Vers la résolution de problèmes par émergence

Habilitation à diriger des recherches de l'Université Paul Sabatier
Spécialité : Informatique

Présentée par

Marie-Pierre Gleizes

Soutenue le 9 décembre 2004 devant le jury suivant :

Rapporteurs :	
Yves Demazeau	Chargé de Recherche CNRS – LEIBNIZ – Grenoble
Joël Colloc	Professeur, Université du Havre
Jean-Paul Haton	Professeur, Université de Nancy I
Examinateurs :	
Claudette Cayrol	Professeur, Université de Toulouse III, (Présidente)
François Charpillet	Directeur de Recherche, INRIA Lorraine, Nancy
Pierre Glize	Ingénieur CNRS – IRIT – Toulouse
Jean-Luc Soubie	Ingénieur INRIA – IRIT – Toulouse
Invité :	
Claude Chrisment	Professeur, Université de Toulouse III

Table des matières

Remerciements

Je suis très reconnaissante à Yves Demazeau pour l'attention qu'il a portée à mes travaux durant plusieurs années et pour avoir accepté de rapporter sur ce travail. Il est à l'origine de mon implication au niveau européen. J'apprécie ses efforts pour la communauté SMA française.

Joël Colloc a découvert tout récemment mes travaux scientifiques, ce qui lui a sans aucun doute demandé un travail conséquent pour le rapporter. Je le remercie vivement pour l'attention et l'accueil qu'il y a apporté.

A ses multiples activités, Jean-Paul Haton a spontanément accepté de rajouter celui du rapport sur mes travaux. J'en suis profondément touchée, d'autant que j'apprécie particulièrement ses qualités humaines et scientifiques.

Claudette Cayrol est une collègue d'enseignement dont je connais la rigueur et le sens des responsabilités. J'ai été très flattée qu'elle accepte d'être au jury et surtout de le présider.

Je remercie François Charpillet -malgré sa charge de travail- d'avoir accepté de participer à mon jury. Cela renforce les relations privilégiées que nos deux équipes entretiennent depuis de nombreuses années.

Merci à Jean-Luc Soubie d'avoir accompagné mes débuts dans le monde de la recherche en m'ayant encadrée durant mon DEA et ma thèse. Maintenant, c'est un ami dont les conseils sont précieux que ce soit dans le monde de la recherche ou dans le monde politique.

Si je soutiens mon Hdr, c'est un peu grâce à Claude Chrisment qui régulièrement est venu m'encourager et m'a poussé à rendre prioritaire ce travail et à aller jusqu'au bout. Merci pour ta ténacité, tes encouragements et ta droiture.

Avec Nathalie Aussenac nous avons simultanément entrepris la rédaction de nos habilitations. Elle n'est sûrement pas étrangère à la structure de ce mémoire.

Une habilitation reflète beaucoup le travail collectif d'une équipe. Un grand merci aux permanents encore en activité ou retraités : Valérie Camps, Bernard Carpuat, Jo Link-Pezet, Christine Régis, Sylvie Trouilhet.

Mais l’équipe c’est aussi des doctorants compétents et motivés, que j’ai plaisir à remercier pour leurs discussions stimulantes et combien animées : Davy capera, Jean-Pierre Georgé, Jean-Pierre Mano, Kévin Ottens et Gauthier Picard.

Une pensée particulière à Carole Bernon pour sa relecture assidue et pour sa coopération dans bien des travaux de recherche et d’enseignement. Merci pour ta compagnie dans beaucoup de missions et pour tous tes plans de voyages (scientifiques !) qui s’avèrent supers.

Un grand merci à André Machonin, pour ton professionnalisme,pour ta relecture, ta ténacité envers des codes récalcitrants pour ses heures de codage dans SYNERGIC, TELEMAC, ABROSE, FORSIC… qui ont occasionnées des échanges animés et ton amitié.

Merci à tous ceux qui nous rendent la vie plus facile au sein du laboratoire ou pour les enseignements et avec qui c’est vraiment bien de travailler Agathe Baritaud, Jean-Pierre Baritaud, Blandine Belcikowski, Evelyne Cassagne, Jean-Pierre Ceccatto, Jean-Claude Debelle, Martine Deperetti, Max Delacroix, Myriam Escobedo, Jean Frontin, Alain Monier, Chantal Morand, Christine Piquemal-Baluard, Maurice Rieux, Michèle Romens, Denise Roncier et Marie-Louise Sitbon et bien d’autres que ne pourrais citer par manque de place.

Un merci bien mérité à mes amis et collègues d’enseignement préférés Amal Sayah et Jean-Marie Rigaud. Vous avec qui j’ai vraiment appris ce qu’était le métier d’enseignant, vous avez la rigueur, la pédagogie et l’amour du travail bien fait.

Un supplément de merci à Gauthier Picard et à Jean-Pierre Georgé pour avoir partagé mon bureau ces dernières années, pour avoir répondu au téléphone et assuré le secrétariat, tâche qui ne vous incombait pas. Merci d’avoir supporté cela avec bonne humeur et humour.

Et enfin je voudrais remercier celui sans qui ce travail n’aurait pu être réalisé celui qui m’a épaulé dans mon travail de recherche durant toutes ces années et qui accompagne ma vie merci à toi Pierre. Merci aussi à mes trois enfants qui souhaitent avoir une maman beaucoup plus disponible. Ils vont me retrouver à temps plein les week-ends et les vacances c’est promis, merci pour votre patience. Je vous aime tous les quatre…

Introduction

Cette introduction a pour objectif d'expliciter la problématique qui a été au cœur de mes recherches, de définir le contexte scientifique et social dans lesquels mes travaux ont été menés et de faire la synthèse de mes contributions.

1. Problématique de recherche

L'approche largement admise et pratiquée pour la conception de systèmes artificiels place le concepteur comme un être omniscient du système et de son environnement. Or, les systèmes difficiles à concevoir sont peu, voire pas du tout, maîtrisables dans leur globalité par des êtres humains. On peut les qualifier de complexes relativement à la compréhension que nous en avons. Dans cette catégorie de systèmes figurent par exemple : la gestion de systèmes distribués, la coordination des secours dans des gestions de crise, la gestion d'agenda, la prévision… Mon objectif est de participer au développement de théories, méthodes et techniques – nécessairement nouvelles – pour concevoir et contrôler de tels systèmes. En effet, le contrôle du système doit être réalisé sans avoir accès à une connaissance complète même si certains éléments de cette connaissance peuvent être appréhendés. Les théories usuelles d'apprentissage œuvrent pour combler ce manque d'information au moment de la spécification du système. Mais, la plupart de ces techniques utilisent des connaissances a priori pour guider l'apprentissage et cela introduit un biais de représentation dont il faudrait se passer. Ma préoccupation scientifique est donc entièrement orientée vers la conception ascendante de tels systèmes, induisant un travail théorique sur l'émergence de fonctionnalités globales à partir de fonctionnalités locales. Cela m'a conduit à rechercher des principes organisateurs qui gouvernent les comportements collectifs émergents et à fournir des outils pour les mettre en œuvre.

L'objectif général de mes recherches est l'étude de la résolution de problèmes par un système complexe dont la fonction globale est le résultat du comportement des entités qui le composent, des interactions entre ces entités et des interactions entre le système et son environnement. Cet objectif s'est décliné tout au long de mes travaux, tout d'abord en mettant l'accent sur la notion de distribution. Puis, dans une deuxième période, le point

central a été la prise en compte d'environnements dynamiques et de systèmes ouverts. La problématique consistait à trouver des moyens d'adapter le système pour que la solution produite reste acceptable pour l'utilisateur. Actuellement, c'est la résolution de problèmes par émergence qui est au centre de mes travaux. La démarche consiste tout d'abord à clarifier cette notion pour les systèmes artificiels, puis à s'intéresser exclusivement aux agents du système en améliorant les règles locales de comportement pour garantir un comportement global cohérent et finalement à apporter une méthode de conception de ces systèmes.

Dans certains contextes dans lesquels le problème est résolu grâce à des connaissances pluridisciplinaires (comme la reconnaissance de la parole, le développement de produits…), la centralisation de la résolution ne permet plus de répondre aux besoins. C'est pourquoi, je me suis intéressé à la résolution distribuée de problèmes. Elle est caractérisée par le fait que plusieurs résolveurs de problèmes doivent participer à la construction d'une solution. Au cœur de cette problématique, la question est de coordonner les différents résolveurs pour avoir une résolution efficace. Classiquement, ces systèmes font du partage de tâches. Ceci consiste à effectuer, jusqu'à ce que le problème soit résolu ou qu'il y ait un échec, le cycle suivant [Gleizes, 1995]: le problème est décomposé en sous-problèmes, ils sont ensuite alloués aux résolveurs, puis les différents résultats retournés par les résolveurs sont synthétisés. Dans ce cadre, mes travaux se sont focalisés sur l'étude de la communication entre agents et du contrôle : comment le distribuer ? Comment continuer à contrôler la résolution ? Comment continuer à avoir un comportement correspondant à ce que le concepteur attend ?

La puissance croissante des ordinateurs a permis de traiter des applications de plus en plus complexes, la complexité provenant des nombreuses entités en interaction qui sont autonomes, hétérogènes, évolutives. De plus, avec l'essor du Web, les nombreuses possibilités de connexion entre les différents matériels ont impliqué une prise en compte d'environnements dynamiques et de systèmes ouverts et hétérogènes. Des théories formelles permettant de représenter et de raisonner sur le temps, l'espace et la dynamique d'un monde évolutif ont été développées. Mais, elles ne sont pas adaptées aux situations suivantes :

- L'environnement du système est dynamique, rendant inopérant l'énumération exhaustive des situations que le système rencontrera.
- Le système est ouvert et donc dynamique car il est constitué d'un nombre variable de composants.
- La tâche à réaliser par le système est tellement complexe qu'une conception parfaite ne peut être garantie par les concepteurs.
- La manière de réaliser la tâche assignée au système est difficile, voire impossible, à appréhender dans sa globalité par le concepteur, l'on peut aussi considérer que de telles spécifications sont inutiles, voire impossibles.

Que des systèmes artificiels puissent faire face à des situations réellement imprévues implique un axe de recherche aboutissant à une méthode de conception de systèmes différente de l'approche globale descendante traditionnelle fondée sur des modèles du monde. L'objectif est donc de concevoir des systèmes multi-agents adaptatifs, concevoir signifiant ici trouver un guide pour élaborer et contrôler le comportement des agents du système afin que le tout ait un comportement cohérent. Le défi est de trouver de nouvelles approches pour développer de nouveaux systèmes informatiques et pour mettre en œuvre ces nouvelles applications sans se heurter aux difficultés que l'on rencontre déjà telles que

la prise en compte d'une complexité croissante et la réalisation de systèmes robustes et autonomes. Parmi les systèmes naturels, physiques, sociologiques ayant les mêmes caractéristiques, l'étude des mécanismes et processus qui leur permettent de fonctionner montre qu'ils ont un facteur commun : la dimension émergente des phénomènes. Il paraît donc légitime d'étudier ce phénomène afin de pouvoir en comprendre le fonctionnement ou, au moins, être capable de l'utiliser pour la conception de systèmes artificiels. Ceci pourrait permettre de construire des systèmes plus complexes, robustes, autonomes et adaptatifs. Ainsi, des phénomènes désirables pourront survenir dans l'univers des systèmes artificiels. A contrario, si nous n'y prenons garde, ces phénomènes seront non seulement inattendus mais aussi indésirables. Pour y faire face, une orientation selon nous, serait que la communauté scientifique étudie et élabore de nouvelles théories portant sur l'émergence.

