) existence of agents in the e-market. In other words, agents do not "live" only during a specific transaction but much longer, upon the subscription paid by their owners at the time they were launched (i.e., an actor may "hire" an agent for a month, a year, etc.). This is highly associated with the personalization of the agents involved, through the maintenance of each actor's profile. For instance, a customer's agent can be assigned with a number of general interests (e.g., classical music, cruises) and preferences (e.g., one may dislike the black color on any product) of its actor, which can be enriched with more detailed ones each time the customer initiates a transaction, takes a decision to buy a certain product from a certain supplier, etc.

More precisely the development of the software agents proposed in our system is based on a generic and reusable architecture. My modular conception for an agent's architecture presented in Section 2.1 is applied here. The purchaser and seller agents are composed of three modules (namely, the communication, coordination and decision making modules), which run concurrently and intercommunicate by exchanging internal (i.e., intra-agent) messages. The first two modules are identical;

the third one is different, reflecting the different roles that purchaser and seller agents play in the system.

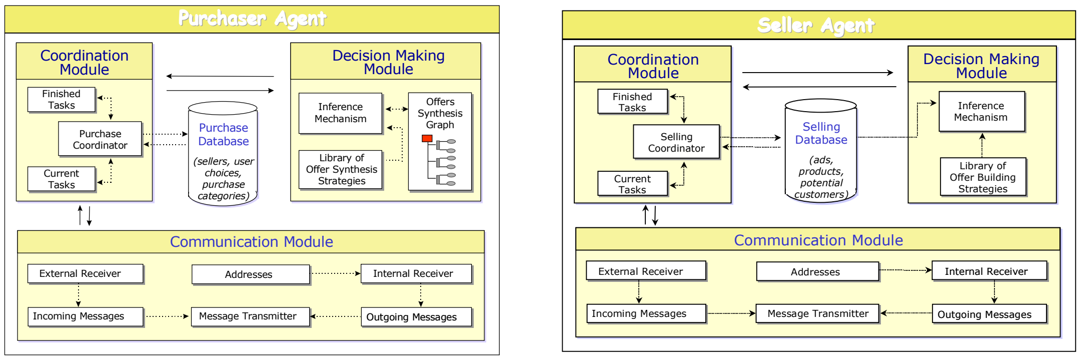


Figure 12: Architecture of a purchaser agent and a seller agent

The communication module of a purchaser agent (or seller agent) is responsible for the agent's interaction with its environment, that is the seller agents (purchaser agents) and the human user it assists. It sends and receives messages, while internally interacts with the coordination module.

The coordination module handles the parts of the cooperation protocol that concern any type of interaction between (i) the purchaser and the seller agents, and (ii) the purchaser agent (or the seller agent) and the customer it assists.

The decision-making module of a purchaser is composed of three components, namely an inference mechanism, a library of offer synthesis strategies, and the offers synthesis graph. It actually deploys the agent's reasoning mechanism that: (i) implements the behavior of the agent by using appropriate rules; for instance, the agent acts proactively upon the reception of some messages, sent by seller agents, and (ii) performs a synthesis and a comparative evaluation of the offers proposed by the seller agents; this process ultimately aims at finding the best offer (to be then recommended to the customer), according to the customer's choices and the information at hand.

Finally, the decision-making module consists of an inference mechanism and a library of offer building strategies. As in a purchaser agent, the inference mechanism of a seller also implements its proactive behavior. Furthermore, it uses the appropriate strategies to build offers for a requested or promoted product. Each such strategy reflects the selling policy to be followed by the seller agent, and may depend on the specific customer, product to be sold, merchant status, and so on (for instance, a different policy may be adopted when selling a new than a second-hand car).

As already said, E-market transactions in our system are initiated either by an actor or an agent. In the first case (see Fig. 13), a customer looking for a certain good or service contacts his/her purchaser agent and initiates a purchase transaction (Fig. 13, purchInitMsg message); in turn, the purchaser agent requests (from all or some seller agents) offers that may fulfill its actor's interests (Fig.13, offerReqMsg messages). Whenever a match between a purchaser and a seller agent is established, the latter gets information about the customer's buying criteria, preferences that may hold among them, as well as constraints explicitly imposed. By getting such a request, and presuming that the appropriate information exists in its selling database, a seller agent can directly build and propose an offer that is as close as possible to the purchase request (offerPropMsg message, sent by S1: seller agent). Otherwise (i.e., not enough information in the database), it has first to contact its merchant for an update of the related specifications (merSpecUpdReqMsg and merSpecUpdAnsMsg messages, exchanged between Sn: seller agent and its associated merchant, before the offerPropMsg message, sent by Sn: seller).

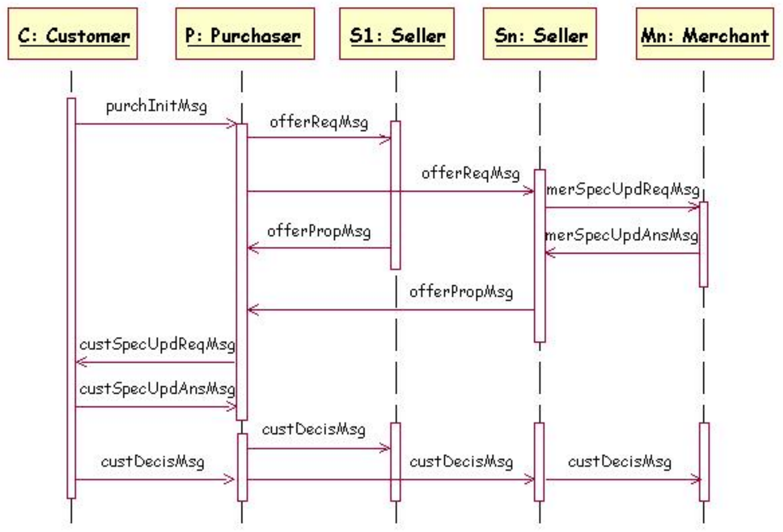
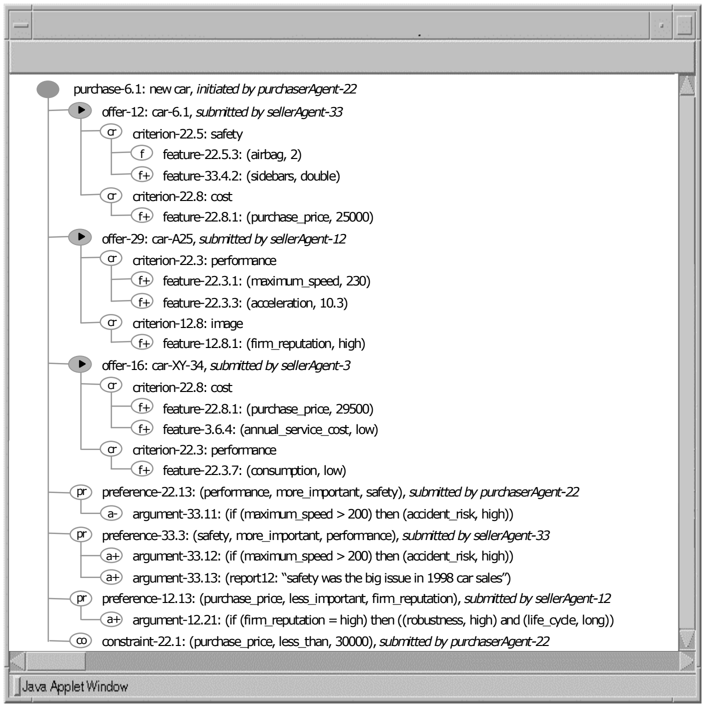


Figure 13: Agent-human interaction diagram

So, our system enables the e-market's seller agents to refine (some of) a customer's purchase criteria during a transaction, argue in favor or against them, or even bring up new information to persuade him/her to accept their offers.

Finally, our approach is able to handle incomplete, inconsistent and conflicting information during a purchase transaction, and perform a progressive synthesis and comparative evaluation (across a set of attributes) of the existing proposals. This is performed through the use of a highly interactive tool, based on multi-criteria decision theory, which enables customers easily examine alternative scenarios (by selecting which of the proposals' attributes to be taken into account) and recommends the best solution according to the information at hand.

More precisely, as soon as a purchaser agent gets a new offer proposal, it integrates it with the ones already arrived and constructs an offers synthesis graph, which is presented to the customer through the web interface shown in Figure 14. Preferences and constraints are kept together at the bottom part of the graph, since these refer to the overall purchase transaction. Each graph entry has an activation label indicating its current status (it can be active or inactive). By default, all entries are initially active. Viewing the graph through a standard web browser, the customer is able to inactivate any of its nodes (by using the mouse and clicking on them), the rationale being that their corresponding data types do not suit to his/her interests.



```
purchase-6.1: new car, initiated by purchaserAgent-22
    ▶ offer-12: car-6.1, submitted by sellerAgent-33
        cr criterion-22.5: safety
            f  feature-22.5.3: (airbag, 2)
            f+ feature-33.4.2: (sidebars, double)
        cr criterion-22.8: cost
            f+ feature-22.8.1: (purchase_price, 25000)
    ▶ offer-29: car-A25, submitted by sellerAgent-12
        cr criterion-22.3: performance
            f+ feature-22.3.1: (maximum_speed, 230)
            f+ feature-22.3.3: (acceleration, 10.3)
        cr criterion-12.8: image
            f+ feature-12.8.1: (firm_reputation, high)
    ▶ offer-16: car-XY-34, submitted by sellerAgent-3
        cr criterion-22.8: cost
            f+ feature-22.8.1: (purchase_price, 29500)
            f+ feature-3.6.4: (annual_service_cost, low)
        cr criterion-22.3: performance
            f+ feature-22.3.7: (consumption, low)
    pr preference-22.13: (performance, more_important, safety), submitted by purchaserAgent-22
        a- argument-33.11: (if (maximum_speed > 200) then (accident_risk, high))
    pr preference-33.3: (safety, more_important, performance), submitted by sellerAgent-33
        a+ argument-33.12: (if (maximum_speed > 200) then (accident_risk, high))
        a+ argument-33.13: (report12: "safety was the big issue in 1998 car sales")
    pr preference-12.13: (purchase_price, less_important, firm_reputation), submitted by sellerAgent-12
        a+ argument-12.21: (if (firm_reputation = high) then ((robustness, high) and (life_cycle, long))
    co constraint-22.1: (purchase_price, less_than, 30000), submitted by purchaserAgent-22
```

Figure 14: The offers synthesis graph.

Each preference has a consistency label, which can be consistent or inconsistent. Each time a preference is inserted in the offer synthesis graph, a mechanism checks if the constituent features or criteria of it exist in another (already inserted) preference. If yes, the new preference is considered either redundant, if it also has the same importance relation, or conflicting, otherwise. A redundant preference is ignored (not inserted in the graph), while a conflicting one is put next to the previously inserted preference, the rationale being to gather together conflicting preferences and stimulate the user to contemplate on them (that is, to select which one to inactivate), until only one becomes active.

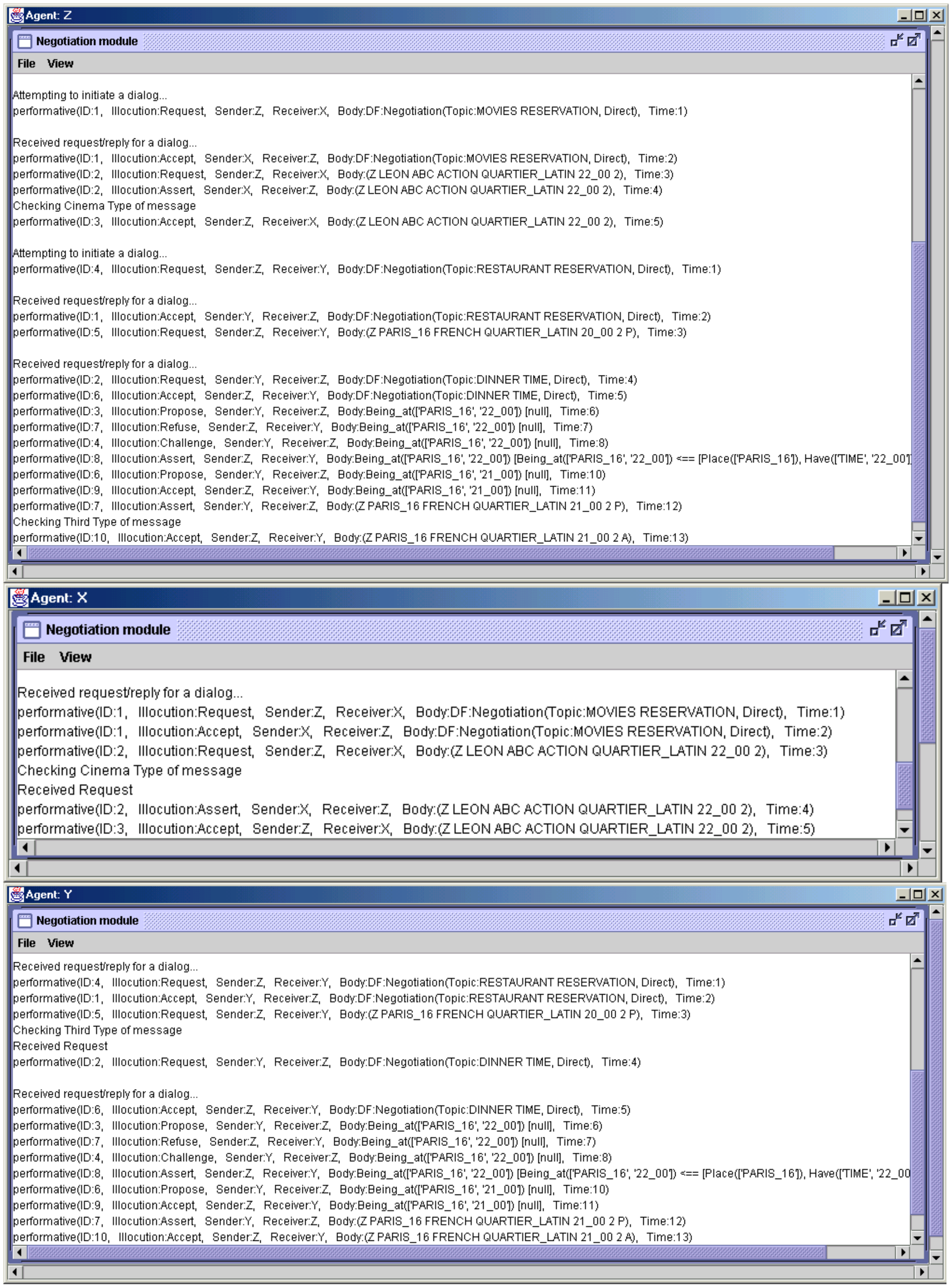
Active and consistent preferences participate in the weighting scheme. A detailed description of the algorithm used to assign weights to an offer's features appears in [Karacapilidis & Papadias, 98]. The basic idea is that the weight of a feature (or a criterion) is increased every time it is more important than another one (and decreased when is less important), the final aim being to extract a total order of offers. Since only partial information may be given, the choice of the initial maximum and minimum weights may affect the purchaser agent's recommendation.

5.2 Buying and Selling of Services

The case of buying and selling of services validates the applicability of the conversation protocol we have presented (see Section 3.1). In our system, a (human) user wanting to find a service (for instance, to reserve a table at a restaurant and/or some movie tickets) interacts with his artificial agent in order to describe the service he/she is looking for [Karacapilidis & Moraïtis, 04]. Using his "acquaintances" in the e-market (i.e., service providers), the artificial agent looks for the most appropriate one in order to first, establish a dialogue with him and then, submit a query with the characteristics of the service required. If the provider is able to fully satisfy the query submitted (i.e., all criteria and features can be satisfied), he informs the service demander about the solution found and the dialogue is over. Otherwise, that is only a part of the service requested can be satisfied, a subsidiary dialogue may start between the demander and provider agents aiming at finding alternatives that satisfy the request (such alternatives reside in the knowledge base of the providers). If they arrive to an agreement, the dialogue is over; otherwise, the transaction fails and the dialogue is closed. As in the case of buying/selling goods, artificial agents can also interact with their (human) users in order to get further information about the alternatives suggested each time. In this work, we present transactions carried out between artificial agents only; more specifically, we describe the way they autonomously convey dialogues about alternatives and we associate it with the representation of their profiles and behavior.

In the following, we present an example of buying and selling of services. Let agent Z acting towards making some reservations for his owner, who is actually a user of our system that looks for some services. His goal is to find a solution that combines dining at a good restaurant and going to the movies. To achieve this goal, Z contacts agents X and Y, which act as representatives (information providers) of the city's cinemas and restaurants, respectively.

Figure 15 illustrates the inter-agent dialogues performed by the agents Z, Y and X concerning a real application about the organization of a soirée. As it can be noticed, agent Z has first a successful dialogue with X about the movie reservation and then a dialogue with Y about the restaurant reservation. Because of the movie's reservation, the restaurant reservation necessitates nested dialogues. In fact, a dialogue about dinner-time is necessary, in order to solve the conflict with the already fixed movie's time.



```
Attempting to initiate a dialog...
performative(ID:1,  Illocution:Request,  Sender:Z,  Receiver:X,  Body:DF:Negotiation(Topic:MOVIES RESERVATION, Direct),  Time:1)

Received request/reply for a dialog...
performative(ID:1,  Illocution:Accept,  Sender:X,  Receiver:Z,  Body:DF:Negotiation(Topic:MOVIES RESERVATION, Direct),  Time:2)
performative(ID:2,  Illocution:Request,  Sender:Z,  Receiver:X,  Body:(Z LEON ABC ACTION QUARTIER_LATIN 22_00 2),  Time:3)
performative(ID:2,  Illocution:Assert,  Sender:X,  Receiver:Z,  Body:(Z LEON ABC ACTION QUARTIER_LATIN 22_00 2),  Time:4)
Checking Cinema Type of message
performative(ID:3,  Illocution:Accept,  Sender:Z,  Receiver:X,  Body:(Z LEON ABC ACTION QUARTIER_LATIN 22_00 2),  Time:5)

Attempting to initiate a dialog...
performative(ID:4,  Illocution:Request,  Sender:Z,  Receiver:Y,  Body:DF:Negotiation(Topic:RESTAURANT RESERVATION, Direct),  Time:1)

Received request/reply for a dialog...
performative(ID:1,  Illocution:Accept,  Sender:Y,  Receiver:Z,  Body:DF:Negotiation(Topic:RESTAURANT RESERVATION, Direct),  Time:2)
performative(ID:5,  Illocution:Request,  Sender:Z,  Receiver:Y,  Body:(Z PARIS_16 FRENCH QUARTIER_LATIN 20_00 2 P),  Time:3)

Received request/reply for a dialog...
performative(ID:2,  Illocution:Request,  Sender:Y,  Receiver:Z,  Body:DF:Negotiation(Topic:DINNER TIME, Direct),  Time:4)
performative(ID:6,  Illocution:Accept,  Sender:Z,  Receiver:Y,  Body:DF:Negotiation(Topic:DINNER TIME, Direct),  Time:5)
performative(ID:3,  Illocution:Propose,  Sender:Y,  Receiver:Z,  Body:Being_at(['PARIS_16', '22_00']) [null],  Time:6)
performative(ID:7,  Illocution:Refuse,  Sender:Z,  Receiver:Y,  Body:Being_at(['PARIS_16', '22_00']) [null],  Time:7)
performative(ID:4,  Illocution:Challenge,  Sender:Y,  Receiver:Z,  Body:Being_at(['PARIS_16', '22_00']) [null],  Time:8)
performative(ID:8,  Illocution:Assert,  Sender:Z,  Receiver:Y,  Body:Being_at(['PARIS_16', '22_00']) [Being_at(['PARIS_16', '22_00']) <== [Place(['PARIS_16']), Have(['TIME', '22_00']
performative(ID:6,  Illocution:Propose,  Sender:Y,  Receiver:Z,  Body:Being_at(['PARIS_16', '21_00']) [null],  Time:10)
performative(ID:9,  Illocution:Accept,  Sender:Z,  Receiver:Y,  Body:Being_at(['PARIS_16', '21_00']) [null],  Time:11)
performative(ID:7,  Illocution:Assert,  Sender:Y,  Receiver:Z,  Body:(Z PARIS_16 FRENCH QUARTIER_LATIN 21_00 2 P),  Time:12)
Checking Third Type of message
performative(ID:10,  Illocution:Accept,  Sender:Z,  Receiver:Y,  Body:(Z PARIS_16 FRENCH QUARTIER_LATIN 21_00 2 A),  Time:13)
```

```
Received request/reply for a dialog...
performative(ID:1,  Illocution:Request,  Sender:Z,  Receiver:X,  Body:DF:Negotiation(Topic:MOVIES RESERVATION, Direct),  Time:1)
performative(ID:1,  Illocution:Accept,  Sender:X,  Receiver:Z,  Body:DF:Negotiation(Topic:MOVIES RESERVATION, Direct),  Time:2)
performative(ID:2,  Illocution:Request,  Sender:Z,  Receiver:X,  Body:(Z LEON ABC ACTION QUARTIER_LATIN 22_00 2),  Time:3)
Checking Cinema Type of message
Received Request
performative(ID:2,  Illocution:Assert,  Sender:X,  Receiver:Z,  Body:(Z LEON ABC ACTION QUARTIER_LATIN 22_00 2),  Time:4)
performative(ID:3,  Illocution:Accept,  Sender:Z,  Receiver:X,  Body:(Z LEON ABC ACTION QUARTIER_LATIN 22_00 2),  Time:5)
```

```
Received request/reply for a dialog...
performative(ID:4,  Illocution:Request,  Sender:Z,  Receiver:Y,  Body:DF:Negotiation(Topic:RESTAURANT RESERVATION, Direct),  Time:1)
performative(ID:1,  Illocution:Accept,  Sender:Y,  Receiver:Z,  Body:DF:Negotiation(Topic:RESTAURANT RESERVATION, Direct),  Time:2)
performative(ID:5,  Illocution:Request,  Sender:Z,  Receiver:Y,  Body:(Z PARIS_16 FRENCH QUARTIER_LATIN 20_00 2 P),  Time:3)
Checking Third Type of message
Received Request
performative(ID:2,  Illocution:Request,  Sender:Y,  Receiver:Z,  Body:DF:Negotiation(Topic:DINNER TIME, Direct),  Time:4)

Received request/reply for a dialog...
performative(ID:6,  Illocution:Accept,  Sender:Z,  Receiver:Y,  Body:DF:Negotiation(Topic:DINNER TIME, Direct),  Time:5)
performative(ID:3,  Illocution:Propose,  Sender:Y,  Receiver:Z,  Body:Being_at(['PARIS_16', '22_00']) [null],  Time:6)
performative(ID:7,  Illocution:Refuse,  Sender:Z,  Receiver:Y,  Body:Being_at(['PARIS_16', '22_00']) [null],  Time:7)
performative(ID:4,  Illocution:Challenge,  Sender:Y,  Receiver:Z,  Body:Being_at(['PARIS_16', '22_00']) [null],  Time:8)
performative(ID:8,  Illocution:Assert,  Sender:Z,  Receiver:Y,  Body:Being_at(['PARIS_16', '22_00']) [Being_at(['PARIS_16', '22_00']) <== [Place(['PARIS_16']), Have(['TIME', '22_00
performative(ID:6,  Illocution:Propose,  Sender:Y,  Receiver:Z,  Body:Being_at(['PARIS_16', '21_00']) [null],  Time:10)
performative(ID:9,  Illocution:Accept,  Sender:Z,  Receiver:Y,  Body:Being_at(['PARIS_16', '21_00']) [null],  Time:11)
performative(ID:7,  Illocution:Assert,  Sender:Y,  Receiver:Z,  Body:(Z PARIS_16 FRENCH QUARTIER_LATIN 21_00 2 P),  Time:12)
performative(ID:10,  Illocution:Accept,  Sender:Z,  Receiver:Y,  Body:(Z PARIS_16 FRENCH QUARTIER_LATIN 21_00 2 A),  Time:13)
```

Figure 15: Inter-agent dialogues for a soirée organization

Marketing

Résumé

Dans cette section je présente mon travail en marketing. Ce travail concerne la proposition d'un système multi-agent pour implémenter une méthodologie de selection de stratégies pour la meilleure pénétration de nouveaux produits dans un marché. Le système que nous proposons répartit les agents en deux niveaux: un niveau fonctionnel et un niveau structurel. Au niveau fonctionnel, nous avons adopté une séparation en trois types d'agents, proposé dans la littérature, à savoir: des agents de tâches, des agents d'information et des agents d'interface. Au niveau structurel nous avons des agents élémentaires et des agents complexes. Les agents complexes peuvent être considerés comme des organisations d'agents et aussi appartenir aux trois types présentés auparavant, c.a.d. les agents de tâches, agents d'information et agents d'interface.

L'architecture des agents élémentaires respecte mon point de vue d'architecture modulaire et est fondée sur un modèle reutilisable, inspiré du modèle classique des agents BDI (croyances, désirs, intentions) mais aussi d'autres architectures présentées dans la littérature. Elle est constituée de trois modules, un module de raisonnement, un module de planification de coopérations et un module de communication. L'architecture des agents complexes est fondée sur le même type de modules. Les modules planification de coopérations et communication étant exactement les mêmes, la seule différence constitue le module de raisonnement. Celui-ci est composé d'un ensemble d'agents élémentaires qui interagissent afin de résoudre d'une manière coopérative les problèmes pris en compte par un agent complexe. Cette organisation d'agents est constituée de manière dynamique.

La résolution d'une tâche complexe est fondée sur deux types de processus. Un processus top-down qui assure que la décomposition de la tâche en plusieurs sous-tâches est effectuée à travers les différentes couches d'agents (c.a.d. élémentaires et complexes) considerés, alors qu'un processus bottom-up effectue la synthèse de différentes solutions proposées aux différents niveaux.

Plus précisément maintenant, les agents élémentaires que nous proposons en accord avec la méthodologie que nous implémentons sont les suivants:

-Agent Analyse de Données (AAD): un agent de ce type effectue une analyse de données d'entrée.

-Agent Choix de Marque (ACM): un agent de ce type utilise comme entrée des tables multi-critères afin de choisir la méthode appropriée de choix de marque et ensuite effectuer de manière efficace la modèlisation du comportement du consommateur.

-Agent UTASTAR (AUTS): un agent de ce type exécute la méthode multi-critères UTASTAR. Le résultat est une table d'utilités concernant une analyse de produits alternatifs du marché.
-Agent Expert de Marchés (AEM): un agent de ce type choisit une stratégie de marché fondé sur des scénari et d'autres connaissances spécifiques (p.ex., informations sur les canaux de distribution, etc.).

Les agents complexes de notre système sont crées à partir des agents élémentaires ci-dessus et ils sont les suivants:
-Agent Génération de Scénari: un agent de ce type est composé d'un agent ACM, d'au moins un agent AAD et d'un agent AUTS.
-Agent Sélection de Stratégies: un agent de ce type est composé d'un agent ACM, d'un agent AEM et d'un agent AUTS.

6 Marketing

This application has given us the opportunity to develop our proper multi-agent platform and then design and implement a specific multi-agent system in order to implement an original consumer-based methodology for product penetration strategy selection in real world situations. Different aspects and implications of this work are presented in [Matsatsinis & al., 99a; 99b; 99c; 99d; 02].