La problématique de résolution de problèmes par ces systèmes a été traitée dans mes travaux :

- au niveau théorique, par une réflexion sur la manière de construire de tels systèmes. Comment trouver les mécanismes qui font qu'un collectif a un comportement adéquat. Comment définir des modèles. Ce travail d'équipe a donné lieu à la définition de la théorie des AMAS.
- au niveau expérimental, par le développement d'applications. L'aspect expérimental a toujours été volontairement privilégié dans les travaux de l'équipe car une application valide les idées avancées, et influence le travail théorique. Vis-à-vis du monde industriel, les applications représentatives du domaine permettent de convaincre de la portée et de la puissance de ces nouvelles techniques. Ceci facilite ensuite le transfert de technologie comme celui que nous avons réalisé entre l'équipe et la société ARTAL Technologies concernant le logiciel de prévision des crues STAFF (« Software Tool for Adaptative Flood Forecast ») [Sontheimer, 2000]. Ces applications sont aussi utilisées lors de la formation des étudiants pour illustrer les concepts théoriques.
- au niveau outillage, en donnant aux concepteurs des outils informatiques réutilisables, génériques afin de les aider à développer des systèmes à fonctionnalité émergente.

Mon travail de recherche a naturellement suivi les progrès effectués dans la compréhension de la résolution de problèmes et des systèmes composés d'entités interagissant. Mon travail pour mener à bien mes objectifs, et ceux de mon équipe, peut être structuré en trois grandes périodes :

1- La première période a permis de comprendre l'articulation du local au global lors de la conception de systèmes par une approche descendante. Durant cette période, notre travail a porté sur l'étude de mécanismes d'échange entre les agents qui soient indépendants de l'application. Le problème central à résoudre était celui du contrôle, c'est pourquoi j'ai étudié la coopération.

2- La deuxième période a consisté à définir ce qu'était la coopération et à s'intéresser aux systèmes ouverts dont la fonction globale est difficilement appréhendable. La coopération est utilisée pour guider le comportement des agents du système au cours de la résolution et pour permettre aux agents au sein du système de trouver l'organisation adéquate. La résolution de problèmes influence la manière de concevoir les systèmes et

c'est la conception ascendante qui est alors préconisée. Nous nous sommes alors intéressés aux systèmes fonctionnellement adéquats et nous avons élaboré une théorie validée par de nombreuses expérimentations.

3- Au cours de la dernière période, nous avons étudié la technologie de coopération dans le cadre de la résolution de problèmes fondée sur l'auto-organisation coopérative des agents et nous avons proposé des outils d'aide à la conception.

2. Contexte des recherches : les systèmes multi-agents

Le monde des agents est en réelle effervescence au vu du nombre de chercheurs qui utilisent ce mot et au vu du nombre de workshops et de conférences qui lui sont dédiés. Un agent est défini comme une entité physique ou virtuelle, autonome, située dans un environnement, capable d'agir dans un environnement, de communiquer avec d'autres agents, de percevoir son environnement, de se reproduire (éventuellement) [Ferber, 1995]. De plus, un agent possède usuellement un objectif individuel (fonction de satisfaction), des ressources, une représentation partielle de l'environnement, des compétences et il offre ses services. Son comportement est fonction de ses observations, de ses connaissances, de ses croyances, de ses compétences, de ses interactions.

Très tôt une distinction s'est opérée entre les travaux centrés sur la notion d'agent et les travaux centrés sur la notion de système multi-agent.

Dans les systèmes à agents, un agent poursuit un but individuel et interagit nécessairement avec d'autres agents ou avec des ressources. C'est le cas, par exemple d'un agent assistant chargé d'élaborer un voyage pour un utilisateur en utilisant les ressources présentes sur le Web. La problématique est centrée sur la manière de concevoir un agent intelligent et se rapproche de l'Intelligence Artificielle comme cela est présenté dans le livre de Russel et Norvig [Russel, 1995].

Dans les systèmes multi-agents, le système, composé d'agents, a une fonction à réaliser ou une tâche à résoudre. La problématique est centrée sur le collectif et sur la manière d'obtenir un comportement intelligent du système. Par exemple, un système multi-agent peut avoir à résoudre un problème de gestion de réseau de communications. Il est bien évident que de fortes synergies existent entre les recherches réalisées sur les deux types de systèmes. C'est dans le domaine des systèmes multi-agents que j'ai mené mes travaux.

2.1. Les systèmes multi-agents

Les systèmes multi-agents développés actuellement peuvent être classés en trois catégories [Boissier, 2004]:

- les simulations dont l'objectif est la modélisation de phénomènes du monde réel, afin d'observer, de comprendre et d'expliquer leur comportement et leur évolution. Ce

sont par exemple, des applications de simulation de phénomènes sociaux, environnementaux, éthologiques…

- les applications dans lesquelles les agents jouent le rôle d'êtres humains. La notion d'agent simplifie la conception de ces systèmes et amène de nouvelles problématiques centrées utilisateur telles que la communication, la sécurité… Les systèmes de ventes aux enchères dans laquelle les agents jouent les rôles de commissaire priseur et d'acheteurs, représentent une classe d'applications de cette catégorie.
- la résolution de problèmes, telle qu'elle avait été définie en Intelligence Artificielle, étendue à un contexte distribué. Dans ce cadre, l'objectif est de mettre en œuvre un ensemble de techniques pour que des agents, pertinents pour la résolution d'une partie ou l'ensemble du problème, participent de manière efficace et cohérente à la résolution du problème global.

Mes travaux se situent plus particulièrement dans cette troisième catégorie. Pour la résolution de problèmes une approche consiste à coder les agents et les interactions en se basant sur un modèle organisationnel prédéfini, comme dans la méthode AGR de Gutknecht et Ferber [Gutknecht, 1998]. Une autre approche, utilisée notamment dans mes travaux, consiste à donner des règles de comportement locales aux agents et à observer la mise en place d'une organisation, organisation dite émergente. Toute la difficulté étant de définir ces règles locales.

Par conséquent, un système multi-agent est un macro-système composé d'agents autonomes qui interagissent dans un environnement commun pour réaliser une activité collective cohérente, bien qu'ils puissent chercher à atteindre des objectifs individuels parfois contradictoires [Gleizes, 2000a]. Le résultat de l'organisation des agents correspond au système multi-agent. Ses principales caractéristiques sont que le système est ouvert ou fermé, homogène ou hétérogène et autonome ou non. Un système est dit ouvert si en cours de fonctionnement des agents peuvent s'ajouter ou disparaître. Un système est dit homogène si tous ses agents ont la même architecture. Un système est autonome s'il peut décider de son comportement sans intervention d'un contrôle extérieur (notamment sans l'intervention du concepteur) [Parunak, 2001].

En tant que concepteur de systèmes, nous cherchons à faire des systèmes qui réalisent bien ce pour quoi ils ont été conçus. Ainsi, un système est en adéquation fonctionnelle avec son environnement si la fonction qu'il réalise est parfaitement appropriée à son objet : son comportement correspond à ce qui a été spécifié. Mais, l'adéquation fonctionnelle ne peut être hypothétiquement décidée que par un observateur extérieur au système et au monde dans lequel celui-ci est plongé. Cet observateur devant être, de surcroît, omniscient dans ce monde.

Les systèmes multi-agents ont donc toute leur place pour résoudre des problèmes complexes de manière distribuée dans des situations où le concepteur ne peut pas maîtriser totalement le système dans sa globalité. Cela peut être dû à la taille du programme, à la dynamique du système parce qu'il est ouvert ou à la dynamique de l'environnement. Un système devant évoluer dans un environnement dynamique peut très vite ne plus être adapté à cet environnement. Nos systèmes multi-agents doivent donc avoir la propriété de s'adapter. La notion d'adaptation peut être étudiée au niveau de l'agent, c'est le cas notamment des travaux de Zahia Guessoum [Guessoum, 2003] mais aussi au niveau du système, ce qui est le cas dans mes travaux. Il est vrai que pour certains chercheurs du

domaine, tout système multi-agent est adaptatif. Considérant que dans un système multi-agent, les agents sont autonomes, proactifs, sociaux… le comportement du système peut être qualifié d'adaptatif [Pfeifer, 2001]. C'est ce que j'appelle l'adaptation faible. Par exemple, dans une vente aux enchères, on peut voir de l'adaptation au niveau des tractations pour l'achat d'un produit mais l'organisation du système est définie au préalable et reste stable, dans ce système, on a un commissaire priseur et des acheteurs.

Je qualifie d'adaptation forte, un système qui modifie son comportement en réaction aux modifications de son environnement dans le but de continuer à être adéquat. Par exemple, un mécanisme constitué de pièces mécaniques qui change de forme physique pour réaliser une fonction différente.

Un système multi-agent adaptatif est donc défini comme un système multi-agent qui est capable de changer son comportement en cours de fonctionnement pour l'ajuster dans un environnement dynamique, soit pour réaliser la tâche pour laquelle il a été conçu, soit pour améliorer sa fonction ou ses performances. Il est caractérisé par le fait d'être plongé dans un environnement dynamique, de réaliser une tâche (fonction) et d'être composé d'agents en interaction [Bernon, 2002c]. La conception de ces systèmes m'a amenée à m'intéresser aux notions d'émergence et d'auto-organisation.

Les systèmes multi-agents auxquels je m'intéresse, sont utilisés pour des applications présentant une ou plusieurs des caractéristiques suivantes :

- Le problème à résoudre est complexe, dans le sens où l'espace de recherche est très grand. Par exemple, un problème de gestion d'agenda fait partie de ces types de problèmes.
- Le concepteur sait ce que doit faire le système, mais il ne connaît pas un algorithme permettant d'y arriver ; c'est le cas, par exemple, dans des systèmes de résolution de problèmes faisant intervenir plusieurs expertises.
- L'environnement dans lequel évolue le système est dynamique, il est alors très difficile pour le concepteur d'énumérer toutes les situations que le système rencontrera. Un système de gestion de crise, une équipe de football sont représentatifs de systèmes plongés dans des environnements dynamiques.
- Le système est ouvert. Une application de commerce électronique sur le Web gérant des clients et des fournisseurs de service est typique de ce genre de système.
- Le système est incomplètement spécifié. Le concepteur n'a pas les moyens de spécifier complètement son système.

2.2. L'émergence et l'auto-organisation

La notion d'émergence peut être définie de manière intuitive comme une propriété macroscopique d'un système qui ne peut pas être inférée à partir de son fonctionnement microscopique. Parce qu'un couplage structurel existe entre le système et son environnement, le comportement émergent du système constitue une réponse aux sollicitations et aux contraintes de l'environnement. Cette notion est étudiée depuis fort longtemps dans les disciplines telles que la philosophie, la physique, la thermodynamique, la cybernétique … Dans les années 1980, elle s'est répandue dans les domaines concernés par la conception de systèmes artificiels comme la vie artificielle, les automates cellulaires,

les classifieurs, les systèmes complexes adaptatifs tels que ceux développés au Santa Fe Institute et les systèmes multi-agents et a pris le nom de calcul émergent [Forrest, 1990], [Forrest, 1991] qui précise que c'est le calcul produit par une collection d'agents en interaction qui est émergent.

Pour les chercheurs, le challenge est de concevoir des systèmes de manière ascendante sous la forme d'un collectif d'agents en interaction entre eux et avec leur environnement et dont le comportement global est émergent. L'étude de théories de l'émergence devrait permettre de guider la conception de tels systèmes. Les quatre conditions de l'existence d'une théorie de l'émergence sont, selon nous :

1. Se situer dans le domaine de la théorie des systèmes.
2. Porter sur les parties du système.
3. Ne dépendre ni de la finalité du système ni de son environnement.
4. Etre indépendante du support matériel sur lequel elle pourrait s'incarner (biologique, technologique, …).

Dans nos travaux, nous avons établi une théorie de l'émergence qui repose sur un mécanisme d'auto-organisation.

L'auto-organisation fait référence à un processus au cours duquel la structure du système apparaît souvent sans une contrainte explicite qui provient de l'extérieur du système (www.calresco.org/links.htm). Ce qu'il est important de souligner dans cette définition, c'est qu'il peut y avoir un couplage entre le système et son environnement mais l'environnement ne doit pas explicitement contrôler le comportement du système.

La première conférence dédiée à l'auto-organisation date des années 1959, ce processus a d'abord été étudié dans le domaine de la cybernétique par Ashby (1952) et Von Foerster (1960), puis dans le domaine de la thermodynamique avec les travaux de Ilya Prigogine dans les années 1977. Actuellement, l'étude de l'auto-organisation comme mécanisme de conception de systèmes intéresse fortement la communauté informatique et notamment dans le domaine des systèmes multi-agents. Les mécanismes inspirés des systèmes naturels sont repris et adaptés comme la stigmergie (en utilisant des phéromones, par exemple), la spécialisation de rôles chez les rats ou le comportement coopératif. Mes travaux concernent particulièrement la mise en œuvre de l'auto-organisation par coopération.