6.1 Consumer-Based Methodology for Products Penetration Strategy Selection

To support the product development process an original consumer-based methodology has been proposed in [Matsatsinis & Siskos, 99] proposed. It is based on the use of different models for data analysis, multi-criteria analysis and brand personal choice. During the market survey, every consumer expresses his evaluations on a set of reference products involved in the research, on the base of a group of criteria. Finally, he is requested to rank the products according to the order of preference. The collection of this kind of data requires a specific questionnaire. The initial phase of this methodology aims to acquire an overall frame of the particular survey. This is followed by the use of data analysis models in order to determine consumer and market features. This task is called "Market Segmentation". Market trends are identified through this approach. Concurrently, the multi-criteria method UTASTAR [Siskos & Yannacopoulos, 85] is applied to the multi-criteria consumer preferences, in order to determine the criteria explaining each of the consumer's choices. This method assesses a utility function $u(g)$ which is as consistent as possible with the consumer ranking, where $g=(g_1, g_2, ..., g_n)$ is the vector of the criteria on which the products are evaluated. The consumer's utility function is assumed to be additive: $u(g) = p_1u_1(g_1)+p_2u_2(g_2)+...+p_nu_n(g_n)$, where $u_i(g_i)$ is the estimated marginal utility of the criterion g_i, normalized between 0 and 1, and p_i is a weighting factor of the i-th criterion, the sum of weights being equal to one: $\sum^{n}_{i=1}p_i=1$.

The UTASTAR method estimates for each consumer separately his utility function, which is as consistent as possible with his rank order of the products used; the relative importance of the criteria is then derived from this utility model. This preference disaggregation analysis is called "Criteria Analysis". The use of models of consumer personal choice allows the market simulation and the calculation of the market shares of the competitive products taking part in the research. This aims at the selection of the most suitable model approach, as close as possible to the real market shares ("Brand Choice Task"). The next step concerns the design of the new product by simulating its introduction into the market using the multi-criteria estimations. It is followed by the application of alternative scenarios. With the help of the selected brand choice model, the market simulation and the calculation of the new market shares to be expected (after the introduction of the new product), are performed. This process involves "Scenario Generation and Complex Scenario Generation". Based on the results of the scenarios application, the choice of the most appropriate penetration strategy for the new product is made. This is the main task and is called "Penetration strategy selection".

6.2 Agent's Types, Functionalities, Structure

Agents are simultaneously considered according to two different levels: a functional and a structural level. According to the functional level, we have adopted the three types of agents proposed in [Sycara & Zeng, 96]: task agents, information agents and interface agents assuming task's fulfillment through cooperation, information gathering tasks, and mediation between users and artificial agents respectively. In the structural level we have elementary agents based on a generic reusable architecture and complex agents considered as an agent organization created dynamically in a hierarchical way.

In Figure 16 we present an agent-based system used by decision-makers, who can be corporation board members, each simulating his own scenarios and finally selecting a penetration strategy for a new or an existing product in a board meeting.

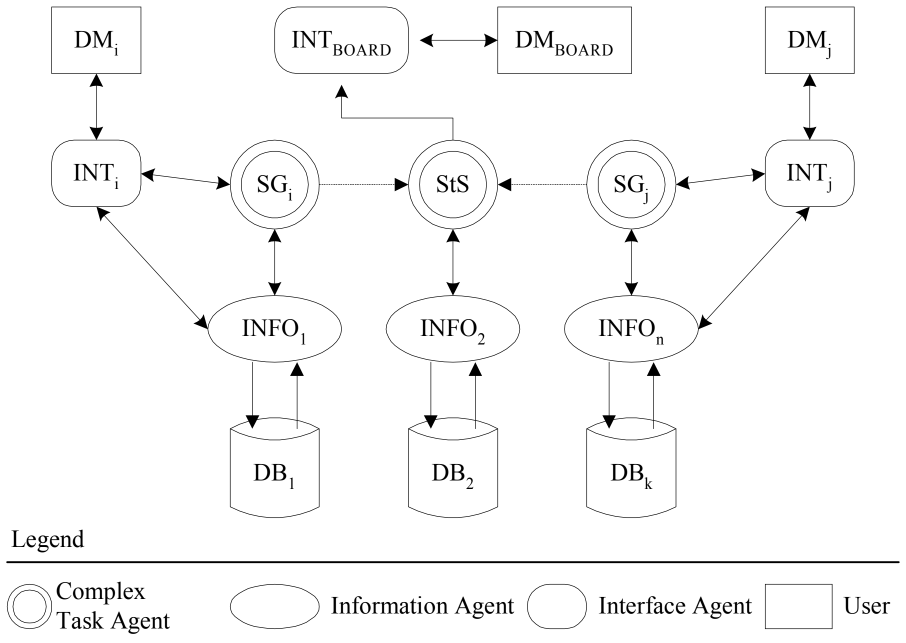


Figure 16: The system's architecture

An agent's knowledge is acquired during a knowledge acquisition stage, using different domain experts knowledge, and through interactions with the other agents of the system as well as the human users. As we have said before, in our approach, agents are considered according to two different levels: a functional level and a structural level.

The functionalities of interface agents are those we can find in the literature (e.g. [Laurel, 1997; Sycara & Zeng, 1996]): initiation of a task, responsibility of system interactions with the user, results presentation to user queries, in a way appropriate to the user's profile (e.g. according to the level of responsibility in an organization), determination of what categories of task agents should be involved, so that a user query is correctly taken into account.

The functionalities of information agents are also those we can find in the literature (e.g. [Knoblock & Ambite, 97; Sycara & Zeng, 96]). Their goal is to provide information and expertise on various topics, by drawing on relevant information from the system's general database, remote heterogeneous databases in the Internet, other information agents or interface agents.

Finally, task agents specialize in performing specific tasks. They can interact with all types of agents in order to carry out their jobs. These are the most sophisticated agents of our system and they can have an elementary or complex structure. For the application presented in this paper, we conceived different types of task agents (elementary and complex) each corresponding to different generic tasks (e.g. perform data analysis, generate a scenario), involved in the methodology presented before. We can have several occurrences performing the same specific task (for example several agents performing data analysis).

We therefore have the following types of elementary task agents:

- Data Analysis (DA) agent: such an agent performs data analysis on an input data set. (see Figure 1, e.g. correspondence analysis, principal components analysis, etc.). Such an agent has the knowledge that enables him to choose appropriate data analysis methods, which are effective on any particular input data set. Finally he can combine and evaluate each applied method's outputs.
- Brand Choice (BC) agent: such an agent uses inputted multi-criteria tables in order to choose the appropriate brand choice model(s) and effectively model the behavior of the consumers that participated in a particular market research (e.g. LUCE, Mc Fadden 1, etc.).
- UTASTAR agent (UTS): such an agent performs the UTASTAR multi-criteria method on an inputted market research. He can locate and distinguish multi-criteria questions while identifying alternative products used in any market research. Its output is the utility table.
- Market Expert agent (ME): such an agent selects a market strategy depending on scenarios and on knowledge that includes corporate information, distribution channels information, etc.

The agents, which are used for the presented application modeling, are based on a generic reusable architecture that we conceived, following my modular conception of an agent architecture [e.g. (Moraïtis, 94; Boussetta, Cohen & Moraïtis, 96)], and inspired by the general BDI type philosophy (see for example [Georgeff & Ingrand, 89]) and the different agent architectures presented in the literature (e.g. [Brazier & al., 97; Sycara & Zeng, 96; Witting, 92]). Different functional agent types have the same basic architecture principles, regardless of the category they belong to. However, the different modules are more or less sophisticated according to their specific type (e.g. the planning model of an information agent is simpler than that of a task agent). Our agent architecture (Figure 17) is composed of three modules (Communication, Planning and Reasoning module) that intercommunicate through internal message exchanging (called intra-agent messages).

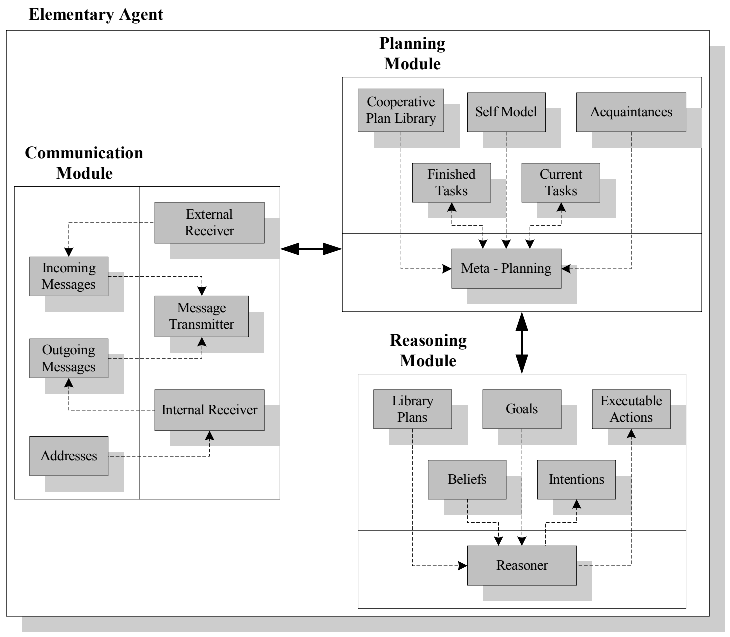


Figure 17: An elementary agent structure

These modules run concurrently. An agent remains idle while no messages arrive to his communication module. As soon as a message arrives, the communication module determines its importance and, after transforming it to an intra-agent message, sends it to the planning module by means of a message queuing mechanism. All modules adopt this behavior and remain idle while no messages are available for procession. The same intra-agent queuing mechanism facilitates all modules. Thus, intra-agent control is achieved via the intra-message mechanism. By using these elementary agents, we build complex agents, taking into account the methodology's complex tasks achievement. We consider that by using the complex agent concept to gather together agents involved in some complex task (if the task's nature allows it) achievement, the system's scale and coordination complexity can be decreased, making the application's modeling easier. Actually, coordination, even within a large-scale application, is carried out, either between agents within relatively small-scale groups or between a reduced number of complex agents that are entities of an upper layer. The involved complex agents are:

- Scenario Generation (SG) agent: such an agent is composed by at least a DA, a BC and an UTS agent. He coordinates the scenario and complex scenario generation task (Figure 18).
- Strategy Selection (StS) agent: such an agent is composed by a BC, a ME and an UTS agent. He coordinates the penetration strategy selection task.

Complex agents can belong to the three functional types defined before. The architecture of a complex agent is similar to the one of an elementary agent. Therefore he is composed of the same three modules (Communication, Planning and Reasoning module), which intercommunicate through internal message exchanging. The intra-agent control (interaction between the three components) is the one of the elementary level.

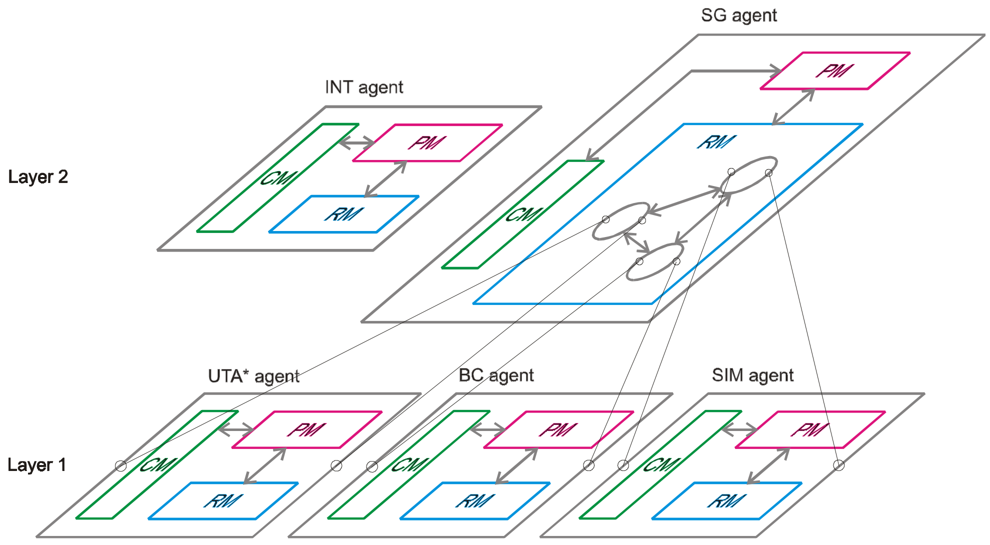


Figure 18: A Complex Agent Structure

The difference is situated in the structure of the reasoning module. The group of agents (elementary and/or complex) that compose it assumes its role. The task achievement of an agent (parent) developed in n-layer is therefore the result of the set of agent's (his descendants) cooperation belonging to the previous (n-1) layer. The reasoning module could be therefore considered as an agent organization.

The interaction between a complex agent's reasoning and planning modules (like in the elementary level) of an (i+1) layer agent is established through the i layer agents that are components of the reasoning module of the (i+1) layer agent (for example the "Brand Choice" Agent sends the results of his work to the planning module of the "Scenario Generation" Agent, Figure 18). In this context, an inter-agent message sent by an i layer agent to an (i+1) layer agent is transformed to an intra-agent message of the (i+1) layer agent. The organization of reasoning module agents is dynamically generated. Its role is to achieve any task(s) allocated by the planning module. The agent's organization generation process is initiated by selecting appropriate agent(s) (according to the task's nature) chosen by the planning module. Agents can be of a different nature (e.g. static, mobile), not necessarily implemented in the locality of the parent agent, but they are, however, aware that they have the same parent. In Figure 18, a UTA* agent, a BC agent and a SIM agent compose the reasoning module of a complex SG agent. The UTA* agent provides the multi-criteria analysis service (using his knowledge on analyzing a market research and selecting multi-criteria questions), the BC agent selects a brand choice model for the multi-criteria analysis (using his knowledge base on model selection), while the SIM agent simulates scenarios for the user (using either its knowledge base or user input provided via an INT agent to the SG agent). The reader will find in [Matsatsinis & al., 03] all the technical details concerning the implementation of complex agents.

Actually, we can say that two processes facilitate a complex task achievement. A top-down process assumes that the task's decomposition in several subtasks is achieved across the different layers, while a bottom-up process performs the synthesis of different solutions proposed at different layers. We can have complex information agents when an information retrieval must be accomplished through the achievement of several specific information-gathering goals. Different specialized information agents representing layers of a complex information agent can take these goals into account. We can also have a complex interface agent able to take into account (through his elementary interface agents) the different points of view of board members during a distributed decision making process.