2.3. Le contexte social

La résolution distribuée de problèmes est le thème de recherche que j'ai découvert grâce à Jean-Luc Soubie au cours de ma thèse intitulée « Spécification d'une architecture de système multi-expert ». J'ai ensuite poursuivi mes recherches sur ce thème après ma nomination en tant que maître de conférences en 1988 dans l'équipe SMAC (Systèmes Multi-Agents Coopératifs) dirigée par Pierre Glize. Cette équipe a donc largement participé et influencé les travaux que j'ai pu mener notamment grâce aux discussions avec les permanents, mais aussi avec les étudiants que j'ai encadrés au cours de stages d'IUT, d'IUP, de DEA ou de thèse. Pierre Glize a toujours souhaité que la manière de mener nos

recherches prenne toujours en compte l'applicabilité des théories ou concepts élaborés. C'est pourquoi nous avons mené de nombreuses expérimentations que ce soit dans le cadre de projets nationaux, européens ou de transferts de technologie vers les industriels de la région, notamment avec la société ARTAL Technologies.

Au cours de ces projets, j'ai appris, grâce à la collaboration avec Alain Léger de France Télécom et Hans Einsiedler de Deutsche Telekom, à mener des projets et à planifier le travail. Notamment, quand Alain Léger et moi-même avons remplacé temporairement le responsable du projet suite à sa mutation.

L'appartenance à un laboratoire de recherche important a influencé les projets que j'ai été amenée à initier et coordonner. Ainsi, une collaboration fructueuse s'est établie avec les collègues spécialistes du génie logiciel au sein du projet RNTL ADELFE et avec les personnels de TNI Valiosys, spécialistes d'UML et d'Opentool.

Le travail dans la communauté française au travers de groupes de travail en tant qu'animatrice et/ou participante m'a permis d'avoir des discussions fructueuses qui ont alimenté mes travaux. Notamment, les travaux sur l'interaction avec Claudette Sayettat, Caroline Wintergerst, Vincent Chevrier, Michel Riveill ; sur l'émergence avec Jean-Pierre Müller, Joël Quinqueton, Sylvie Pesty, Laurent Magnin et Pierre Glize ; sur l'auto-organisation au sein du groupe MARCIA du PRC-GDR avec Yves Demazeau, Vincent Chevrier, Jorge Anacleto, Suzanne Pinson, Laurent Magnin Pierre Glize, Jacques Ferber ; et sur les méthodes avec Philippe Mathieu, Jean-Paul Barthes, Zahia Guessoum, Pierre Deloor.

Depuis 2001, ma participation à Agentlink et à la FIPA me permet de confronter et de discuter de mes travaux avec des chercheurs européens ; notamment en ce qui concerne l'auto-organisation dans le cadre du Technical Forum Group « Self-organisation in MAS » et en ce qui concerne les méthodologies avec les membres du Technical Comittee Methodology de la FIPA et le groupe de travail associé à AgentLink MSEAS. Ce dernier est devenu le Technical Forum Group « Agent-Oriented Software Engineering ».

3. Contributions

L'approche adoptée pour la conception d'un système multi-agent est l'approche ascendante. En effet, appréhender à un niveau global toute la complexité d'une application est quasiment impossible dès lors que cette complexité atteint un certain degré. Par exemple, peut-on appréhender de manière globale le comportement des entités jouant un rôle dans la recherche d'information sur Internet ? Peut-on donner l'algorithme de gestion d'un réseau de machines informatiques ? A partir d'un parc d'une centaine de machines cela devient très difficile. Il en est de même dans des environnements dynamiques comme une application militaire, où des missions sont sujettes à des imprévus (projet SCALA) [Degirmenciyan-Cartault, 2004]. Pour ces raisons, la manière de concevoir les systèmes multi-agents a toujours été motivée par la recherche des comportements des agents et non du système. Pour cela, l'information sur la manière d'obtenir la fonction globale n'est pas utilisée au niveau des agents et le contrôle du comportement du système ne peut pas être centralisé.

En 1987 avec le système ASYMEX, c'est la manière de communiquer par envoi de messages qui était novatrice dans un système de résolution distribuée de problèmes, face au courant dominant qui utilisait des tableaux noirs comme moyen de communication. La communication directe s'avère prédominante dans les systèmes multi-agents actuels. Conscients du problème d'intercompréhension qu'il peut y avoir entre deux entités informatiques, nous l'avons résolu en utilisant des règles de réécriture. De nos jours, les travaux sur l'utilisation d'ontologies par les agents tentent de remédier à ce problème.

En 1993, la plate-forme de développement de Synergic, opérationnelle et validée par des applications, représente un de nos principaux résultats. A ce moment-là, le travail sur une plate-forme n'est pas encore une activité au centre des développements dans les systèmes multi-agents. Les systèmes développés sont plutôt ad hoc. L'originalité de Synergic réside dans le fait que les agents, en fonction de l'état de leur résolution, décident de manière autonome de communiquer avec d'autres agents. Cette décision de l'initialisation et du contenu de la communication n'est pas dictée par une entité extérieure à l'agent. Les agents de Synergic possèdent quatre caractéristiques principales : ils sont autonomes et poursuivent un objectif individuel, ils raisonnent avec leurs connaissances sociales et les communications sont intentionnelles et spontanées. Dans le contexte de cette plate-forme, j'ai étudié la problématique liée aux systèmes ouverts. L'outil développé pour faciliter l'ajout d'un module de connaissances dans le système ne permet pas de faire cet ajout en cours de fonctionnement. Par contre, les modules de connaissances peuvent être ajoutés au gré des besoins des concepteurs, sans avoir une vue globale préalable.

La coopération est étudiée pour concevoir des systèmes efficaces de résolution de problèmes. Le sens que nous lui accordons va au-delà de la notion de « travailler ensemble ». C'est plus l'attitude coopérative que peut avoir un agent que nous essayons d'expliciter. Nous avons montré que la mise en œuvre de comportements coopératifs dans les agents du système permettait d'améliorer les performances de la résolution. L'étude de la résolution distribuée de problèmes a donné lieu à une définition des systèmes multi-agents et d'un modèle d'agent dit coopératif. Ce modèle d'agent est composé de cinq modules : de compétences, de représentations (des autres, de lui-même et de l'environnement), d'un langage de communication, d'aptitudes et d'une attitude sociale coopérative. De plus, à l'issue de cette période, les principales caractéristiques des systèmes multi-agents que nous souhaitions étudier, ont été déterminées. Les travaux de cette période ont mis en évidence le rôle clé de l'interaction. En effet, celle-ci est porteuse du contrôle de la résolution dans ces systèmes et établit l'organisation du système.

Les interactions représentent une des problématiques prise en compte au sein du groupe de travail « interaction » de l'AFIA (1994 - 1997). De même dans le cadre du groupe MARCIA du PRC-GDR (1995 – 1997), les interactions étaient étudiées comme moteur de l'organisation. Comme dans beaucoup d'autres travaux de recherches en IAD-SMA, les interactions sont aussi devenues notre objet d'étude.

En 1999, nos résultats montrent que les systèmes coopératifs sont efficaces pour résoudre des problèmes et que la coopération représente une heuristique de résolution. L'attitude sociale d'un agent correspond à la notion classique de coopération étendue à la détection et au traitement de situations non coopératives. Notre principale contribution durant cette période est l'élaboration de la théorie des AMAS (Adaptive Multi-Agent Systems) qui est le résultat d'années de recherches et d'un travail collectif dans l'équipe

SMAC. Elle permet d'appréhender la conception de systèmes de manière ascendante, ce qui la distingue des méthodes classiques de conception généralement descendantes. Elle se rapproche de la manière de travailler des concepteurs de systèmes multi-agents réactifs. La différence réside dans le fait que cette démarche est utilisée indépendamment de la granularité des agents. Nous avons étudié la coopération comme moyen de résoudre des problèmes au sein du jeu des loups et de l'agneau et de l'application du Tileworld. La coopération comme moyen de résolution est aussi étudiée par d'autres chercheurs tels que Mataric [Mataric, 1994], Goldman [Goldman, 1994], Sen et Sekaran [Sekaran, 1995]…

Dans la même période, les projets sur la recherche d'information ou le commerce électronique nous ont fait prendre en compte la nécessité de réaliser des systèmes qui s'adaptent à un environnement dynamique pour tendre vers l'adéquation fonctionnelle. Un système multi-agent basé sur la théorie des AMAS a été conçu pour gérer les représentations d'agents. La manière d'apprendre sur les compétences d'autres agents est une technique complètement nouvelle par rapport aux techniques d'apprentissage existantes, comme l'apprentissage par renforcement pour ne citer que la plus utilisée dans le domaine des systèmes multi-agents.

Les systèmes naturels comme les décrivent Maturana et Varela [Maturana, 1994] nous ont beaucoup influencés. Le couplage du système avec son environnement contraint le système à s'adapter. La théorie des AMAS s'écarte donc des théories logiques classiques utilisées dans les systèmes multi-agents, c'est une nouvelle manière d'appréhender et de concevoir les systèmes dans lesquels la fonction globale émerge des interactions entre les agents. Décomposer un système en agents et donner des compétences aux agents (leur donner la capacité de résoudre des problèmes ou d'exécuter des tâches) est une activité qui est réalisée par tous les concepteurs de systèmes multi-agents de cette période. L'originalité de notre approche réside dans le fait que nos travaux précédents sur la coopération nous guidaient aussi pour définir le comportement des agents. En fait, l'attitude coopérative telle que nous la définissons contrôle, au niveau local, le comportement de l'agent. Durant cette période, l'objectif était de valider la théorie sur des applications, c'est pourquoi j'ai participé à la réalisation des applications suivantes : Tileworld, ANTS, ARCADIA, ABROSE, FORSIC.

En 2004, l'étude de l'émergence comme moyen de conception de systèmes de résolution de problèmes a permis d'enrichir la notion de coopération. La réalisation d'un environnement de programmation émergente a mis en évidence que, suivant les caractéristiques des applications, la notion de feed-back était plus ou moins simple à mettre en œuvre mais surtout que la conception d'un système de programmation émergente à l'aide de systèmes multi-agents adaptatifs était faisable.

En parallèle et en lien avec les travaux précédents, une méthode de conception des systèmes à fonctionnalité émergente a été conçue. L'originalité par rapport aux méthodes existantes est qu'elle guide l'utilisateur quant au type d'agent utilisé : les agents coopératifs, en cela c'est une méthode dédiée ou spécifique. Le travail sur la méthode a largement contribué à modéliser et formaliser la théorie au travers de la définition des stéréotypes et du méta-modèle (en cours de définition) [Bernon, 2004]. Dans ce cadre, je participe depuis 2001 au groupe de travail européen sur l'ingénierie du logiciel basée sur les agents au sein du réseau d'excellence AgentLinkII puis AgentLink III.

En résumé, mes principales contributions sont :

- au niveau théorique, l'étude des interactions, de la coopération, la théorie des AMAS, la définition du méta-modèle d'un système multi-agent adaptatif, l'étude de l'émergence en résolution de problèmes ainsi que l'étude du rôle de l'environnement et l'élaboration du processus de développement de ces systèmes.
- au niveau applications, la participation à des projets internes à l'équipe comme le Tileworld, le jeu du loup et des agneaux, ETTO, EPE, et à des projets nationaux ou européens comme ANTS, ARCADIA, ABROSE, FORSIC et SYNAMEC.
- au niveau outillage, avec la participation à la plate-forme Synergic et à l'atelier ADELFE.

4. Plan du mémoire

Ce mémoire présente, de manière chronologique, une synthèse des travaux de recherche entrepris et encadrés depuis ma thèse, en 1987, jusqu'à aujourd'hui. L'objet central de mes recherches a toujours été l'étude de la résolution de problèmes par un système composé d'éléments en interaction, qui se retrouve dans les trois grandes périodes décrites dans ce mémoire au cours des chapitres 1, 2 et 3.

Le chapitre 1 correspond à la période 1987-1993 au cours de laquelle je me suis intéressée à la résolution distribuée de problèmes au sein d'un système multi-expert et donc plus particulièrement, aux problèmes liés à la distribution des connaissances et du contrôle. Ces différentes phases sont illustrées l'une, avec une application en transfert de technologie et l'autre, avec une application en médecine. La mise en œuvre d'applications a orienté mes travaux vers la prise en compte de l'ouverture d'un système. Les problèmes liés au contrôle de la résolution vont nous amener à penser que le contrôle passe par les interactions et à étudier ce concept.

Le chapitre 2 est dédié à la période 1994-1999 pendant laquelle les problèmes scientifiques abordés sont relatifs à l'ouverture d'un système, à la résolution dans des environnements dynamiques et à la conception de systèmes adaptatifs. La résolution de problèmes entre des éléments en interaction est réalisée en donnant aux agents une attitude coopérative. La coopération comme moyen de résolution est illustrée dans l'application du Tileworld et avec un système composé de fourmis fourrageuses. Puis, la coopération comme moyen d'adaptation est illustrée sur des systèmes de recherche d'informations.

Le chapitre 3 retrace la période 2000-2004 qui a été consacrée à la résolution de problèmes par émergence. Tout d'abord, nous avons essayé de définir la notion d'émergence pour des systèmes artificiels. Ensuite, l'émergence a donné une méthode de résolution expérimentée en programmation émergente et en conception mécanique. Finalement, une méthode pour concevoir des systèmes de résolution par émergence a été développée et expérimentée sur la conception d'un emploi du temps.

Chacun des trois chapitres débute par une présentation des objectifs et des motivations de mes recherches. Puis, le contexte dans lequel ces recherches se sont effectuées et leur positionnement sont décrits. Ensuite, mes contributions sont données de manière synthétique et sont illustrées dans le cadre d'une application. La dernière partie de chaque chapitre est consacrée à exposer les enseignements, les apports des travaux présentés mais aussi à donner les questions auxquelles aucune réponse n'a été apportée et les futures problématiques à envisager.

La conclusion générale développe les pistes de recherche à mener dans les années à venir.

5. Bibliographie relative à l'introduction

[Bernon, 2002] Bernon C., Gleizes M-P., Picard G., Glize P., *«The Adelfe Methodology For an Intranet System Design»*, Fourth International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS-2002), 27-28 May 2002, Toronto (Ontario, Canada) at CAiSE'02.

[Bernon, 2004] Bernon C. Cossentino M., Gleizes MP., Turci P., Zambonelli F., *« A study of some Multi-Agent Meta-Models »*, The Fifth International Workshop on Agent-Oriented Software Engineering (AOSE-2004), Giorgini P., Mueller J.P., Odell J. Editors, New York, USA 19 July 2004

[Boissier, 2004] Boissier O., Gitton S., Glize P., *« Caractéristiques des systèmes et des applications »*, Observatoire Français des Techniques Avancées : Systèmes Multi-Agents Série ARAGO 29

[Degirmenciyan-Cartault, 2002] Degirmenciyan-Cartault I., Marc F., *« SCALA : une approche multi-agent pour la conception de systèmes complexes »*, Journées Francophones sur les Systèmes Multi-Agents, P. Mathieu et J.P. Müller éditeurs, Hermès, 2002

[Ferber, 1995] Ferber J., *« Les systèmes multi-agents »*, InterEditions

[Forrest, 1990] Forrest S., *« Emergent computation: self-organizing, collective, and cooperative phenomena in natural and artificial computing networks »*, Proceedings of the ninth annual CLNS conference, 1990.

[Forrest, 1991] Forrest S., *« Emergent computation : Self-organizing, Collective, and cooperative phenomena »*, Natural and Artificial Computing networks - Special issue of Physica D - MIT Press - North-Holland.

[Gleizes, 1995] Gleizes M-P., *« Recueil de transparents du cours Intelligence Artificielle Collective et Planification »*, DEA Représentation des Connaissances et Formalisation du Raisonnement de L'Université Paul Sabatier Toulouse III.

[Gleizes, 2000] Gleizes M-P., Georgé J-P., Glize P., *« A theory of complex adaptive systems based on co-operative self-organisation. Demonstration in electronic commerce »*,

Self-organisation in Multi-agent Systems (SOMAS), July 27-28, 2000, UK, A Workshop organised by the Emergent Computing Network

[Guessoum, 2003] Guessoum Z., *« Modèles et Architectures d'Agents et de Systèmes Multi-Agents Adaptatifs»,* HdR de l'Université Pierre et Marie Curie Décembre 2003

[Gutknecht, 1998] Gutknecht, O., Ferber, J., *« A meta-model for the analysis and design of organizations in Multi-Agent Systems »,* Editor, Proceedings of ICMAS'98, Demazeau, Y. Editor, Paris, France. IEEE Computer Society, 1998

[Mataric, 1994] Mataric Maja J., *« Interaction and Intelligent Behavior »,* PHD of Philosophy Massachussetts Institute of Technology May 1994

[Maturana, 1994] Maturana H. R., Varela F.J., *« L'arbre de la connaissance »,* Addison Wesley 1994

[Parunak, 2001] Parunak, H. Van D., Brueckner, S., *« Entropy and self-organization in multi-agent systems »,* Proceedings of Autonomous Agents'01, p. 124-130, 2001.

[Pfeifer, 2001] Pfeifer R., Scheier C., *«Understanding intelligence »,* The MIT Press,

[Russel, 1995] Russel S., Norvig P., *« Artificial Intelligence. A modern Approach »,* Prentice Hall Series in Artificial Intelligence

[Sekaran, 1995] Sekaran M., Sen S., *« To Help Or Not To Help »,* Seventeenth Annual Cognitive Sciences Conference, Pitsburg, Pennsylvannia, July 22-25 1995

[Sontheimer, 1999] Sontheimer T., *« Modèle adaptatif de prévision de crues par systèmes multi-agents auto-organisateurs »,* Rapport de stage Institut Universitaire Professionnalisé - Direction Régionale de l'Environnement Midi-Pyrénées Université paul Sabatier Toulouse 1999

Chapitre I. La résolution distribuée de problèmes

1. Objectifs - Contexte

Dans cette section je présente d'abord mes motivations et les problématiques abordées dans mes travaux de recherche dans la période 1987 – 1993. Le contexte des recherches dans cette même période est ensuite décrit ainsi qu'un état de l'art relatif à mes travaux.

1.1. Objectifs

Mes recherches durant ma thèse se situent dans la continuité du développement de systèmes experts, entrepris auparavant dans mon équipe d'accueil. Mon sujet de thèse a concerné la résolution distribuée de problèmes et la spécification d'une architecture de système multi-expert, appelée ASYMEX permettant de pallier les limites des systèmes experts. En effet, dès que la connaissance experte devient importante le maintien de la cohérence de la base de connaissance devient complexe, le temps de réponse de ces systèmes devient très long (il est à noter qu'un moteur d'inférences passe 90% de son temps à filtrer la base de connaissances). De plus, le champ d'application des systèmes experts évolue vers des domaines où la résolution de problèmes nécessite des connaissances pluridisciplinaires, tels que la reconnaissance de la parole [Erman, 1975] [Lesser, 1977], la gestion de planning dans MEDIA [Vailly, 1987], l'analyse de scènes dans SACSO [Debord, 1982], la gestion d'entreprise dans DECIDEX [Benchimol, 1986]… Les systèmes experts doivent donc intégrer la connaissance de plusieurs experts et de nouvelles architectures de systèmes doivent être conçues. L'objectif était donc de distribuer la connaissance experte au sein de plusieurs entités dans un système et de définir et mettre en œuvre le contrôle pour résoudre une tâche globale par coopération des différentes entités.

La mise en œuvre de l'architecture ASYMEX dans une application de transfert de technologie a permis de se rendre compte que la centralisation du contrôle avait des inconvénients dès lors qu'un module expert devait être ajouté. De plus, la centralisation du contrôle est un goulet d'étranglement au cours de la résolution. C'est pourquoi après ma thèse, mes travaux ont consisté à spécifier le concept d'autonomie des agents composant un système [Soubie, 1996]. Le point central des recherches dans l'équipe a alors concerné

le contrôle et sa distribution au sein des agents du système. Le contrôle dans un système multi-agent peut être défini comme : décider quel agent va agir à un moment donné dans la résolution. C'est ce qui permet de déterminer quel(s) agent(s) doit(vent) recevoir des informations et quelles informations lui (leur) communiquer pour que la résolution s'effectue, ceci amène à résoudre les problèmes suivants :

- Comment représenter la connaissance sur les capacités des agents ? Un agent, pour décider de faire appel à un autre agent, doit avoir une représentation des autres, appelée croyances ou accointances. Ces connaissances permettent aussi de décider si un agent peut être utile dans la résolution.
- Comment les agents vont-ils communiquer entre eux ? La manière la plus fréquente de solliciter un agent dans la résolution distribuée de problèmes consiste en l'envoi d'une requête par un agent demandeur à un agent récepteur. L'agent récepteur de la requête peut refuser de répondre à cette requête.

En 1989, au terme de ces travaux, le besoin d'avoir une architecture réutilisable était une contrainte forte pour la conception d'une plate-forme de développement de systèmes composés de plusieurs entités. La modularité exigée a naturellement conduit nos travaux vers une décentralisation du contrôle, car la mise à jour de cette connaissance dans un module centralisé nuisait à la modularité et à la facilité de mise à jour. En effet le superviseur représente à l'exécution un goulet d'étranglement et son unicité ne garantit pas la robustesse du système. Le contrôle au sein de ces systèmes va devenir un des axes prépondérant de recherche dans l'équipe, car des modèles nouveaux et d'autres manières de concevoir ces systèmes sont à concevoir. Un des objectifs était donc de donner des outils aux concepteurs pour les aider au cours du développement pour la construction de la représentation du savoir-faire d'un agent par un autre agent mais aussi pour faciliter l'implémentation du contrôle (de la connaissance de gestion de ces modules dans la résolution), et pour rendre ces systèmes plus robustes. Mes recherches vont donc s'orienter vers la représentation du savoir-faire : Comment représenter les connaissances qu'un agent a sur un autre agent ? Comment aider le concepteur à construire cette connaissance ? Peut-on l'automatiser ?

Dans ce cadre nous avons conçu la plate-forme de développement de systèmes multi-agents Synergic. Au cours de la spécification et la conception de Synergic, je me suis plus particulièrement intéressée à l'ajout de nouveaux agents dans le système et quelles en étaient les conséquences. La distribution du contrôle a impliqué une étude de la gestion des accointances par un agent et de la communication entre eux. Ces problématiques ont fait l'objet du DEA et de la thèse de Sylvie Trouilhet qui conduit au phénomène émergent.

1.2. Contexte

La période 1987-1993 correspond au début des travaux dans un nouveau champ de recherche baptisé en 1988 : « Intelligence Artificielle Distribuée et Systèmes Multi-Agents » [Bond, 1988], [Haton, 1989] [Erceau, 1991], [Avouris, 1992]. L'idée générale est de décomposer un système dit complexe en entités élémentaires et les faire coopérer pour résoudre une tâche complexe. Huhns définit l'IAD comme la solution collaborative à des problèmes globaux par un groupe d'entités distribuées Les agents peuvent être des éléments d'exécution simples ou bien des entités complexes exhibant un comportement

rationnel. La résolution de problèmes est collaborative car de l'information partagée mutuelle est nécessaire pour produire une solution. Le groupe d'agents est décentralisé logiquement par le contrôle et les données et quelquefois physiquement [Huhns, 1987].

Les problématiques de recherche appartenant au domaine de l'IAD-SMA sont selon Bond [Bond, 1988]: le parallélisme, les systèmes multi-agents et la résolution distribuée de problèmes.

Le parallélisme. Le parallélisme est un moyen d'accélérer le temps de traitement des logiciels classiques de l'IA tels que les algorithmes de parcours de graphes ou les moteurs d'inférences dans leur phase de filtrage des bases de connaissances.