Services d’Information: le système IMAGE

Résumé

Dans cette section je présente mon travail dans le domaine de services d’informations. Plus précisément mon travail concerne la conception et le développement d’un système multi-agent qui constitue le coeur du système IMAGE. Ce système a comme objectif de fournir aux utilisateurs de services d’informations, mobiles et personnalisées basés sur la localisation des utilisateurs, ainsi que des moyens pour accéder à ces informations mais aussi leur paiement. Alors certains des objectifs principaux pour IMAGE sont:
-Développer des services avancés de navigation, localisation et de commerce éléctronique
-Réorganiser le business model en introduisant l’Agent Mobility qui joue le rôle d’intermédiaire entre les acteurs impliqués

Le système IMAGE est constitué de modules suivants:
-Un module d’interface avec l’utilisateur
-Un module intelligent qui constitue notre système multi-agent
-Un module de services géo-référenciels (localisation/navigation)
-Un module de services de commerce éléctronique
-Un module GIS
-Un module de SGBD

Plus précisément le système multi-agent que nous développons est constitué de différents types d’agents qui sont les suivants:
-Agent Interface: ce type d’agents est le lien avec l’interface utilisateur du système et il peut a) prendre en considération des demandes simples non personnalisées b) authentifier l’utilisateur c) décider si l’utilisateur va être servi par un agent assistant ou assistant personnalisé
-Agent Assistant: ce type d’agents peut prendre en considération des demandes complexes d’utilisateurs qui n’ont pas un profil enregistré
-Agent Assistant Personnalisé: ce type d’agents peut prendre en considération des demandes complexes d’utilisateurs qui ont un profil enregistré. Il sera capable d’adapter le service aux habitudes et préférences personnelles de l’utilisateur, en gardant et exploitant l’historique des services demandés dans le passé.
-Agent Guide de Voyage: ce type d’agents peut prendre en considération des demandes concernant des services géo-référentiels, le GIS et le SGBD.
-Agent Educateur Touristique: ce type d’agents peut prendre en considération des demandes concernant des informations touristiques pour une ville (p.ex., musées, théatres, monuments, etc.).

-Agent Services: ce type d'agents peut prendre en considération des demandes concernant des services tels que le contrôle de disponibilité de tickets, des places, etc.
-Agent Evénements: ce type d'agents reçoit des informations par le SGBD concernant des nouveaux événements et informe les agents assistants personalisés.
-Agent Services SMA: cet agent est responsable de garder un enregistrement et la trace des agents du système et de fournir des informations (sous forme de pages jaunes) sur les agents des autres systèmes IMAGE qui sont géographiquement distribués.

7 Information Services: The IMAGE System

My work in this domain concerns the design and implementation of a multi-agent system, which constitutes the core element of the Image system. This system aims to provide the users with mobile, personalized, location based information services, how to reach them and how to pay for them with flexible mobile and stationary means. To meet this aim the following objectives are defined for Image:
- To design and develop an open and modular service platform, which as a transparent central point co-ordinates both end user data (user request) and service provider data (provider response);
- To develop advanced key services, such as navigation, localization applications, and e-commerce services, and facilitate the easy integration between them;
- To re-organize the business models and relationships at the potential sites by introducing a new business role, the Mobility AGENT, that intermediates between the actors involved in the service delivery (the network of content providers, service and product providers and end users);
- To verify the integrated platform and the inter-operation between Agents of different test-beds;
- To prove the feasibility of the proposed platform through the conduction of thorough financial and marketing analysis;
- To facilitate the further implementation of the Image system across Europe, to devise a strategic plan for achieving this and to provide guidelines for Agent and service providers.

7.1 The IMAGE Agent Scenario

A potential Image customer is able to set an inquiry to the Agent: the satisfaction of his/her need (for example "I would like to visit a museum"). However, for the sake of the flexibility and the modularity of the project's end product the user is enabled to a simper request, such as a travel request only: a journey between two geographical points, O (origin) and D (Destination). The origin could either be defined by the potential customer ("I know where I would like to start my journey") or be a task for the Agent ("I do not know where I am now") by utilizing localization technologies. The destination could be again directly defined by the customer ("I would like to go there, don't ask why") or determined by his/her current needs as mentioned previously ("Museum"), or a combination of both options. The request is accompanied by a set of parameters, containing general profile & general preference, and also current preferences (for example transport mode, optimizing journey time or price, etc.). Nevertheless the Agent is capable of providing the user with new options as well.

The core element of the Image Agent dealing with the users' re-quests is the intelligent module (a multi-agent system), which bases its operation on a geographical information system (GIS platform) that is interpreting the incoming information from both the end user and the service providers into geographic co-ordinates. The same tool is also responsible for tracking the geo-data (transport nodes, tourism activities, retail activities, etc.) of the geographic environment where the Agent is active. Thus the Agent is acknowledging the 3-D background characteristics of the potential customer's travel and also the characteristic points along the route.
User requests are obtained and analyzed into a bundle of separate services by the user interface (UI), which is closely co-operating with the intelligent Agent and the GIS platform.

7.2 General Description of IMAGE Modules

Image investigates the link between current agent research, emerging standards (FIPA, XML, and W3C) and strong user needs in this new mobile world.
The work of implementing the Agent has been divided in the development of the following individual modules: a) User interface, b) Intelligent module, c) geo-referenced services (Localization/Navigation), d) e-commerce/payment services, e) GIS platform and f) Data Management/Interface which in combination are performing all the necessary steps for an integrated service delivery according to the end user's request(s).

The Image Agents will operate between the Image modules:

- User Interface Module
- Intelligent module
- Geo-referenced service module (Localization / Navigation)
- E-commerce service module
- GIS platform module
- Data Management / Interface module.

-User Interface Module (UI)

Image will design and develop a suitable Use interface module (UI) for handling interaction with the user in a way that is consistent across various types of end device. This will entail a task-oriented approach to the interface design, leaving the various device-dependent I/O options open. This should also permit sufficient customization and adaptation of the UI by both the end user and any new service provider who wishes to plug into the Image platform. Such a design should also be particularly suited to new generation hand-held devices, as it will permit appropriate flexibility in the choice of media for presentation to and from the user (text, graphics, keypad, voice, touch or stylus etc.) will develop an interactive, bi-directional User Interface able to present all provided services and analyze the user's request in an consistent way, decomposing it into a set of basic services.

-Intelligent Module (IM)

The intelligent module (IM) (the IMAGE agents) will 'intelligently' manage, process and monitor end users' requests and individual profiles/preferences for geo-referenced and time-dependent servicing. The intelligent module bases its operation on a geographical information system that is interpreting the incoming information from both the end user and the service providers into geographic co-ordinates. The same tool is also responsible for tracking the geo-data (transport nodes, tourism activities, retail activities, etc.) of the geographic environment where the Image Agent is active. The core of the IMAGE is actually the intelligent module's Agents and the user interface module.

-Service Modules for Geo-referencing (Localization LM /Navigation NM)

The services for geo-referencing consist of localization and navigation techniques, which will be developed in Image localization services, aim to acquire the exact position of the end user in the geographic environment. The navigation services aim to guide the end user to the required point of interest (POI) so as to satisfy his/her need. Positioning and navigation are particularly important as transport and tourism services always require geographic references. The necessary components are either newly developed for this or existing components are extended in such a way that they do justice to the Image system requirements i.e. they support the Agents to a large extent. This requires the development of new features and interfaces.

-E-Commerce/Payment Service Module (EM)

Many of the Image services will require some form of financial transaction, either before, during or at some stage after the service has been provided to the user. To achieve this a standard set of rules and a technical realization of an E-commerce sub-module for Image will be investigated and implemented. The main idea is that each service provider takes care of the payment functions related to their own services. Image will provide a payment procedure only for the Image services. Image services are such services, which are provided by the Image server (customer identification, customer profile etc)

The E-commerce/payment service module (EM) will use the identified modes of payment used with Image Agent. The roles of the different players/levels of the services will be analyzed in order to develop the most appropriate common solution for the Image Agents. For the E-commerce/payment it is important to define a common set of interfaces (new and standardized) and security solutions between the different levels of the Image system. The E-commerce module will depend on the payment services offered by other vendors and the user payment preferences. Special attention will be paid on the different laws applied on money transfer across the European countries. The E-commerce/payment module provides a common mechanism for the transaction data exchange and clearing functions in a multi-application environment. This segregation is required due to the enhanced security considerations for financial transactions.

-GIS Platform Module (GIS)

The GIS platform will initially be developed on a test-bed dependent basis. A robust handling of spatial information will assure the satisfaction of the end user request through a set of tools for retrieval and processing of the relevant data and production of the results. The GIS infrastructure will support additional information of different nature to be linked to specific locations.

The project's philosophy being to design and develop an open GIS platform is fully compliant to the Open GIS Consortium's (OGC) objective to remove barriers in access and use of spatial information implied by the proliferation of software and data standards for geo-information. The Internet will be used as a standard access means for all project related information and the database developed can be used elsewhere and implemented with any technology and software tools that support the Open GIS specifications.

-Data Management/Interface Module (DM)

Image will develop a data management module that will interface with external entities databases in order to deliver the required data. The project data sources are both private and public and the data, which will have to be integrated and processed, have been built on different standards and concepts. Image will propose a standard travel and traffic data access interface allowing the connection to the data in a uniform way, independently of the providers' set-up. The goal is to develop an open platform in the sense that various formats and data capture and management concepts will be supported. A set of generic thoroughly analyzed interfaces will be implemented in order to assure that data retrieval will be successfully carried out beyond the planned operations. Specifically, with respect to the point of view of the e-commerce data, the Data Management Module mainly manages all related information from and to the external applications and/or to the E-commerce/payment module.

The data interfaces will be developed so that they in combination will perform all the necessary steps for an integrated service delivery according to the end user's request(s). Generic data interfaces include components that work with heterogeneous and new data coming from various sources/databases. The data interfaces will form part of the actual Image platform, from where different kinds of intelligent agents (forming the intelligent module) can retrieve data that fits the user's requirements. Standards (e.g. DATEX) and ongoing work (e.g. TRIDENT) in the domain of travel and traffic data exchange will be taken into particular account while defining and implementing the data interfaces. They will greatly help in the process and will drive Image towards solutions that are interoperable with existing and forthcoming systems. The Data Management module will be capable to push information regarding new traffic or recreational city events to the Intelligent module.

7.3 The Intelligent Module Architecture

In order to take into account the requirements concerning the intelligent module, we have proposed a set of intelligent agents types, which are shown in the overall Image architecture diagram (Fig. 19). More specifically:

Interface Agent type: This agent is the link with the user interface, he can a) handle simple not personalized queries, b) authenticate the user, c) decide if he will be served by an assistant or a personalized assistant and finally he can forward messages from the user interface to the assistants and messages from the assistants to the user interface.

Assistant Agent Type: This agent can handle complex queries from users who do not have a profile.

Personalized Assistant Agent type: This agent can handle complex queries from users who do have a profile. He will be able to adapt the service according to user's habitual patterns, by keeping and processing the history of service requests and profile modes of the particular user, presenting, thus, the user with a personalized.

Travel Guide Agent type: This Agent handles queries relevant to the Geo-referenced services, the GIS and the Data Management/Interface module (wraps them).

Educator/Tourist agent type: This Agent handles queries (wrapping the Data Interface) relevant to tourist information for a city (e.g. museums, theatres, monuments, etc).

Services Agent type: This agent handles queries (wrapping the Data Interface) relevant to services like checking for availability of tickets/seats.

Events Handler Agent type: He receives new events from the data management module and subsequently informs the personalized user assistant agents.

Any service operator/provider will be able to contact the interface agent and enjoy the Image services. The system (see Figure 19) will function as in the case of a user interface request.

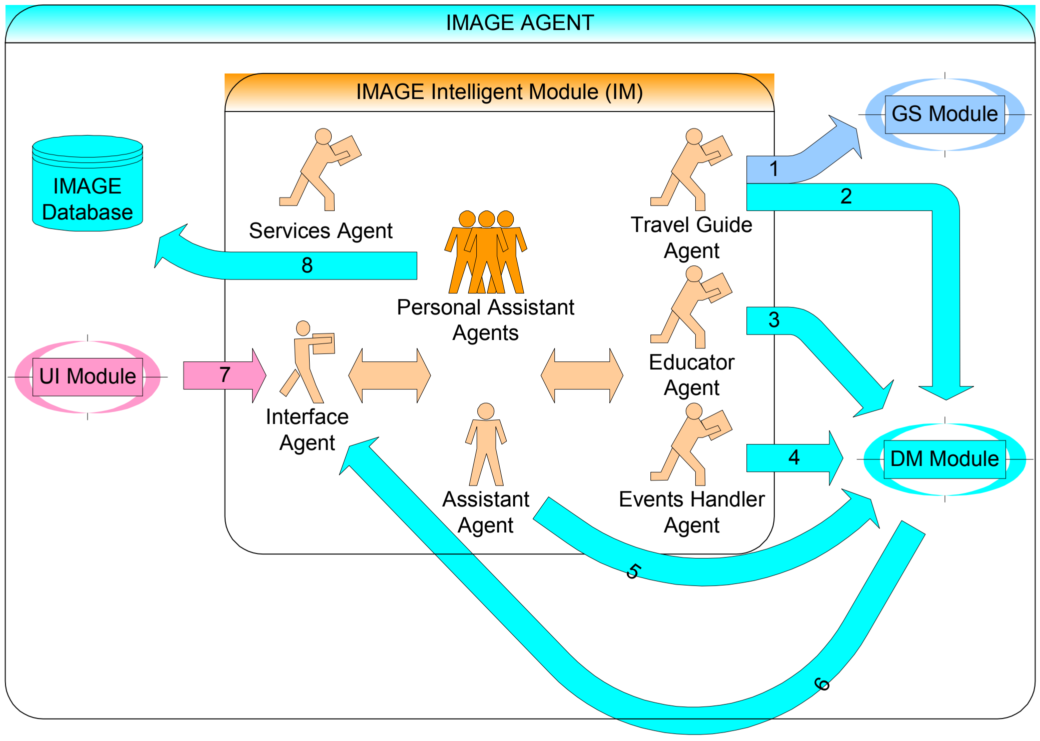


Figure 19: Agents and their relationships with the other modules

Presently, our work concerns the design and implementation of the previously defined agents. As I will present in Section 9, we have combined the Gaia methodology [Wooldridge, Jennings & Kinny, 00] and the JADE framework [Bellifemine & al., 02] for the analysis, design and implementation of our multi-agent system. Simultaneously, our work concerns the components and technologies that will enable communication with external to MAS systems. More precisely:

- The UI (user interface), which will enable users to access the MAS services through mobile devices (PDAs, mobile phones) and the Internet. The UI invokes MAS services by exchanging XML messages through plain TCP/IP sockets.
- The DM (Data management module), which offers on-line traffic data, access to user position through GPS, GIS information regarding points of interest (theatres, banks, hotels, etc). The DM pushes data (like new events) to the IM using XML messages through plain TCP sockets while the IM accesses DM services through SOAP.
- The GS (Geo-services), which include the address geocoding, map generation and route calculation functions through SOAP/XML messaging.