Les systèmes multi-agents. La recherche dans les systèmes multi-agents consiste à étudier le comportement d'une collection d'agents autonomes dont le but est de résoudre un problème donné. Gasser et Huhns [Gasser, 1989] les définissent comme des réseaux faiblement couplés de résolveurs de problèmes qui travaillent ensemble pour réaliser une tâche globale qui va au-delà de leurs capacités individuelles.

La résolution distribuée de problèmes. La résolution distribuée de problèmes consiste à savoir comment la résolution d'un problème peut être divisée et distribuée entre un certain nombre de modules qui coopèrent en distribuant et partageant des connaissances sur le problème et ses solutions. Toutes les stratégies d'interactions font partie intégrante du système. [O'Hare 1996].

Les premiers travaux qui lancèrent l'IAD-SMA sont les travaux sur les acteurs d'Hewitt [Agha, 1985], le modèle des Beings de Lenat [Lenat, 1975] et les architectures basées sur le tableau noir avec le système Hearsay II (1971 -1976) [Erman, 1975] dans lequel le contrôle est un programme centralisé, Crysalis (1976-1983) dans lequel le contrôle est hiérarchisé [Terry, 1988]. Ces travaux sont caractérisés par la distribution des connaissances et des traitements mais le contrôle est en général centralisé. Les architectures sont très ad hoc pour une application donnée et ne sont pas réutilisables et ces systèmes ne sont pas ouverts. C'est dans ce cadre, que j'ai développé ASYMEX, dont l'architecture se différentie des architectures dominantes basées sur le tableau noir parce que les agents communiquent par envoi de messages et ne partagent pas une structure de données commune.

A partir des années 1990, une communauté scientifique s'intéressant aux problèmes spécifiques de l'IAD et des SMA se forme à partir des premiers workshops aux Etats-Unis de 1980 à 1988 sponsorisés par AAAI[1], [Huhns, 1987], [Gasser, 1989] et en Europe avec le Workshop MAMAAW[2] de 1989 à 1996 (interruption d'un an en 1995) dont Yves Demazeau est l'un des fondateurs [Demazeau, 1990], [Demazeau, 1991], [Werner, 1992]. En 1992, la communauté scientifique française se réunit au sein d'un groupe de travail fondé par Jean Erceau et les premières Journées Francophones d'IAD-SMA ont lieu à Toulouse en 1993, l'équipe SMAC est chargée de leur organisation. Le défi de cette période est de distribuer la connaissance mais aussi le contrôle au sein des agents [Minsky, 1988]. Les systèmes qui sont l'objet de l'étude à ce moment-là sont des systèmes d'aide à la décision dont les connaissances nécessaires à la résolution sont issues de plusieurs domaines d'expertises. Ces systèmes sont aussi appelés systèmes multi-experts [Gleizes, 1990]. Cette deuxième génération de systèmes s'est beaucoup intéressée à la

[1] American Association for Artificial Intelligence
[2] Modelling Autonomous Agents in a Multi-Agent World

distribution du contrôle, à la modélisation des agents en proposant de nombreuses architectures agents et à la réutilisation des architectures. Beaucoup de travaux de cette période, concernant la mise en œuvre du contrôle sont les systèmes dont la communication entre les agents est basée sur la notion de tableau noir [Engelmore, 1988] tels que : ATOME (1986 – 1989) [Laasri, 1987], [Laasri, 1989] au LORIA où le contrôle est réalisé de manière hiérarchique avec les niveaux stratégie, tâche et spécialiste, BB1 [Hayes-Roth, 1985], [Hayes-Roth, 1986] où le problème du contrôle est résolu par un système de tableau noir, DVMT[3] (1981-1991) qui utilise plusieurs systèmes basés sur le tableau noir. Mais aussi des systèmes dans lesquels la communication est réalisée par envoi de messages tels que : le modèle de coordination du réseau de contrats de Davis et Smith [Smith 1981], [Davis 1983] et la plate-forme MACE[4] (1985-1990) [Gasser, 1987] qui fournit un langage pour la conception d'agents.

Durant cette période deux écoles se côtoient au sein de ce champ de recherche, l'école cognitive d'une part et l'école réactive d'autre part. L'école cognitive vient de l'Intelligence Artificielle et s'inscrit dans une volonté de faire coopérer des systèmes experts classiques comme dans le système ARCHON (Architecture for Cooperating Heterogeneous Online Systems) (1989 – 1994) [Wittig, 1992] [Burg, 1994], un projet Européen phare. Ces systèmes dits cognitifs sont caractérisés par un faible nombre d'agents de granularité forte c'est-à-dire des agents qui ont des capacités de raisonnement proche des systèmes experts. Castelfranchi dans ses travaux s'inspire fortement des sociétés humaines et propose une théorie basée sur la dépendance entre les agents [Conte, 1990], [Castelfranchi, 1992]. En 1991, Rao et Georgeff utilisent une architecture de type BDI[5] définie par Bratman en 1987 [Bratman, 1987], pour la conception des agents et l'appliquent au sein du système PRS[6] [Georgeff, 1987]. La problématique principale est de concevoir des modèles pour faire de la résolution distribuée de problèmes dont les questions essentielles concernent les communications entre les entités du système et la mise en oeuvre du contrôle dans ces systèmes. Nos travaux sur Synergic sont réalisés dans cette problématique, ils se caractérisent par le fait que nous voulions une plate-forme de développement réutilisable.

L'école réactive est issue de la vie artificielle et de la biologie. Les systèmes réactifs sont constitués d'un grand nombre d'agents « non intelligents ». Un agent réactif est caractérisé par le fait qu'il a un comportement câblé de type « réflexe », pas de mémoire, pas de représentation explicite de son environnement et qu'il communique en général via l'environnement. Les travaux les plus importants pour la conception de systèmes réactifs sont ceux de Brooks qui préconise une nouvelle voie pour la conception de systèmes intelligents avec des architectures telles que l'architecture de subsumption [Brooks, 1986]. Le système d'éco-résolution [Bura, 1991] développé dans l'équipe de Jacques Ferber fait partie de cette mouvance.

Les grandes questions que les chercheurs ont à résoudre sont [Gasser, 1991]:

- Comment utiliser plusieurs processeurs pour résoudre un problème ?
- Comment décomposer une tâche en sous-tâches ?

[3] Distributed Vehicle Monitoring Testbed
[4] Multi-Agent Computing Environment
[5] Belief Desire Intention
[6] Procedural Reasoning System

- Comment allouer les sous-tâches aux différents agents du système ?
- Comment traiter les interactions et la communication ? Quels langages et quels protocoles utiliser? Que communiquer ? Quand communiquer ?
- Comment modéliser les autres agents au sein d'un agent ?
- Comment coordonner les agents ?
- Comment assurer que le comportement global est cohérent ?
- Quels modèles, quels outils sont proposés aux développeurs ?

C'est dans ce contexte, où la notion de distribution est au cœur de la problématique multi-agent, que mes travaux s'inscrivent Tout d'abord, la distribution des connaissances est abordée dans le cadre du système ASYMEX et de l'application au transfert de technologie. Puis, la distribution du contrôle est prise en compte dans l'environnement SYNERGIC et l'application en télémédecine. Cette période se termine par des recherches sur la conception de systèmes ouverts et sur les interactions entre agents.

2. La distribution des connaissances du domaine

Le modèle ASYMEX développé pour la conception de systèmes multi-experts, permet la coopération de connaissances expertes réparties dans des modules de connaissances. Cette distribution de la connaissance soulève les problèmes suivants : Quelle connaissance grouper dans un module ? Comment mettre en œuvre le contrôle dans un tel système ? Comment effectuer la communication entre les modules ?

2.1. Les différents types de connaissances

Les types de connaissances répertoriés pour la résolution de problèmes dans des domaines d'application caractérisés par l'intervention de plusieurs domaines d'expertises sont : la connaissance monodisciplinaire d'un expert et la connaissance sur le processus de résolution de la tâche globale. Cette dernière comporte trois types de connaissances : la connaissance interdisciplinaire qui permet de décomposer la tâche en sous-tâches et d'intégrer les résultats de ces sous-tâches, la connaissance sur les capacités de résolution des modules monodisciplinaires qui permet de faire appel à eux, ainsi que la connaissance sur l'environnement du système pour interagir avec l'utilisateur et/ou les modules du système.

Les connaissances monodisciplinaires ont chacune été regroupées dans un module de connaissances par domaine d'expertise pour réduire la complexité de la coordination et le nombre d'interactions entre les modules et donc le temps de communication.

La connaissance interdisciplinaire n'est pas distribuée dans chacun des modules experts mais a été regroupée dans un module appelé superviseur. En effet, la distribution de cette connaissance aurait engendré des difficultés pour maintenir la cohérence globale

de la résolution, des coûts de communication importants pour la recherche du bon module ou pour l'élaboration d'accords mutuels pour un contrat de travail.

2.2. La connaissance sur le contrôle

Le contrôle de la résolution est centralisé dans le superviseur. Il se décompose en trois parties :

- la connaissance interdisciplinaire liée au domaine : la base de connaissance interdisciplinaire
- la connaissance sur les capacités de résolution, les conditions d'activation des modules monodisciplinaires : la base de savoir-faire ou de métaconnaissances
- la connaissance sur le contrôle du processus et la résolution : le programme allocateur qui est intégré dans le moteur d'inférences. Il permet d'appeler la base de savoir-faire et les modules de connaissances des divers domaines d'expertise.

2.3. La communication entre modules

Dans le contexte des recherches, c'est le modèle du tableau noir qui est largement plus répandu pour faire communiquer des modules de connaissances [Engelmore, 1988]. Dans ASYMEX, nous considérons qu'un module de connaissances doit posséder son espace propre, sa base de faits. La communication s'effectue par envoi de messages. Ce choix a été fait car la modularité de l'architecture est ainsi conservée. En effet, les modules de connaissances pour se comprendre peuvent utiliser le même vocabulaire ou non. Dans le cas où le vocabulaire est différent, les experts doivent écrire des règles de réécriture qui permettent de mettre en relation deux termes qui signifient la même chose. D'un point de vue plus technique, le coût des communications est minimisé car les volumes et la fréquence des échanges sont très réduits

2.4. Application à SATIN

Le modèle d'architecture ASYMEX pour le traitement et le contrôle des connaissances réparties a été expérimenté sur une application de système d'aide à la décision pour effectuer un transfert de technologie : le système SATIN (Système d'Aide au Tranfert et à l'Innovation) [Cohen, 1986b]. Au début des années 80, le processus de transfert technologique et de l'innovation était un sujet de questionnement important car il est supposé être une des sources du redémarrage économique [Cohen, 1986a]. Il est un fait que l'ensemble des partenaires socio-économiques impliqués dans ce processus avait des difficultés structurelles à coopérer. Pour réduire ces dysfonctionnements et considérant ce domaine comme scientifiquement pertinent en intelligence artificielle, nous avons conçu un logiciel prototype d'aide à la décision, SATIN.

SATIN facilite le transfert technologique et l'innovation par l'intégration et la coopération d'expertises juridique, économique, technique, financière et documentaire relevant du transfert de technologie [Cohen 1987]. Ces modules sont des systèmes à règles de production, mais la coopération entre les différentes expertises est exprimée sous forme de connaissances symboliques et déclaratives sur les bases de connaissances, c'est-à-dire sur les compétences et les environnements de résolution des bases de connaissances du système. La base de connaissances interdisciplinaire est basée sur la méthode de l'analyse de la valeur associée au développement de produit [Chevallier, 1988] et a été réalisée en collaboration avec Jean Chevallier (AFAV, CNES), expert en analyse de la valeur. La base de savoir-faire comporte la connaissance sur le savoir-faire des différentes bases de connaissances du système [Nitas, 1986]. Cette première étude a été conçue comme un processus automatisé de coopération entre différentes compétences, car c'est le système qui décide lui-même de leur enchaînement en raisonnant sur les savoir-faire.

2.5. Contributions

L'architecture a donc été conçue selon les principes suivants :

- la connaissance interdisciplinaire gère la coopération,
- l'organisation des connaissances facilite les modifications et la mise en œuvre du contrôle,
- le temps de résolution des modules est supérieur au temps dévolu à leur communication,
- les modules de connaissances sont autonomes.
- La distribution des connaissances a les avantages suivants :
- la duplication des compétences dans plusieurs agents est inutile,
- le maintien de la cohérence est facilité lorsque la compétence d'un agent est modifiée,
- lors du développement d'un nouveau module, le binôme expert-cogniticien doit seulement spécifier le nouveau savoir-faire sans se préoccuper de l'existence d'autres agents,
- l'analyse des connaissances du module de savoir-faire permettait d'analyser aisément le processus d'interactions entre agents.

Ce travail sur la distribution des connaissances au sein d'un système a apporté une solution en terme d'architecture modulaire pour la conception des modules de connaissances. Ces modules ont leur propre mémoire et ils échangent des messages pour communiquer. Les systèmes réalisés à cette période, ont comme hypothèse implicite de conception que les agents se comprennent. Conscients du problème d'intercompréhension qui peut y avoir entre deux entités informatiques, nous l'avons abordé en utilisant des règles de réécriture. De nos jours, les travaux sur l'utilisation d'ontologie par les agents tentent de remédier à ce problème. Cette manière de communiquer s'avère actuellement prédominante dans les systèmes multi-agents actuels. Ce qui était loin d'être le cas dans les années 1980.

3. La distribution du contrôle dans l'environnement de développement de systèmes multi-agents : Synergic

Synergic [Gleizes, 1992] est un environnement de développement de systèmes multi-agents (Figure 1). La connaissance relative au domaine traité ainsi que la connaissance du contrôle sont distribuées. En effet, un agent ne doit pas être capable de résoudre seul tous les problèmes soumis au système. Une distribution pertinente de la connaissance de contrôle permet à un agent en fonction de l'état courant de sa résolution, d'interagir directement avec un autre agent en lui communiquant les informations nécessaires et suffisantes, et d'accroître l'efficacité et la sûreté de fonctionnement du système.

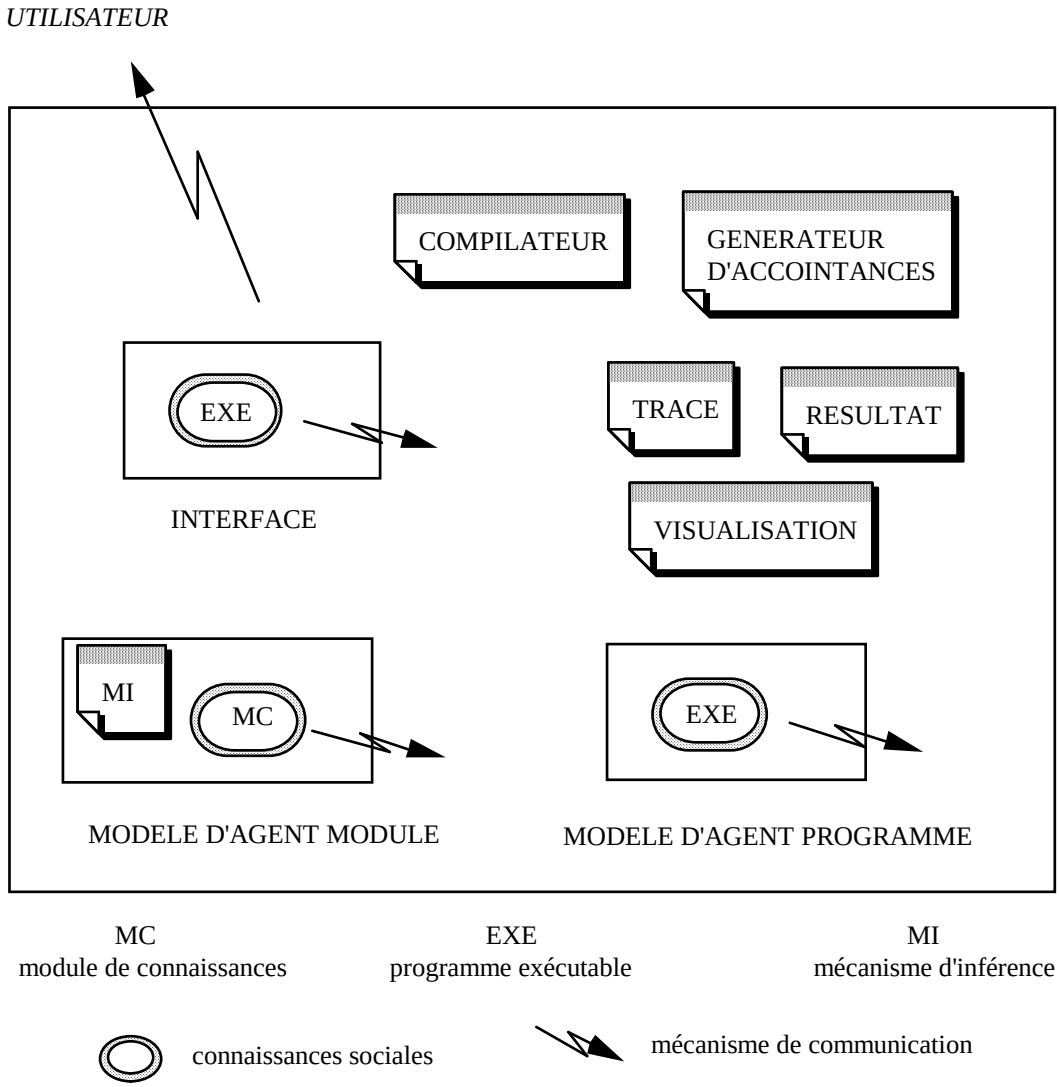


Figure 1 - Architecture de Synergic

3.1. Les agents

La connaissance du domaine nécessaire à la résolution est distribuée dans les agents du système avec les critères définis dans les travaux précédents sur ASYMEX. Les agents sont en fait déterminés par l'application traitée. En général, les agents les plus cognitifs pris en compte par Synergic, correspondent au champ d'investigation d'un expert humain c'est-à-dire à un sous-ensemble pointu d'un domaine d'expertise. La très forte interdépendance des connaissances d'un domaine justifie aussi leur agrégation pour limiter le nombre de communications entre les agents. Les agents plus réactifs sont des logiciels classiques ou des mécanismes. Tous ces agents sont décrits dans le système car ils interviennent à un certain moment pour résoudre des tâches spécifiques qui correspondent à leurs compétences. Ils sont déterminés par le demandeur de l'application avec l'aide du cogniticien.

Les agents d'une même application peuvent être conçus soit avec les techniques de Synergic, soit réalisés de manière totalement indépendante de la plate-forme. Les agents qui utilisent les techniques de raisonnement disponibles dans Synergic sont des modules de connaissances dont la forme externe de la connaissance correspond à des règles de productions (Figure 2). Ils utilisent le moteur d'inférences MIME4 [Nitas, 1987] fourni avec la plate-forme.

```
si    le diagnostic de la péritonite = probable
et    la température du patient < 37.8
alors       le diagnostic de la péritonite = impossible
```

Figure 2 - Un exemple de règle de production

Les autres agents sont du type boîte noire. La représentation des problèmes ainsi que les techniques utilisées pour les résoudre ne sont pas connues de Synergic. Dans cette catégorie on trouve : des capteurs, des programmes mathématiques, des programmes d'affichage d'images...

Cycle de vie d'un agent

Pour garantir l'autonomie cognitive de chaque agent, un processus informatique leur est assigné. Mais l'activité étant fortement liée à la communication (ce qu'ils reçoivent et ce qu'ils envoient), ils peuvent avoir des phases d'inactivité plus ou moins brèves. Chaque agent a le même cycle de vie, composé des étapes suivantes :

tantque l'agent est en vie faire
  début
  attendre des informations;
  traiter ces informations;
  construire d'éventuels messages pour d'autres agents
fin tantque;

Durant ce cycle de vie, un agent passe par trois états, représentés dans le schéma suivant :

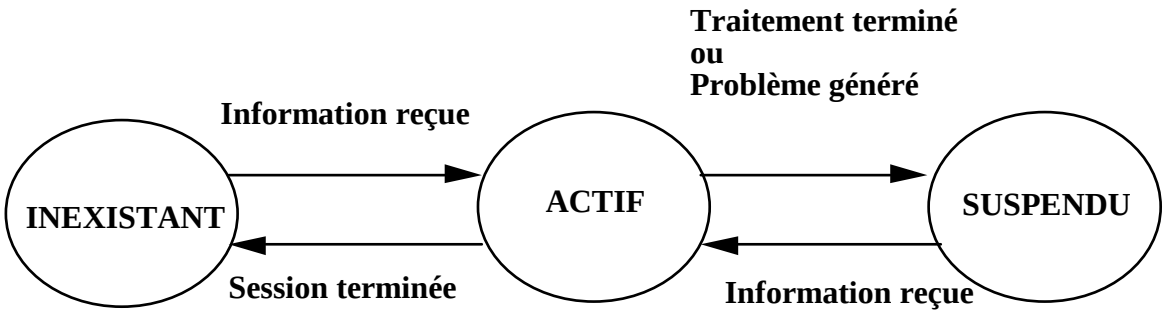


Figure 3 - Les états d'un agent

A l'initialisation d'une session, tous les agents sont dans un état passif appelé INEXISTANT. C'est une réception de message qui va les activer c'est-à-dire passer d'un état INEXISTANT à un état ACTIF. Un agent ACTIF va effectuer un traitement fonction de ses compétences et des informations qu'il a reçues. A la fin de son traitement, il va se mettre en attente d'informations, son état est dit SUSPENDU. Dans ce cas :

- soit il est SUSPENDU parce qu'il a terminé. Il communique éventuellement des informations à d'autres agents avant d'être inactif. C'est une réception de message qui aura pour effet de le réveiller et de le mettre dans l'état ACTIF.
- soit il est SUSPENDU parce qu'il est bloqué dans sa session. Il a engendré des tâches qu'il a communiquées à d'autres agents et se met en attente des réponses.

Traitement des informations : le moteur d'inférences

Le traitement des connaissances au sein d'un agent est effectué par le moteur d'inférences MIME4 qui travaille indistinctement en chaînage avant, chaînage arrière ou chaînage mixte. Il s'exécute sur un module de connaissances compilé, composé de règles de production qui sont ordonnées selon plusieurs niveaux; un niveau contient des données synthétiques des niveaux inférieurs. Le plus haut niveau détient des connaissances stratégiques alors que le niveau le plus bas renferme des connaissances de base. C'est le moteur qui gère les changements de niveau.

3.2. La représentation des connaissances sociales

Le contrôle de la résolution permet à un agent d'intervenir dans la résolution. Dans Synergic un agent, pour faire appel à un autre agent au cours de la résolution, possède une représentation des connaissances sociales c'est-à-dire de ce qu'un autre agent est capable de faire. Cette connaissance appelée métaconnaissance dans le système ASYMEX est nommée accointance dans le domaine des systèmes multi-agents. Ce terme a été emprunté à la terminologie du langage d'acteur [Agha, 85]. Les accointances représentent le fait qu'un agent a la connaissance des autres agents de la société. Un agent doit avoir la

capacité de représenter cette connaissance c'est-à-dire de la mémoriser et de raisonner sur elle.

Dans Synergic cette connaissance est distribuée dans les agents : chaque agent possède un certain nombre d'accointances sur les agents de son voisinage. Le concepteur du système dote les agents de cette connaissance au moment du développement du système. Elles décrivent le graphe des interactions possibles entre les agents, tout comme les règles d'un système de production décrivent les relations entre les objets du domaine d'expertise. Ce type de connaissance doit être explicite et donc représenté dans le même langage que celui du savoir de l'expert. Les accointances sont une représentation statique des interactions entre les agents. Au cours d'une session, seule une partie de ce graphe sera sollicitée tout comme dans les systèmes à base de connaissance, une seule partie de la base des règles est activée au cours d'une résolution. Un agent, outre sa connaissance sur le domaine, détient aussi les connaissances sur les relations potentielles qu'il peut avoir avec les autres agents. Ces relations sont exprimées à partir des concepts et des objets du domaine traité. Elles donnent la sémantique des interactions des agents du domaine.