Diagnostic

Résumé

Dans cette section je présente mon travail dans le domaine du diagnostic. Plus précisément je présente l'architecture des agents conçus et développés ainsi que des éléments sur le modèle de coordination utilisé afin de développer un système muti-agent pour la surveillance d'un réseau téléphonique. En effet l'idée était d'associer des agents aux différents équipements du réseau, qui analyseraient d'une manière coordonnée les alarmes émises par ces équipements, afin d'aider les spécialistes à effectuer un meilleur et plus rapide diagnostic.

Le concept du plan est utilisé dans le raisonnement de l'agent ainsi que dans ses interactions avec les autres agents du système (c.a.d. modèle de coordination). Un plan modélise donc les connaissances de l'agent en tant qu'entité individuelle et sert de support aux échanges nécessaires à la résolution distribuée d'un problème. Le formalisme de représentation de plans retenu est celui des réseaux de Petri récursifs. Un réseau de Petri ordinaire est composé d'un ensemble de transitions et d'un ensemble de places. Les transitions correspondent aux actions à entreprendre et les places aux différents états du système modélisé.

Les réseaux de Petri récursifs permettent de distinguer des transitions abstraites raffinables en un nouveau plan, des transitions élémentaires correspondant aux actions à éxecuter et des transitions de fin qui achèvent un plan. Notre modèle d'agent est composé de trois modules assurant des fonctionnalités différentes: le module de planification, le module de décision et le module de communication. Le module de planification assure l'éxecution de plans. Le module de décision contient l'expertise de l'agent et effectue le raffinement des actions abstraites (c.a.d. les transitions abstraites) d'un plan. Finalement le module de communication assure la communication inter-agents par envoi de plans. Le modèle de coordination fondé sur les réseaux de Petri récursifs permet la spécification des activités concurrentes et le raisonnement sur des actions simultanées et des processus continus

8 Diagnosis

My work in this domain concerns the design and implementation of a multi-agent system for fault diagnosis and monitoring of a telecommunication network. Actually, the idea was to associate agents to different equipments in a telecommunication network (i.e. the French telecommunication network), who will analyze in a coordinated way the alarms issued by these equipments, in order to help the specialists making a better and faster diagnosis. Due to confidentiality reasons, I will focus the presentation of this work [Boussetta & al., 98; Boussetta, Cohen & Moraïtis, 96; Moraïtis, Boussetta & Cohen, 95a; 95b; 95c; Haddad & al., 95a; 95b; Boussetta, Mazouzi & Moraïtis; 96; Moraïtis & al., 96; Haddad & al., 96] only on the structure of the agents and the coordination model (which is based on distributed planning).

8.1 The Agents Architecture

All agents have the same structure. It is only the underlying domain knowledge of each agent that differentiates them. An agent is composed of three modules: the planning module, the decision module and the communication module (see Fig. 20). This architecture is inspired by the COSMIMA architecture, presented in [Moraïtis, 94]. The three modules run concurrently. Interaction among modules is based on a message queuing mechanism (which is associated to each module).

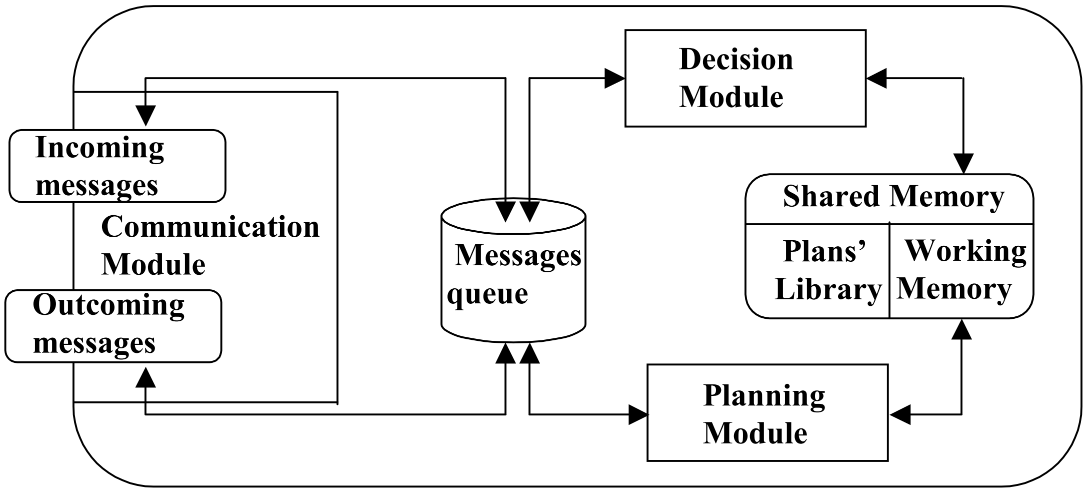


Figure 20: Agents architecture

Modules have access to a shared memory, which is divided to two parties: the plans' library and the objects managed by the agent. Modules communicate by notification sending (i.e. intra-agent messages). These notifications are sent in the queue that is associated to the modules concerned.

-The Planning Module

This module ensures the execution of plans. An agent interleaves planning and execution. The knowledge of the agent consists of the domain objects and the plans of actions that he manipulates. Plans are generic (non instantiated) and grouped in a library. They are instantiated by the decision module by using the domain objects. The adopted model of plans is that of Recursive Petri Nets, presented in [El Fallah & Haddad, 96]. So, the know-how of the agent resides in plans' skeletons defined during the creation phase and methods associated to actions (the reader will find a more detailed description in the next section).

-The Decision Module

The main functionality of the decision module is the refinement of an abstract action of a plan. It is the planning module that notifies the need of refinement. A notification arrival can generate an immediate refinement, a pending refinement (until some missing resources are available), or a refinement achievement (when the notification informs about the existence of the missing resources). A need of refinement corresponds to the instantiation of the variables of the possible plans' skeletons, (which form candidate plans) and the choice of a candidate plan.

-The Communication Module

The communication module ensures the perception of the environment and the exchange of messages among agents (by using his acquaintances list). Its mechanism relies on two processes managing the incoming and outcoming communications. The incoming communication process is "listening" for messages coming from the environment or the other agents. Once a message arrives, it transforms it in the appropriated form and prepares a notification, which is put in the queue of the module concerned. The flow of the notifications depends on the type of the messages and it is therefore independent of the messages' content. The outcoming communication process handles the demands for interaction with the other agents that exist in the queue (i.e. plan merging demand, object sending, plans sending, etc.). Taking into account the content of notifications, it prepares the messages to be sent (after having consulted the acquaintances list of the agent).

8.2 Agents Coordination

Agents' coordination is based on a formal model of representation and handling of plans [Boussetta & al., 98]. It is based on Recursive Petri Nets (RPT) (see [El Fallah & Haddad, 96]), which support the specification of concurrent activities, reasoning about simultaneous actions and continuous processes, a theory of verification and a mechanism of transformation (e.g. abstraction, refinement, merging). The main features of the RPN formalism are domain independence, broad coverage of interacting situations and operational coordination. So a RPN models a plan. More precisely a plan organizes a collection of actions, which can be performed sequentially or concurrently in some specific order. A plan involves both elementary actions associated with irreducible tasks (but not instantiated ones) and complex actions (i.e. abstract views of the task). Methods may be viewed as some way to perform an action. Several methods can be associated with an action. In RPN formalism, nodes represent the states of the system (i.e. resources, processes, etc.), while transitions model actions and their firing correspond to executing these actions. There are three types of transitions: elementary, abstract and end transitions. Elementary transitions correspond to actions to be executed by the agent, while abstract transitions correspond to new plans of actions. These plans are located in the agent himself or in his acquaintances. Finally end transitions close the subnets.

In [Boussetta & al., 98], we show how an approach for interleaving of planning and execution, based on the RPN semantics, can be used to coordinate agents' plans in a dynamic environment.

The RPN formalism provides a graphic expression of the synchronization of the parallel activities, which represent the execution of the agents' tasks. It also allows the expression of the actions' scheduling (causal or temporal relation order) as well as the information sharing among agents. In addition, it authorizes the dynamic assignment of tasks and the update of the plans of resolution by merging or refinement. The hierarchical aspect of the RPN allows us to consider a distributed system at different levels of abstraction and to refine the abstract transitions (i.e. an abstract transition can be seen as a subnet), when necessary. In addition, the approach we have adopted allows us to dynamically choose among several concurrent plans based on the structural properties of the RPN.

In our application domain (i.e. the surveillance of a telecommunication network), each agent executes constantly a plan called "supervising". This plan is generally refined after the arrival of an alarm. Figure 21 gives an example of a RPN structure along with a possible refinement for the alarm "A1" by the plan P1.

.

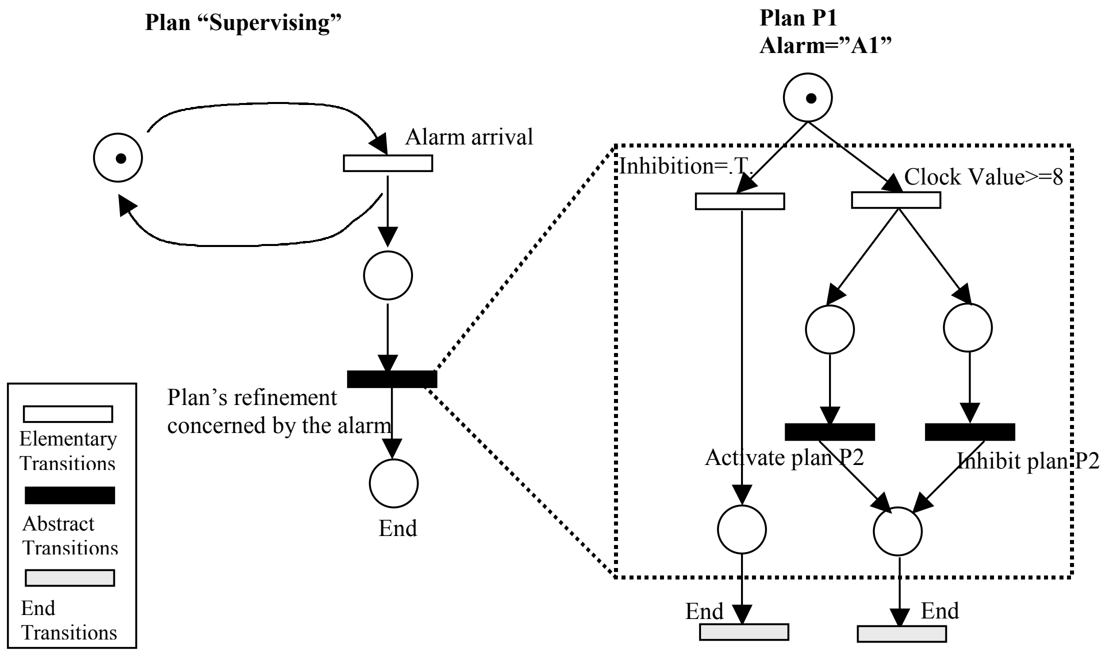


Figure 21: Example of plan's execution

Génie Logiciel Orienté Agent

Résumé

Dans cette section je présente un travail experimental dans le domaine du génie logiciel orienté agent. Plus précisément il s'agit du travail que nous effectuons dans le cadre du projet IST IMAGE. Ce travail concerne la combinaison de la méthodologie Gaia , qui couvre les phases d'analyse et de design du génie logiciel et la plate-forme JADE qui couvre la phase d'implémentation, pour une analyse, un design et une implémentation faciles, de systèmes multi-agents. Ainsi nous avons essayé de partager notre expérience avec ceux qui sont intéressés par le développement des systèmes réels, fondés sur le concept d'agent.

Gaia est une méthodologie générale qui supporte à la fois les deux niveaux, à savoir la structure individuelle de l'agent et la société d'agents, dans le processus de développement de systèmes multi-agents. L'objectif du processus d'analyse de Gaia , est l'identification de rôles et la modélisation des interactions entre les rôles définis. Les rôles sont constitués de quatres attributs: responsabilités, permissions, activités et protocoles. Les responsabilités sont de deux types: proprietés "liveness"-le rôle doit ajouter quelque chose de positif au système et proprietés de "safety"-le rôle doit éviter que quelque chose de mauvais arrive au système. Liveness décrit les tâches qu'un agent doit achever étant donné les conditions environementales, alors que safety assure qu'un état d'affaires acceptable est maintenu durant le cycle d'éxecution.

JADE est un cadre de développement logiciel où les tâches ou les intentions de l'agent sont implémentées avec l'utilisation de comportements. Les comportements sont des "threads" logiques d'éxecution qui peuvent être composés de diférentes façons afin d'achever des combinaisons d'exécution complexes et ils peuvent être initialisés, suspendus ou engendrés à chaque moment..