Dans ses travaux Sylvie Trouilhet [Trouilhet, 1993] met en évidence trois types d'accointances pur un agent de type module de connaissances Synergic.

Les accointances peuvent traduire le fait qu'en possession de certaines informations un agent est susceptible de résoudre une tâche ou simplement qu'un agent est intéressé par une information.

Cette connaissance est fréquemment écrite dans le module par des énoncés de faits tels que:

```
# diagnostic de doul-abdo de l'agent Généraliste
# fonction-respiratoire de l'agent Vital
```

Généraliste et Vital sont des agents de Télémac.

Lorsque des partenaires d'un domaine interdisciplinaire se transmettent un dossier contenant des informations corrélées. Ce ne sont plus des faits indépendants mais une collection: nous disons qu'il y a interaction avec des schémas d'accointances. Dans Synergic, ces schémas sont exprimés sous forme de règles de production telles que:

```
si état du patient de l'agent Traumatisme-Thoracique = blessé
et siège du problème de l'agent Traumatisme-Thoracique = 2
et fonction-neurologique du patient de l'agent Traumatisme-Thoracique
...
alors diagnostic de thorax de l'agent Traumatisme-Thoracique
```

Dans cette règle Traumatisme-Thoracique est le nom d'un agent.

La troisième forme de description d'accointances est relative aux classes d'objets. Par exemple si deux agents emploient la classe `animal`, il est impossible de nommer au préalable la liste exhaustive des animaux cités durant une session. Il faut donc transmettre implicitement la liste de toutes les instances d'une classe: ce sont des accointances de classes d'objets. Dans Synergic l'énonciation par l'utilisateur du fait `médor est_un`

`animal` suffira pour déclencher la communication de données sur `médor` vers d'autres agents.

Pour un agent de type boîte noire, les données en entrée, en sortie ainsi que son rôle doivent être précisément connues.

Les accointances jouent donc un rôle central car :

- c'est un savoir qui doit être explicite car il faut raisonner avec, il peut évoluer et peut être expliqué [Piquemal-Baluard, 1992],
- ce savoir est propre à chaque agent pour garantir leur autonomie respective et obtenir réellement une résolution distribuée,
- les accointances décrivent de manière déclarative les entités communes entre les agents et donc leur rôle respectif dans la société,
- elles sont exploitées pour la construction du contenu sémantique des messages à communiquer.
- elles participent à la mise en oeuvre du contrôle dans le système multi-agent. Les accointances tracent le réseau statique des interactions qui sera parcouru au cours de la session. Le parcours dynamique détermine l'intervention des agents dans le traitement global du système.

3.3. Les interactions entre les agents

Dans Synergic, il n'y a pas de notion d'appel et de retour d'agents. Toutes les interactions sont vues en terme de communication. Les requêtes et les réponses sont créées dynamiquement et transmises automatiquement sans distinction. La communication par envoi de messages se fait d'un agent vers ses accointances. Les agents d'une application se comportent comme s'ils coopéraient par partage de tâches et par partage de résultats, bien que rien ne soit spécifiquement programmé pour cela. C'est le contenu du message qui indique à l'agent si on lui demande de résoudre une tâche ou d'intégrer de nouveaux résultats. L'état de ses connaissances lui indique le contenu des messages qu'il envoie à son tour. Aucun message n'est inutile : l'agent destinataire apporte un élément à la résolution et/ou met à jour ses connaissances. La négociation des tâches se fait implicitement car :

- les agents sont volontaires pour s'entraider,
- un agent fera tout ce qu'il peut pour résoudre une tâche, qui est de sa compétence et qui n'est pas en conflit avec ses propres objectifs,
- les connaissances d'accointances indiquent les autres agents partenaires qui pourraient être utilement sollicités pour résoudre des sous-tâches.

Trois éléments interviennent dans les interactions des agents de Synergic : les agents ont des connaissances d'accointances, ils effectuent des actes de langage et utilisent des maximes de la conversation pour garantir une bonne coopération et percevoir les intentions de leurs interlocuteurs.

Les accointances entre agents décrivent l'organisation statique de la société. Une connaissance d'accointances exprime le point de vue que peut avoir un agent sur un autre: c'est une croyance. Ces points de vue peuvent être multiples puisque les agents sollicitent un tiers en fonction de leurs centres d'intérêt, de leurs objectifs et de leurs connaissances. Les connaissances d'accointances (règles et faits) définissent également le contenu sémantique des informations à communiquer. Elles relient les concepts communs à deux agents et confèrent à un agent la possibilité de raisonner sur autrui. Ces concepts peuvent porter sur des entités ou des classes d'objets. Lorsque les terminologies entre les agents sont différentes, l'agent a la possibilité d'employer des règles de réécriture.

Des maximes sont employées pour assurer la concision et la pertinence des messages envoyés. Elles garantissent la véracité des informations et déterminent les instants propices à la communication :

- maximes de manière issues de Grice [Grice, 1975]. L'obscurité d'expression et l'ambiguïté sont levées par les règles de réécriture. Une règle de réécriture relie deux entités synonymes. Les informations contiennent la signature de l'émetteur. Pendant son raisonnement, l'agent a mémorisé, pour chacune de ses accointances, tous les faits de communication transmissibles. Un agent qui partage un objet avec un autre agent n'a pas à préciser toutes les entités communes. "Objet de l'agent X" sous-entend tous les "attribut objet opérateur valeur". Après avoir déterminé tous les faits communicables, l'agent transcrit chaque élément sous forme symbolique, il les marque avec son état et signe de son nom. L'agent généraliste envoie par exemple à Fonctions-Vitales:

```
"? bilan-vital du patient de l'agent Généraliste"
"! âge du patient = 27 de l'agent Généraliste".
```

- maxime de relation issue de Grice. Les informations envoyées correspondent soit aux conditions nécessaires pour résoudre une tâche, soit à des résultats partiels que l'agent n'a pas déjà envoyés. Conditions, tâches et résultats partiels font partie des compétences du récepteur. Dans ce nouveau contexte, l'agent va inférer de nouvelles règles, ce qui lui était impossible auparavant.
- maximes de quantité issues de Grice. Seules les informations communes et modifiées depuis le message précédent sont envoyées. Si l'objet commun est une classe, l'agent sélectionne les entités communes réévaluées.
- maximes de qualité issues de Grice. Un agent communique lorsque ses connaissances ont atteint un état stable. Les informations communiquées issues des connaissances sont donc, pour lui, vraies. Un état stable pour la communication se rencontre quand l'agent se trouve bloqué au cours de son raisonnement parce qu'il a fini ou qu'il a généré des tâches qu'il ne sait pas résoudre.

Les agents de Synergic effectuent quatre actes de langage : affirmer, infirmer, questionner et ignorer. Ces quatre actes forment un sous-ensemble très restreint des actes énoncés par Searle [Searle, 1979], mais ils sont indispensables pour percevoir l'intention associée au contenu propositionnel. L'émetteur indique la façon dont il faut considérer la proposition par un marqueur de force illocutoire : ! pour une affirmation, ˜ pour une négation, ? pour une question et # pour une ignorance. La reconnaissance de ce marqueur amène le destinataire à utiliser d'une manière appropriée la proposition énoncée. Le destinataire est capable de faire l'acte demandé car le contenu propositionnel envoyé fait

partie de ses compétences énoncées dans une ou plusieurs règles d'accointance. L'émetteur pense qu'il peut le faire ; en effet, le contenu propositionnel est déduit en fonction des connaissances d'accointances inférées qui dépendent elles-mêmes de la résolution c'est-à-dire des connaissances inférées du domaine. Il choisit donc le destinataire en fonction de ses intentions, du contexte de la résolution et de l'état des connaissances d'accointances.

Contrairement aux systèmes de traitement d'informations classiques, il n'y a pas de notion d'appel/retour systématique lors des communications. Après l'envoi d'un message, l'agent reprend son traitement sans attendre. Si une synchronisation est nécessaire, elle s'effectue par la connaissance. Un message est composé d'informations communes entre l'émetteur et le destinataire. Chaque information correspond à un des quatre actes de langage possibles (affirmer, infirmer, demander, ignorer).

L'agent communique spontanément quand son raisonnement est stabilisé. Un état stable pour la communication se rencontre quand l'agent se trouve bloqué au cours de son raisonnement parce qu'il a fini ou qu'il a généré des tâches qu'il ne sait pas résoudre. Il envoie les messages aux autres agents dans l'ordre où leurs compétences sont sollicitées dans son raisonnement. Pendant son raisonnement, l'agent a mémorisé, pour chacune de ses accointances, tous les faits de communication transmissibles. Après avoir déterminé tous les faits communicables, l'agent transcrit chaque élément sous forme symbolique et les marque avec son état. Pour éviter toute ambiguïté, les messages sont signés du nom de l'expéditeur.

Les agents de Synergic héritent des mécanismes de communication. Le concepteur de l'application n'a donc pas à expliciter à quels moments l'agent doit communiquer et ce qu'il doit envoyer. Par contre, les agents ont besoin d'avoir une représentation des autres agents.

3.4. L'organisation

En général dans les systèmes multi-agents, l'organisation correspond à la structure organisationnelle du système ou à son architecture. Elle est définie par les liens d'échanges d'information et de contrôle entre les agents du système [Weiβ, 1999]. La société d'agents que met en oeuvre l'environnement Synergic peut être organisée de manière hiérarchique ou hétérarchique.

L'organisation hiérarchique ou semi-hiérarchique peut se déduire de l'observation de la topologie du système à modéliser. Un agent est un exécutant lorsqu'il n'est employé que pour résoudre des problèmes provenant d'autres agents. Cette relation maître-esclave va se traduire par le fait que l'agent qui demande un service a la connaissance du savoir-faire de celui qui rend le service. Dans ce type d'organisation, les agents coopèrent par partage de tâches et/ou par partage de résultats. C'est le cas de l'application médicale Télémac (section 2.3.6).

Quand un problème est posé au système, tel le diagnostic du malade, l'agent Généraliste décompose ce problème en sous-problèmes, tel le diagnostic des douleurs abdominales. Quand il ne peut plus décomposer le problème, il définit grâce à ses accointances que l'agent Douleurs-Abdominales est susceptible de résoudre le sous-problème. Toujours avec cette connaissance, Généraliste va former un ensemble de faits à lui communiquer. La fin de la résolution par l'agent Douleurs-Abdominales le conduit à

communiquer les faits communs qui ont été modifiés à l'agent Généraliste. Si un ensemble de faits communs à l'agent Généraliste et l'agent Fonctions-Vitales sont modifiés dans l'agent Généraliste ils sont automatiquement communiqués vers l'agent Fonctions-Vitales: il s'agit de partage de résultats.

En fait, l'interaction entre deux agents peut être vue uniquement en termes de communication.

L'organisation hétérarchique correspond à des systèmes où il n'y a pas de relation hiérarchique entre les agents. Tous les agents sont au même niveau de spécialité et ont tous la possibilité d'interagir les uns avec les autres, les accointances sont quasiment identiques pour chaque agent, comme par exemple dans l'application des loups et de l'agneau [Gleizes, 1993].

3.5. Les interactions avec l'environnement du système

Le système a la possibilité d'interagir avec son environnement composé de l'utilisateur du système. Pour cela une interface homme-machine :

- permet à l'usager de s'exprimer dans son propre univers, ceci de façon transparente aux techniques de représentation,
- assiste l'usager durant toute la période d'échanges,
- donne l'impression à l'usager de dialoguer avec l'expert de façon naturelle. Pour cela, les questions émanant de l'expert doivent être claires, les plus courtes possibles et homogènes dans leur présentation.