L'idée générale, sur laquelle est fondée la combinaison de Gaia et JADE que nous proposons, concerne la traduction de toutes les formules définisant les proprietés de liveness en comportements JADE ainsi que la dédicace de comportements à l'assurance des conditions de safety. Cette idée, avec tous les autres éléments conceptuels ou techniques nécessaires pour sa mise en œuvre, est utilisée pour développer notre système multi-agents, qui comme nous l'avons signalé dans la section précédente, constitue le cœur du système IMAGE.

9 Agent Oriented Software Engineering (AOSE)

Agent Oriented Software Engineering (AOSE) is one of the fields of the agent domain with a continuous growing interest. The reason is that the possibility to easily specify and implement agent-based systems is of a great importance for the recognition of the add-value of the agent technology in many application fields. During the last few years, there has been a growth of interest in the potential of agent technology in the context of software engineering. This has led to the proposal of several development environments to build agent systems (e.g., Zeus [Collis & Ndumu, 99]; RETSINA [Sycara & al., 02], [Occello & al., 02], etc), software frameworks to develop agent applications in compliance with the FIPA specifications (see for example FIPA-OS [http://fipa-os.sourceforge.net], JADE [Bellifemine & al., 02], etc) and of some promising agent-oriented software development methodologies, as Gaia [Wooldridge, Jennings & Kinny, 00], AUML [http://www.auml.org], Tropos [Giunchiglia, Mylopoulos & Perini, 02], MASE [Wood & DeLoach, 00]. However, despite the possibilities provided by these methodologies, we believe that a further progress must be made, so that agent-based technologies realize their full potential, concerning the full covering of the software life cycle and the proposal of standards to support agent interoperability.

In this work presented in [Moraïtis, Petraki & Spanoudakis, 02] we have presented an attempt towards this direction, by proposing a kind of roadmap of how one can combine the Gaia methodology for agent-oriented analysis and design and JADE, a FIPA compliant agent development framework, for an easer analysis, design and implementation of multi-agent systems. Our objective was to share our experience to conceive and develop a MAS, by combining Gaia and JADE, in the context of the IST IMAGE project (presented above), with people who are interested in the development of real life agent-based systems.

The Gaia methodology is an attempt to define a complete and general methodology that it is specifically tailored to the analysis and design of multi-agent systems (MASs). Gaia is a general methodology that supports both the levels of the individual agent structure and the agent society in the MAS development process. MASs, according to Gaia, are viewed as being composed of a number of autonomous interactive agents that live in an organized society in which each agent plays one or more specific roles. Gaia defines the structure of a MAS in terms of a role model. The model identifies the roles that agents have to play within the MAS and the interaction protocols between the different roles.

The objective of the Gaia analysis process is the identification of the roles and the modeling of interactions between the roles found. Roles consist of four attributes: responsibilities, permissions, activities and protocols. Responsibilities are the key attribute related to a role since they determine the functionality. Responsibilities are of two types: liveness properties – the role has to add something good to the system, and safety properties – the role must prevent and disallow that something bad happens to the system. Liveness describes the tasks that an agent must fulfill given certain environmental conditions and safety ensures that an acceptable state of affairs is maintained during the execution cycle. In order to realize responsibilities, a role has a set of permissions. Permissions represent what the role is allowed to do and in particular, which information resources it is allowed to access. The activities are tasks that an agent performs without interacting with other agents. Finally, protocols are the specific patterns of interaction, e.g. a seller role can support different auction protocols. Gaia has formal operators and templates for representing roles and their attributes and also it has schemas that can be used for the representation of interactions between the various roles in a system.

The operators that can be used for liveness expressions-formulas along with their interpretations are presented in Table 1.

Operator	**Interpretation**
x . y	x followed by y
x | y	x or y occurs
x*	x occurs 0 or more times
x+	x occurs 1 or more times
x^{ω}	x occurs infinitely often
[x]	x is optional
x || y	x and y interleaved

Table 6. Gaia Operators for Liveness Formulas

In the Gaia design process the first step is to map roles into agent types and to create the right number of agent instances of each type. An agent type can be an aggregation of one or more agent roles. The second step is to determine the services model needed to fulfill a role in one or several agents. A service can be viewed as a function of the agent and can be derived from the list of protocols, activities, responsibilities and the liveness properties of a role. Finally, the last step is to create the acquaintance model for the representation of communication between the different agents. The acquaintance model does not define the actual messages that are exchanged between the agents it is rather a simple graph that represents the communication pathways between the different agent types.

JADE is a software development framework fully implemented in JAVA language aiming at the development of multi-agent systems and applications that comply with FIPA standards for intelligent agents. JADE provides standard agent technologies and offers to the developer a number of features in order to simplify the development process:

- Distributed agent platform. The agent platform can be distributed on several hosts, each one of them executes one Java Virtual Machine.
- FIPA-Compliant agent platform, which includes the Agent Management System the Directory Facilitator and the Agent Communication Channel.
- Efficient transport of ACL messages between agents.

All agent communication is performed through message passing and the FIPA ACL [www.fipa.org] is the language that is used to represent the messages. Each agent is equipped with an incoming message box and message polling can be blocking or non-blocking with an optional timeout. Moreover, JADE provides methods for message filtering. The developer can apply advanced filters on the various fields of the incoming message such as sender, performative or ontology.

In JADE, agent tasks or agent intentions are implemented through the use of behaviours. Behaviours are logical execution threads that can be composed in various ways to achieve complex execution patterns and can be initialized, suspended and spawned at any given time. The agent core keeps a task list that contains the active behaviours. JADE uses one thread per agent instead of one thread per behaviour to limit the number of threads running in the agent platform. A scheduler, hidden to the developer, carries out a round robin policy among all behaviours available in the queue. The behaviour can release the execution control with the use of blocking mechanisms, or it can permanently remove itself from the queue in run time. Each behaviour performs its designated operation be executing the core method action().
Behaviour is the root class of the behaviour hierarchy that defines several core methods and sets the basis for behaviour scheduling as it allows state transitions (starting, blocking and restarting). The children of this base class are SimpleBehaviour and CompositeBehaviour. The classes that descend from SimpleBehaviour represent atomic simple tasks that can be executed a number of times specified by the developer. Classes descending from CompositeBehaviour support the handling of multiple behaviours according to a policy. The actual agent tasks that are executed through this behaviour are not defined in the behaviour itself, but inside its children behaviours.

When moving from the Gaia model to an implementation using the JADE framework we have to make some assumptions and definitions. Let's consider the liveness part of each role as its behaviour (usually having the same name with the role) in correspondence with the JADE terminology. Thus a simple or a complex behaviour represents each role. This behaviour is considered as the top-level behaviour of the role. Each behaviour may contain other behaviours, as in the JADE behaviours model. Let the contained behaviours be called lower level behaviours. The ω and || operators on Gaia liveness formulas now have the following meaning. The ω means that a lower level behaviour is added by the behavior that contains it in the Gaia liveness formula and is only removed from the agent's scheduler when the behavior that added it, is removed itself. If such behaviours are more than one, they are connected with the || symbol which denotes that they execute "concurrently". Concurrency in JADE agent behaviours is simulated. As noted before, only one thread executes per agent and behaviour actions are scheduled in a round robin policy.

The procedure we have proposed is quite straightforward. All Gaia liveness formulas are translated to JADE behaviours. Activities and protocols can be translated to JADE behaviours, to action methods or to simple methods of behaviours. The behaviours that start their execution when a message arrives, can receive this message either at the beginning of the action method (simple behaviours) or by spawning an additional behaviour whose purpose is the continuous polling of the message box (complex behaviours). For behaviours that start by a message from the Graphical User Interface (GUI), a GUI event receiver method should be implemented on the agent that starts the corresponding behaviour. Finally, those behaviours that start by querying a data source, or by a calculation, should be explicitly added by their upper level behaviour.

The safety properties of the Gaia roles model must be taken into account when designing the JADE behaviours. Some behaviours of the role, in order to execute properly, require the safety conditions to be true. Towards that end, one at least behaviour is responsible for monitoring each safety condition of a role. Whenever a safety condition is found to be false, the functionality of the behaviours that depend on this safety condition is suspended and the monitoring behaviour initializes a procedure that will reestablish the validity of safety conditions. This procedure, for instance, can be the addition to the agent scheduler of a specific behaviour that will address the task of restoring the validity of safety conditions. In general, this procedure depends on the nature of the implemented system and the safety conditions. When the safety conditions are restored, the suspended functionalities are reactivated.

Summarizing, the following steps should be followed in order to easily translate a Gaia model to a JADE implementation:

- Define all the ACL messages by using the Gaia protocols and interactions models.
- Design the needed data structures and software modules that are going to be used by the agents by using the Gaia roles and agents models.
- Decide on the implementation of the safety conditions of each role.
- Define the JADE behaviours. Start by implementing those of the lowest levels, using the various Behaviour class antecedents provided by JADE. The Gaia model that is useful in this phase is the roles model. Behaviours that are activated on the receipt of a specific message type must either add a receiver behaviour, or receive a message (with the appropriate message filtering template) at the start of their action. Gaia activities that execute one after another (sequence of actions that

require no interaction between agents) with no interleaving protocols can be aggregated in one activity (behaviour method or action). However, for reusability, clarity and programming tasks allocation reasons, we believe that a developer could opt to implement them as separate methods (or actions in an FSM like behaviour).

- Keep in mind that Gaia roles translated to JADE behaviours are reusable pieces of code. In our system, the same code of the behaviours GetAcquainted and MeetSomeone will be used both for the personal assistant and events handler agents.
- At the setup method of the Agent class invoke all methods (Gaia activities) that are executed once at the beginning of the top behaviour (e.g. RegisterDF). Initialize all agent data structures. Add all behaviours of the lower level in the agent scheduler.

10 Conclusion

In this document, I have presented my work done during the last eight years. My effort has been led by my intention to produce theoretical work on different aspects concerning the concept of agent, both as individual and social entity. Moreover, I wanted my work to be useful in the development of a formal multi-agent theory. In parallel, I permanently had the anxiety to keep in touch with the applications domain, trying to prove the applicability of my theoretical results within specific environments.

What has been presented in this document could be recapitulated as follows:

- A point of view about agent architectures based on a modular structure, where each module is responsible for one of the possible capabilities an agent may have (e.g. problem solving, cooperation, communication, etc.) and, therefore, responsible for a part of the overall agent's behavior. This behavior is the result of the different modules interaction. This point of view also suggests the integration of another specific module in the agents' architectures, which is dedicated to the agent's personality and shows how this can have an influence on the other modules.
- The completion of the above point of view with the observation that decision-making is a common characteristic of several deliberation processes involved in the operation of the modules and the idea to propose a unified argumentation based model for their representation.
- The presentation of this argumentation model which is based on the extension of the Logic Programming without Negation as Failure (LPwNF) framework, the integration of the concepts of roles and context to it as well as its use to model personalities.
- A dynamic planning model based on graph representation, taking into account changes that are generated not only by the environment but also by the agent himself and its exploitation to build a multi-criteria distributed planning framework.
- A multi-criteria negotiation, as well as an argumentation-based negotiation approach. The first approach is mainly used in a distributed planning context, but is also able to find a compromise between agents having work and private goals in a more general setting.
- A logical framework for modeling of complex dialogues, adopting the modular agent's architecture and associating each type of the possible dialogues (i.e. negotiation, persuasion, deliberation, etc.) to a specific module, ensuring the automated generation of dialogues and allowing the representation of embedded dialogues.

- Applications of the presented theoretical results in different domains, like e-commerce, marketing, information services, diagnostic and some experimental work on agent software engineering.

This work reflects the road I have covered towards my scientific and professional objectives. As the reader can imagine, the road to go is still too long. I do not know if the chosen itinerary will lead me to the objectives I would like to attain. But at the end of the day, this is not the most important thing. As the Greek poet *Konstantinos Kavafis* says in his "*ITHAKA*":

"WHEN YOU SAIL FOR ITHAKA,
WISH THAT YOUR TRIP BE LONG,
FULL OF ADVENTURES, FULL OF KNOWLEDGE.
THE LAISTRYGONIANS AND THE CYCLOPES,
ANGRY POSEIDON DO NOT FEAR; THINGS
LIKE THESE ON YOUR TRIP YOU'LL NEVER FIND
IF YOUR THOUGHTS ARE PURE, IF ECLECTIC
EMOTIONS FILL YOUR HEART AND YOUR MIND.
THE LAISTRYGONIANS AND THE CYCLOPES,
ANGRY POSEIDON YOU WILL NOT MEET
IF YOU DO NOT CARRY THEM IN YOUR HEART,
IF YOUR MIND IS NOT FILLED WITH THEM.

WISH THAT YOUR TRIP BE LONG.
MANY A SUMMER MORNS ARRIVE
THAT WITH JOY AND PLEASURE YOU ENTER
INTO PORTS THAT YOU'VE NEVER SEEN BEFORE;
TO STOP BY PHOENICIAN TRADING POSTS
AND BUY THINGS OF VARIOUS SORTS:
MOTHER OF PEARL AND CORALS, EBONY AND AMBER,
AND HEDONIC PERFUMES OF ALL SORTS -
AS MANY AS YOU CAN CARRY SENSUAL PERFUMES;
MANY AN EGYPTIAN CITY YOU MUST SEE,
AND FROM THE EXPERTS LEARN AND LEARN.

FOREVER ITHAKA MUST BE IN YOUR MIND.
TO GET THERE IS THE GOAL OF YOUR TRIP.
BUT DO NOT HURRY YOUR JOURNEY AT ALL.
IT IS BETTER IF IT WERE TO TAKE MANY YEARS;
AND YOU AN OLD MAN TO FINALLY ANCHOR THERE,
RICH WITH WHAT YOU GATHERED FROM THIS TRIP,
EXPECTING NO WEALTH THAT ITHAKA WILL GIVE YOU.

ITHAKA ALREADY GAVE YOU THAT GREAT TRIP.
WITHOUT HER, YOU WOULD HAVE NEVER SAILED AT ALL.
BUT SHE HAS NOTHING ELSE TO GIVE YOU FROM NOW ON.

AND IF YOU FIND HER POOR, SHE DIDN'T MISLEAD YOU.
SO WISE THAT YOU ALREADY ARE, SO EXPERIENCED,
YOU NOW COMPREHEND WHAT ITHAKAS REALLY ARE "

(Translated by A. Moskios)

Personal References

1. Kakas A., Moraïtis P., (03). "Agents Negotiating via Argumentation", in *Journal of Autonomous Agents and Multi-Agent Systems, (JAAMAS*), 2003, submitted.
2. Moraïtis P., Tsoukiàs A., (02a). "A Formal Model of Dynamic Planning for Autonomous Agents", in *Journal of Artificial Intelligence Research*, 2002, submitted.
3. Moraïtis P., Tsoukiàs A., (02b). "Multi-Criteria Distributed Planning", in *Journal of Autonomous Agents and Multi-Agent Systems, (JAAMAS*), submitted.
4. Kakas A., Moraïtis P., (03). "Argumentation Based Decision Making for Auronomous Agents", in *2nd International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'03),* pp 883-890, Melbourne, Australia
5. Karacapilidis, N, Moraïtis P., (04). "Inter-Agent Dialogues in Electronic Marketplaces", in *Journal of Computational Intelligence,* Vol. 20, No 1, 2004.
6. Matsatsinis N, Moraïtis P., Psomatakis V., Spanoudakis N, (03). "A Multi-Agent System for Products Penetration Strategy Selection", in *Applied Artificial Intelligence Journal,* 17:901-925, 2003.
7. Moraïtis P., Petraki E., Spanoudakis N., (02). "Engineering JADE Agents with Gaia Methodology", in *International Workshop on Agent Technology and Software Engineering, (AgeS'02),* Germany, 2002.
8. Kakas A., Moraïtis P., (02a). "Argumentative Agent Deliberation, Roles and Context " in *Computational Logic in Multi-Agent Systems (CLIMA02)*, Copenhagen, Denmark, 2002. Also in *Electronic Notes on Theoretical Computer Science*, 70, No. 5, (2002).
9. Kakas A., Moraïtis P., (02b)."Argumentative Deliberation for Autonomous Agents" in *ECAI'02 Workshop on Computational Models of Natural Argument,* pp.65-74, Lyon, France, 2002.
10. Karacapilidis, N, Moraïtis P., (02a)."Engineering Issues in Inter-Agent Dialogues", in *15th European Conference on Artificial Intelligence (ECAI'02)*, pp. 58-62, Lyon, France, 2002.
11. Della Croce, F., Tsoukiàs A., Moraïtis P., (02). "Why is difficult to make decisions under multiple criteria" in *AIPS'02 Workshop on Planning and Scheduling with Multiple Criteria,* pp. 41-45, Toulouse, 2002.

12. Karacapilidis, N., Moraïtis P, (02b). "Modeling Dialogues in Multi-Agent Systems", *in First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'02),* pp. 798-799, Bologna, Italy, 2002.

13. Karacapilidis, N., Moraïtis P., (01a). "Building an Agent-Mediated Electronic Commerce System with Decision Analysis Features", in *Decision Support Systems Journal,* 32 (2001), pp. 53-69, 2001.

14. Karacapilidis, N., Moraïtis P., (01b). "Intelligent Agents for an Artificial Market System", in *5th International Conference on Autonomous Agents (Agents'01),* Montreal, Canada, pp. 592-599, 2001.

15. Karacapilidis, N., Moraïtis P., (00a). "Intelligent Agents Acting as Artificial Employers in an Artificial Market", in *Journal of E-Commerce Research*, *Special Issue on Intelligent Agents in E-Commerce*, Vol. 1, N^{o} 4, November 2000.

16. Moraïtis P., Tsoukias A., (00). "Graph Based Representation of Dynamic Planning", in *14th European Conference on Artificial Intelligence (ECAI 2000),* Berlin, Germany, pp. 516-520, 2000.

17. El Fallah Seghrouchni A., Moraïtis P., Tsoukias A., (00). "An Aggregation-Disaggregation Approach for Automated Negotiation in Multi-Agent Systems", in *International ICSC Symposium on Multi-Agents and Mobile Agents in Virtual Organizations and E-Commerce (MAMA'2000),* Wollongong, Australia, 2000.

18. Karacapilidis, N., Moraïtis P., (00b). "On the Development of Intelligent Agents for a Web-based Electronic Market System", in *4th Pacific Asia Conference on Information Systems (PACIS-2000),* Hong Kong, pp. 687-702, 2000.

19. Moraïtis P., Tsoukias A., (99). "Dynamic Planning Model for Agent's Preferences Satisfaction: First Results", in *Intelligent Agent Technology: Systems, Methodologies and Tools*, World Scientific Company, J. Liu (Ed.), pp. 182-191, 1999 (in *1st Asia-Pacific Conference on Intelligent Agent Technology (IAT'99)*, Hong Kong, 1999; nominated for the Best Paper Award).

20. Matsatsinis N, Moraïtis P., Psomatakis V., Spanoudakis N, (99a). "Intelligent Software Agents for Products Penetration Strategy Selection", in *Modelling Autonomous Agents in a Multi-Agent World (MAAMAW'99)*, short paper, Spain, 1999.

21. El Fallah Seghrouchni A., Moraïtis P., Tsoukias A., (99). "Multi-Criteria Negotiation for Agent-Based Decision Makers", in *5th International Conference of the Decision Sciences Institute*, Athens, *(DSI'99)*, Greece, pp. 622-625, 1999.

22. Matsatsinis N, Moraïtis P., Psomatakis V., Spanoudakis N, (99b). "Towards an Intelligent Decision Support System for Differentiated Agricultural Products

Development", in *5th International Conference of the Decision Sciences Institute*, *(DSI'99)*, Athens, Greece, pp. 1373-1376, 1999.

23. Matsatsinis N, Moraïtis P., Psomatakis V., Spanoudakis N, (99c). "A Multi-Agent System for Agricultural Applications", in *2nd European Conference of European Federation for Information Technology in Agriculture, EFITA'99*, Germany, 1999.

24. Matsatsinis N, Moraïtis P., Psomatakis V., Spanoudakis N, (99d). "An Intelligent Software Agent Framework for Decision Support Systems Development", in *European Symposium of Intelligence Techniques, ESIT'99*, Greece, 1999.

25. Boussetta S., El Fallah A., Haddad S., Moraïtis P, Taghelit M., (98). "Coordination d'Agents Rationnels par Planification Distribuée*", Revue d'Intelligence Artificielle (RIA), Special Issue on Distributed Artificial Intelligence and Multi-Agent Systems*, Vol.12, N°1, pp.73-101, Janvier 1998.

26. Pinson S., Louca J.A, Moraïtis P., (97). " A Distributed Decision Support System for Strategic Planning", *in Special Issue: Intelligent Agents as a Basis for Decision Support Systems, Decision Support Systems Journal,* Vol. 20, 1, pp 35-51, North-Holland Elsevier Science Publishers, May 1997.

27. Pinson, S., Moraïtis P., (96). "An Intelligent Distributed System for Strategic Decisions Making", in *Group Decision and Negotiation*, 6:77-108, Kluwer Academic Publishers, 1996.

28. Moraïtis P., A. Tsoukias, (96). "A Multicriteria Approach for Distributed Planning and Conflict Resolution for Multiagent Systems*", in Second International Conference on Multiagent Systems (ICMAS'96),* Kyoto, Japan, pp. 212-219, 1996.

29. Pinson S., Moraïtis P., Louca., J., (96). "A Cooperative Multi-Agent System for Strategic Decision Making", *in Applications of Artificial Intelligence: Expert Systems, Robots and Vision Systems, Fuzzy Logic and Neural Networks,* N.J. Mamede and C.P. Ferreira (Eds), Advanced Manufacturing Forum, Vol. 1, pp. 41-55, 1996 (also in *EPIA, Portuguese Conference on Artificial Intelligence*).

30. Boussetta S., Cohen D., Moraïtis P., (96). "Un Modèle d'Agent pour La Plannification Distribuée", in *PRC-IA Journée Systèmes Multi-Agents,* pp. 23-35, Toulouse, France, 1996

31. Balbo F., Moraïtis P., Pinson S., (96a). "Une méthode multi-critère pour l'allocation des tâches dans les systèmes multi-agents", in Intelligence Artificielle Distribuée et Systèmes Multi-Agents, Hermes (Eds.), pp. 85-100, 1996 (*4th Journées Francophones en Intelligence Artificielle Distribuée et Systèmes Multi-Agents, JFIADSMA*).

32. Balbo F., Moraïtis P., Pinson S., (96b). AMCA: méthode multicritère pour la coopération d'agents, Rapport Technique LAMSADE, No 138, Université Paris Dauphine, 1996.

33. Haddad S., Moraïtis P., Taghelit M., Boussetta S., Mazouzi H., Salah M., (96). *RAPID1.0: Réseau d'Agents à Planification Interactive Distribuée*, Manuel des Specifications, Partie 2, Rapport Technique N°7, PROJET CNET N° 904-15-4371-123, CNET, FRANCE TELECOM, 1996 (confidentiel).

34. Moraïtis P., Taghelit M., Boussetta S., Mazouzi H., Salah M., (96). *RAPID1.0: RAPID1.0: Réseau d'Agents à Planification Interactive Distribuée*, Manuel des Specifications, Partie 1, Rapport Technique N°6, PROJET CNET N° 904-15-4371-123, CNET, FRANCE TELECOM, 1996 (confidentiel).

35. Moraïtis P., Boussetta S., Mazouzi H., (96). Agent Interface. Rapport Technique N°5, PROJET CNET N° 904-15-4371-123, CNET, FRANCE TELECOM, 1996, (confidentiel).

36. Haddad S., Moraïtis P., Boussetta S, Cohen D., (95a). *Un Modèle Réactif pour Planification Distribuée*. Rapport Technique N°4, PROJET CNET N° 904-15-4371-123, CNET, FRANCE TELECOM, 1995, (confidentiel).

37. Haddad S., Moraïtis P., Boussetta S, Cohen D., (95b). *Un Modèle de Réseux de Petri Récursif pour la Répresentation de Plans.* Rapport Technique N°3, PROJET CNET N° 904-15-4371-123, CNET, FRANCE TELECOM, 1995, (confidentiel).

38. Moraïtis P., Boussetta S., Cohen D., (95a). *Une Architecture d'Agent Cognitif,* Partie2. Rapport Technique N°2, PROJET CNET N° 904-15-4371-123, CNET, FRANCE TELECOM, 1995, (confidentiel).

39. Moraïtis P., Boussetta S., Cohen D., (95b). *Une Architecture d'Agent Cognitif,* Partie1. Rapport Technique N°1, PROJET CNET N° 904-15-4371-123, CNET, FRANCE TELECOM, 1995, (confidentiel).

40. Moraïtis P., (95). Intelligence Artificielle Distribuée et Systèmes Multi-Agents : un état de l'art, Rapport Technique LAMSADE, No 89, Université Paris Dauphine, 1995.

41. Pinson S., Moraïtis P, (95). "Communication and Cooperation in a Distributed Decision Making System", *in Frontiers in Artificial Intelligence and Applications*, Aamodt & Komorowski (eds), vol 28, IOS Press, Amsterdam, Netherlands, Burke, USA, 1995, pp 441-447, (in *Fifth Scandinavian Conference on Artificial Intelligence, SCAI-95*, Trondheim, Norway, 1995)

42. Moraïtis P., Boussetta S., Cohen D., (95c). "*Système Multi-agent Cognitif pour la Surveillence de Réseaux",* in Journées "Gestion et Supervision de Réseaux de Télécommunications", FRANCE TELECOM-CNET, Lannion, 1995.

43. Moraïtis P., Pinson S., (94a). "Strategic Decision Making: an Intelligent Cooperative System", in *International Workshop on Knowledge-Based Systems and Strategic Management*, Abo, Finland, 1994.

44. Moraïtis P., Pinson S., (94b). "Distribution, Coopération, Cohérence dans un Univers de Prise de Décisions Stratégiques", in *2èmes Journées Francophones Intelligence Artificielle Distribuée et Systèmes Multi-Agents*, *(JFIADSMA),* Grenoble, 1994.

45. Moraïtis P., (94). "Paradigme Multi-Agent et Prise de Décision Distribuée", Ph.D. Thesis, University of Paris-Dauphine, France, 1994.

46. Pinson S., Moraïtis P., (93). "A Multi-Agent Approach for Strategic Decisions", in *Workshop on Artificial Economics, International Joint Conference on Artificial Intelligence, IJCAI-93*, pp. 85-96, Chambery, France, 1993.

47. Moraïtis P., (93). "Architecture Distribuée pour des Problèmes d'Aide à la Décision", in *PRC-IA Journée Systèmes Multi-Agents*, Montpellier, 1993.

References

1. Agent UML: http://www.auml.org/

2. Aknine, S., Pinson, S., Shakun, M.F. (02). "An Extended Multi-agent Negotiation Protocol", in *International Journal on Autonomous Agents and Multi-agent Systems*, Jennings, N.R., Sycara, K., (eds.), Kluwer, to appear.

3. Amgoud, L. and Parsons, S., (01). “Agent dialogues with conflicting preferences”, in *ATAL-01*, 2001.

4. Amgoud, L., Maudet, N., and Parsons, S., (00)."Modelling dialogues using argumentation", in *ICMAS-00*, 31-38, 2000.

5. Amgoud, L., Parsons, S. and Maudet, N., (00). “Arguments, dialogue and negotiation”, in *Proceedings of ECAI 2000*, W. Horn (ed.), Berlin, Germany, IOS Press, 338-342, 2000.

6. Bellifemine, F., Caire, G., Trucco, T. and Rimassa, G., (02). Jade Programmer’s Guide, JADE 2.5, http://sharon.cselt.it/projects/jade/, (2002)

7. Bellman R., (57). *Dynamic Programming, Princeton* University Press, Princeton, 1957.

8. Bondarenko, A., Dung, P. M., Kowalski, R. A. and Toni, F., (97). “An abstract, argumentation-theoretic framework for default reasoning”, *Artificial Intelligence,* 93(1-2), 63-101, 1997.

9. Brazier, F.M., Dunin-Keplicz, B.M., Jennings, N.R. and Treur, J., (97). “DESIRE: Modeling Multi-Agent Systems in a Compositional Formal Framework”, Huhns, M., Singh, M. (eds.), in *International Journal of*

Cooperative Information Systems. Special issue on Formal Methods in Cooperative Information Systems: Multi-Agent Systems, 1997.

10. Brewka, G., (01).”Dynamic argument systems: a formal model of argumentation process based on situation calculus”, in *Journal of Logic and Computation,* 11(2), 257-282, 2001.
11. Climaco, J. and Martins, E. (82). “A bicriterion shortest path algorithm”, in *European Journal of Operational Research*, 11:399-404, 1982.
12. Collis, J. and Ndumu, D., (99). *Zeus Technical Manual*. Intelligent Systems Research Group, BT Labs. British Telecommunications, 1999.
13. Cyert, R.M. and March, J.G., (63). *A Behavioral Theory of the Firm*. Englewood Cliffs, New York: Prentice Hall, 1963
14. desJardins, M.E., Durfee, E.H., Ortiz, C.L. and Wolverton, M.J., (99). “A Survey of Research in Distributed, Continual, Planning”, in *AI Magazine*, 20(4), 1999.
15. Dimopoulos, Y. and Kakas, A. C., (95). “Logic Programming without Negation as Failure”, in *Proc. ILPS-95*, 369-384, 1995.
16. Dung, P.M., (95). "On the acceptability of arguments and its fundamental role in non-monotonic reasoning, logic programming and n-person games", *Artificial Intelligence*, 77, 321-357, 1995.
17. Durfee, E.H., (01). “Distributed Problem Solving and Planning”, in *Multi-Agent Systems and Applications*, 9th ECCAI Advanced Course, ACAI 2001 and Agent Link’s 3rd European Agent Systems Summer School, EASSS 2001, LNCS 2086, 118-149, 2001.
18. Durfee, E.H., (99). “Distributed Problem Solving and Planning”, in *Multi-Agent Systems: A Modern Approach to Distributed Artificial Intelligence*, G. Weiss (ed.), MIT Press, 121-164, 1999.
19. El Fallah-Seghrouchni, A. and Haddad, S., (96). “A Recursive model for distributed planning”, in *Proc. ICMAS-96*, AAAI Press, 307-314, 1996.
20. Ephrati, E., Pollack, M.E. and Rosenschein J.S., (95). “A tractable heuristic that maximizes global utility through local plan combination”, in Proc. ICMAS-95, 94-101, 1995.
21. Erol, K., Hendler, J. and Nau, D.S., (94). *Semantics for hierarchical task network planning,* in Tech. Report CS TR-3239, UMIACS TR-64-31, ISR-TR-95-09, University of Maryland, 1994.
22. Faratin P., Sierra C. and Jennings N.R., (98). “Negotiation Decision Functions for Autonomous Agents”, in *Int. Journal of Robotics and Autonomous Systems* 24 (3-4) 159-182, 1998.

23. Faratin, P., Sierra, C., and Jennings, N. R., (00) "Using similarity criteria to make negotiation trade-offs", in *Proc. ICMAS-2000*, Boston, USA, 119-126, 2000.

24. Ferber, J. (99). Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence, Addison-Wesley, 1999.

25. *FIPA-OS*: A component-based toolkit enabling rapid development of FIPA compliant agents: http://fipa-os.sourceforge.net/

26. Firby R.J., (94). "Task networks for controlling continuous processes: issues in reactive planning", in *Proc. AIPS-94,* 49-54, 1994.

27. Gat E., (92). "Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots", in *Proc. AAAI-92*, 802-815, 1992.

28. Georgeff, M.P., and Ingrand F., (89). "Decision-Making in an Embedded Reasoning System" in *Proceedings IJCAI-89,* 972-978, 1989.

29. Giunchiglia, F., Mylopoulos, J. and Perini, A., (02). The Tropos Software Development Methodology: Processes, Models and Diagrams, in *AAMAS-02*, 2002.

30. Hansen P., (80). "Bi-criterion path problems", G. Fandel, T. Gal, (eds.), in *Multiple Criteria Decision Making: Theory and Applications*, LNEMS 177, Heidelberg: Springer-Verlag, 109-127, 1980.

31. Henig M., (94). "Efficient Interactive Methods for a Class of Multi-attribute Shortest Path Problems", *Management Science*, 40, 891 – 897, 1994.

32. Hitchcock, D., McBurney, P. and Parsons, S., (01). "A Framework for deliberation dialogues, Argumentation and its Applications", in *Proc. of the Fourth Biennial Conference of the Ontario Society for the Study of Argumentation,* 2001.

33. Huhns, M. and Singh, M. (Eds.), (98). *Readings in Agents*, Morgan Kaufmann Publishers, Inc., 1998.

34. Jacquet-Lagrèze E. and Siskos J., (82). "Assessing a set of additive utility functions for multicriteria decision making: the UTA method", in *European Journal of Operational Research,* 10: 151-164, 1982.

35. Jennings, N. R., Faratin, P., Lomuscio, A. R., Parsons, S., Sierra, C. and Wooldridge, M., (01). "Automated negotiation: prospects, methods and challenges", in *Int. J. of Group Decision and Negotiation* 10 (2), 2001.

36. Jennings, N.R., Sycara, K. and Wooldridge, M., (98). "A Roadmap of Agent Research and Development", in *Int. Journal of Autonomous Agents and Multi-Agent Systems,* 1 (1), 7-38, 1998.

37. Kabanza, F., (95). "Synchronizing multiagent plans using temporal logic specifications", in *Proc. ICMAS-95*, 217-224, 1995.

38. Kakas, A.C., Mancarella, P. and Dung, P.M., (94). “The Acceptability Semantics for Logic Programs”, in *Proc. ICLP-94,* 504-519, 1994.

39. Karacapilidis, N., and Papadias, D. (98). “A Computational Approach for Argumentative Discourse in Multi-Agent Decision Making Environments”, in AI Communications Journal 11(1), 21-33, 1998.

40. Knoblock, G.A. and Ambite, J.L, (97). “Agents for Information Gathering”, in Bradshaw, J.M. (ed.), *Software Agents*, 347-373, 1997.

41. Kraus, S., (97). “Negotiation and cooperation in multi-agent domains”, in *Artificial Intelligence,* 94, 79-97, 1997.

42. Kraus, S., Sycara, K. and Evenchik, A., (98)."Reaching agreements through argumentation: a logical model and implementation", in *Artificial Intelligence*, 104, 1-69, 1998.

43. Lander, S. and Lesser, V., (92). “Customizing distributed search among agents with heterogeneous knowledge”, In *Proc. First Int. Conf. On Information Knowledge Management,* 335-344, 1992.

44. Lansky, A.L., (90). “Localized search for controlling automated reasoning”, in Proc. Of the DARPA Workshop on Innovative Approaches to Planning, Scheduling and Control, 115-125, 1990.

45. Laurel, B. (97). “Interface Agents: Metaphors with Character”, in Bradshaw, J.M. (ed.), *Software Agents*, 67-77, 1997.

46. Lomuscio, A., Wooldridge, M. and Jennings N.R., (00). “A classification scheme for negotiation in electronic commerce”, in *Agent-Mediated Electronic Commerc,* A European AgentLink Perspective (eds. F. Dignum and C. Sierra), Springer Verlag, 19-33, 2000.

47. March, J.G. and Simon, H.A. (58). *Organizations*, New York: Wiley, 1958.

48. Martial, v., F., (92). *Coordinating Plans of Autonomous Agents*, Heidelberg, Springer-Verlag, 1992.

49. Maslow, A., (54). *Motivation and Personality*. Harper and Row, New York, 1954.

50. Matsatsinis, N.F. and Siskos, Y., (99). “MARKEX: An intelligent decision support system for product development decisions” in *European Journal of Operational Research*, Vol. 113, No. 2: 336-354, 1999.

51. Morignot, P. and Hayes-Roth, B., (95)."Adaptable motivational profiles for autonomous agents", in Knowledge Systems Laboratory, Report No. KSL 95-01, Dept of Computer Science, Stanford University, 1995.

52. Morignot, P. and Hayes-Roth, B., (96). "Motivated agents", in Knowledge Systems Laboratory, Report No. KSL 96-22, Dept of Computer Science, Stanford University, 1996.

53. Müller H.J., (96). “Negotiation Principles”, in *Foundations of Distributed Artificial Intelligence,* O´Hare and Jennings (Eds.), 211-229, 1996.

54. Nii, P.H. (86). Blackboards Systems: The Blackboard model of problem solving and the evolution of blackboard architectures”, in AI Magazine, 1986.

55. Occello, M., Baeijs, C., Demazeau, Y. and Koning, J-L., (02). MASK: An AEIO Toolbox to Develop Multi-Agent Systems. in Cuena et al. (eds), in *Knowledge Engineering and Agent Technology*, IOS Series on Frontiers in AI and Applications, Amsterdam, The Netherlands. 2002.

56. Panzarasa, P., Jennings, N.R. and Norman, T., (02)."Formalizing collaborative decision-making and practical reasoning in multi-agent systems", in *Journal of Logic and Computation,* 12 (1), 2002.

57. Parsons, S. and Jennings, N.R., (96). “Negotiation through argumentation - a preliminary report”, in *Proc. ICMAS-96*, 267-274, 1996.

58. Parsons, S., Sierra, C. and Jennings, N.R., (98). “Agents that reason and negotiate by arguing”, in *Logic and Computation*, 8 (3), 261-292, 1998.

59. Prakken, H. and Sartor, G., (96). “A dialectical model of assessing conflicting arguments in legal reasoning”, in *Artificial Intelligence and Law,* Vol. 4, 331-368, 1996.

60. Reed, C., (98). “Dialogues frames in agent communication”, in *Proc. ICMAS-98*, 246-253, 1998.

61. Russell, S. and Norvig P., (95). *Artificial Intelligence: A Modern Approach*. Prentice-Hall, NJ, 1995.

62. Saaty, T.L., (80). *The Analytic Hierarchy Process*. New York: McGraw-Hill, 1980.

63. Sabater, J., Sierra, C., Parsons, S. and Jennings, N. R., (02). "Engineering executable agents using multi-context systems", in *Journal of Logic and Computation*, 12, 2002.

64. Sadri F., Toni, F. and Torroni, P., (01). “Dialogues for negotiation: agent varieties and dialogue sequences”, in *Proc. ATAL-01*, 2001.

65. Sandholm T. and Lesser V. (95). “Issues in Automated Negotiation and Electronic Commerce: Extending the Contract Net Framework”, in *Proc. ICMAS-95*, San Fransisco, 328-335, 1995.

66. Schoppers M.J., (87). “Universal Plans for Reactive Robots in Unpredictable Environments”, in *Proc. IJCAI-87,* 1039-1046, 1987.

67. Sierra, C., Jennings, N.R., Noriega, P. and Parsons, S., (97)."A framework for argumentation-based negotiation", in *Proc. ATAL-97*, 167-182, 1997.

68. Simon, H.A., (75). *Administrative Behavior*, New York: Macmilan Company, 1975.

69. Siskos, J., and Yannacopoulos, D., (85). “UTASTAR: An ordinal regression method for building additive value functions”, in *Investiçao Operational*, Vol. 5, No. 1: 39-53, 1985.

70. Stentz A., (95). “The Focussed D* Algorithm for Real-Time Re-planning”, in *Proc. IJCAI-95*, 1652-1659, 1995.

71. Sycara, K., (89)."Argumentation: Planning other agents' plans", in *Proc. IJCAI-89*, 517-523, 1989.

72. Sycara, K., (90). “Persuasive argumentation in negotiation”, in *Theory and Decision*, 28:203-242, 1990.

73. Sycara, K., and Zeng, D., (96). “Coordination of Multiple Intelligent Software Agents”, in International Journal of Cooperative Information Systems, World Scientific Publishing Company.

74. Sycara, K., Paolucci, M., van Velsen, M. and Giampapa, J., (02). “The RETSINA MAS Infrastructure”. Accepted by the *Journal of Autonomous Agents and Multi-agent Systems (JAAMS),* to appear.

75. Vincke P., (92). *Multi-criteria Decision Aid*. New York: John Wiley, 1992.

76. Walton, D.N. and Krabbe, E.C.W., (95). *Commitment in dialogue: Basic Concepts of Interpersonal Reasoning*. State University of New York Press, NY, 1995.

77. Weiss, G., (Ed.), (99). Multi-Agent Systems: A Modern Approach to Distributed Artificial Intelligence, 1999.

78. Wilkins, D.E. and Myers, K.L., (95). “A common knowledge representation for plan generation and reactive execution”, in *Journal of Logic and Computation*, 5(6): 7311-761, 1995.

79. Witting, T. (Ed.), (92). *ARCHON: An Architecture for Multi-Agent Systems*, Ellis Horwood Series in AI, 1992.

80. Wood, M.F. and DeLoach, S.A., (00). “An Overview of the Multiagent Systems Engineering Methodology” in *AOSE-2000, The First International Workshop on Agent-Oriented Software Engineering*, Limerick, Ireland, 2000.

81. Wooldridge, M. and Jennings, N.R. (95). “Intelligent Agents: Theory and Practice”, *The Knowledge Engineering Review*, 10 (2), 115-152, 1995.

82. Wooldridge, M., (02). *Introduction to Multi-Agent Systems*, J. Wiley, 2002.

83. Wooldridge, M., Jennings, N.R. and Kinny, D., (00). “The Gaia Methodology for Agent-Oriented Analysis and Design”, in *Journal of Autonomous Agents and Multi-Agent Systems,* Vol. 3, No.3, 285-312, 2000.

84. Zlotkin, G. and Rosenschein, J.S. (91). Cooperation and Conflict Resolution via Negotiation Among Autonomous Agents in Noncooperative Domains, in