Les caractéristiques de l'interface sont différentes selon le type d'interlocuteur. Un interlocuteur très expérimenté peut s'exprimer dans le même langage que celui qui est utilisé dans le système pour représenter des données. Par contre, un usager ayant peu de connaissance et peu d'expérience ne peut raisonner que dans un langage courant. Ainsi, le système établit d'emblée une communication avec l'utilisateur dans un langage quasi-naturel. Une question est présentée avec tous les choix pertinents possibles. L'interlocuteur peut malgré tout ignorer l'information demandée : il a la possibilité de sélectionner une valeur inconnue. De plus, si aucune des solutions proposées n'est satisfaisante, il peut introduire un nouveau choix. La sélection se fait par un outil de pointage plutôt qu'au clavier, amenant ainsi une certaine simplicité dans l'interaction et diminuant le nombre d'erreurs. Les possibilités de réponses sont oui, non, inconnu.

L'utilisateur peut également s'exprimer en langage naturel à tout instant d'une session par des phrases contenant les faits qu'il estime importants sur le moment. Grâce à une analyse lexicale, les termes reconnus par le système sont replacés dans les contextes possibles. En cas d'ambiguïté, l'interface prend l'initiative du dialogue et présente une liste de possibilités à l'utilisateur. Ce dernier n'a plus qu'à effectuer son choix, sans avoir à réécrire sa phrase avec les termes appropriés

3.6. Application à Télémac

Télémac (1989 – 1993) est un système multi-agent d'aide au recueil sémiologique et à l'approche diagnostique réalisé en collaboration avec le C.C.M.M. (Centre de Consultations Médicales Maritimes) du C.H.U. (Centre Hospitalier Universitaire) de Toulouse. La fiabilité des consultations médicales des personnes en situation d'isolement ou d'éloignement effectuées à distance par un tiers est étroitement liée à la qualité des informations cliniques transmises. L'assistant de télémédecine doit donc être guidé pour pratiquer un examen clinique identique à celui qu'aurait réalisé un médecin expérimenté. Télémac est une application de Synergic pour l'aide au diagnostic médical en télémédecine. Ce système a pour but d'aider des capitaines de bateaux à effectuer des diagnostics médicaux sur des marins malades ou blessés.

Télémac fait coopérer divers agents qui correspondent le plus souvent à des spécialités médicales (Figure 4).

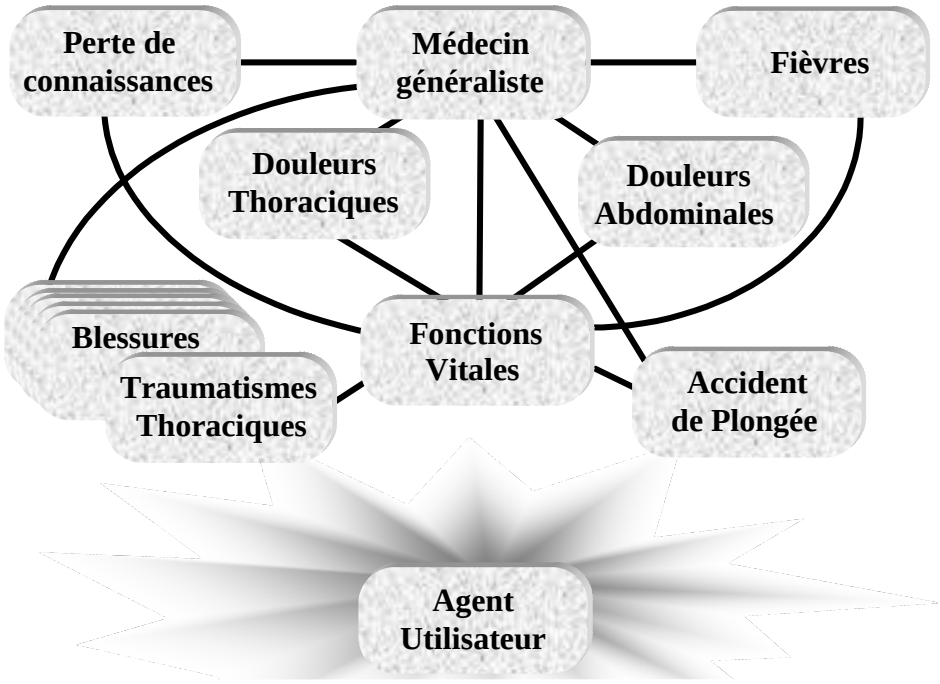


Figure 4 - Les principaux agents de Télémac et leur organisation

Les agents sont au nombre de treize dont douze sont des modules de connaissances et le treizième est un programme d'affichage d'images. Les modules de connaissances permettent d'établir des diagnostics à partir d'un ensemble de symptômes dont le recueil suit une démarche médicale bien précise, ils ont été élaborés par des spécialistes du domaine. Télémac comprend les modules : généraliste, fonctions vitales, douleurs thoraciques, traumatismes thoraciques, douleurs abdominales, pertes de connaissances brèves et blessures (tête, mains, bras, jambes et dos).

L'application Télémac ne va être décrite que pour illustrer le modèle d'agent, la représentation des accointances et la communication entre les agents. Une description plus complète se trouve dans les articles et rapports suivants : [Cannizzo-Puyet, 1991a], [Cannizzo-Puyet, 1991b], [Magues, 1992].

Un agent de Télémac

Un agent de Télémac est constitué de la connaissance du domaine médical : la compétence de l'expert (REGLE 4 Figure 5) ainsi que des accointances possédées par l'agent (REGLE 20 Figure 5). Ces connaissances sont écrites sous la forme de règles de production. Il dispose du moteur d'inférences MIME4 auquel ont été ajoutés des outils de communication entre agents. Ces outils permettent d'ajouter dans la base de faits de l'agent récepteur les différents faits mémorisés par l'agent émetteur au cours de son raisonnement.

```
niveau #2.…
/* REGLE 4 */
si etat du patient = blesse
et conscience du patient est presente
et  l'interrogatoire du patient n'indique pas amnesie-des-evenements et
circonstance de l'accident
et envoi du message-systeme
et localisation des douleurs-ou-blessures
et circonstance de l'accident = chute-de-sa-hauteur
et bilan des fonctions-vitales
 et etude de trauma-dos
et etude de trauma-tete
et etude de trauma-thorax
et etude de trauma-abdo
et etude de trauma-mbsup
et etude de trauma-mbinfet etude de trauma-main
alors diagnostic du patient = realise

niveau #1..…
/* REGLE 20 */
si etat du patient = blesse
et envoi du message-systeme
et localisation de douleurs-ou-blessures est 1
ou examen_systematique de douleurs-ou-blessures est 1
et diagnostic de trauma-tete = realise
alors etude de trauma-tete = realisee
/* REGLE 22 */
si etat du patient = blesse de l'agent ttete
et localisation du douleurs-ou-blessures est 1 de l'agent ttete
et circonstance de l'accident de l'agent ttete
alors diagnostic de trauma-tete de l'agent ttete
```

Figure 5 - Des connaissances de l'agent Traumatisme Thoracique de Télémac

La constitution des accointances a été réalisée au fur et à mesure de l'écriture des modules de connaissances relatifs à une spécialité médicale.

La communication dans Télémac

Dans Télémac, l'agent Généraliste possède des règles d'accointance qui lui permettent de distribuer les sous-tâches qu'il a engendrées. Il possède notamment la règle ci-dessous qui énonce les concepts partagés avec une de ses accointances : l'agent Douleurs-Abdominales ainsi que les conditions de communication. Cette règle indique que l'agent Douleurs-Abdominales peut résoudre le `diagnostic de doul-abdo`, si le patient présente des douleurs à l'abdomen. Pour demander le traitement de cette tâche, Généraliste doit aussi communiquer des informations sur le sexe, l'âge du patient et les caractéristiques de la douleur.

```
      si la localisation de douleurs-ou-blessures est l'abdomen de
l'agent Douleurs-Abdominales
      et l'état du patient = malade de l'agent Douleurs-Abdominales
      et le sexe du patient de l'agent Douleurs-Abdominales
      et l'âge du patient de l'agent Douleurs-Abdominales
      et la topographie de la douleur de l'agent Douleurs-Abdominales
      et l'irradiation de la douleur de l'agent Douleurs-Abdominales
      alors le diagnostic de doul-abdo de l'agent Douleurs-Abdominales
```

En réalité, Douleurs-Abdominales sait trouver "diagnostic de l'abdomen" ; c'est la règle de réécriture ci-dessous qui relie les deux terminologies.

```
      si diagnostic de l'abdomen
      alors diagnostic de doul-abdo de l'agent Généraliste = diagnostic
de l'abdomen
```

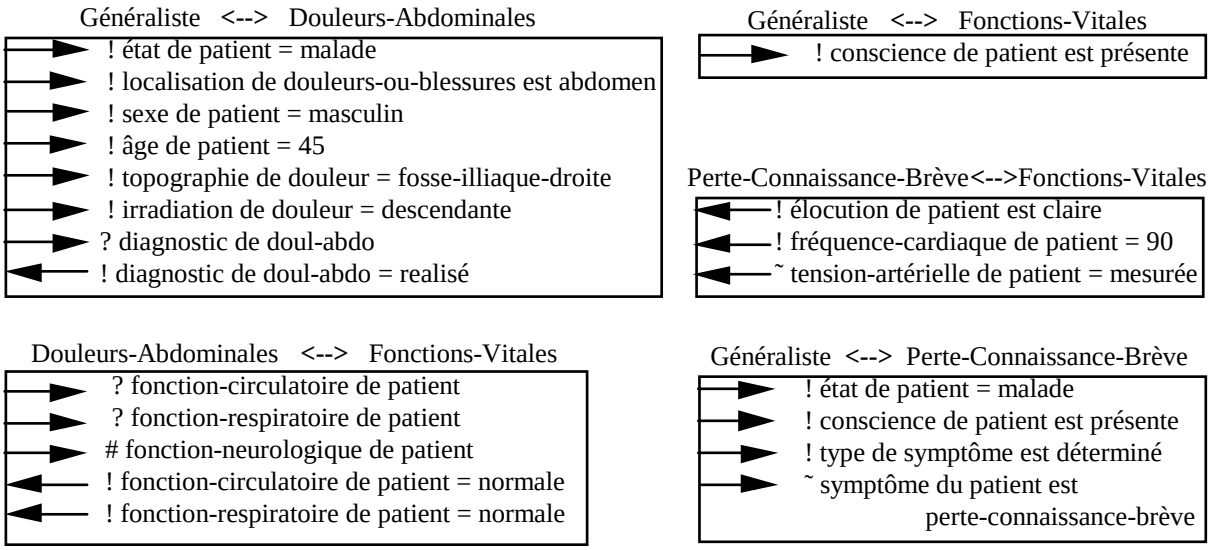


Figure 6 - Historique des messages échangés lors d'une session

Le synoptique précédent donne le contenu des messages échangés entre les agents au cours d'une session dans laquelle le patient présente des douleurs à l'abdomen. Quatre agents ont dialogué entre eux. La flèche qui suit chaque proposition indique le sens de la communication ; dans la réalité, ce dernier est explicitement indiqué dans les messages par

la signature de l'émetteur ("de l'agent ..."). Les probabilités (1) (sur 20) des diagnostics retenus sont affichés à la fin de la session ; ici, quatre diagnostics sont possibles :

```
probabilité de colique-néphrétique est 18
probabilité de appendicite-aiguë est 6
probabilité de colopathie est 4
probabilité de occlusion-intestinale est 4
```

3.7. Contributions

En 1993, la plate-forme de développement de Synergic est opérationnelle et est validée sur des applications telles que : Télémac mais aussi en recherche documentaire avec le projet EURISKO (Christine Régis) [Barthes, 1987] et pour la prévision avec le projet SEC en collaboration avec la DIREN (Pierre Glize, Claudine Féliu et Eric Montgénie) [Féliu, 1994], [Montgénie, 1994].

Les agents de Synergic possèdent quatre caractéristiques principales : ils sont autonomes et poursuivent un objectif individuel, ils raisonnent avec leurs connaissances sociales et les communications sont intentionnelles et spontanées. Dans ce modèle, les agents peuvent s'organiser en adéquation avec le type de l'application : groupe hiérarchique ou hétérarchique, agent dirigeant les activités d'un sous-groupe de la société.

En parallèle avec l'élaboration du modèle d'agent de Synergic et de la plate-forme, la dynamique dans le système a été étudiée et fait l'objet de la section suivante. Cette dynamique consiste à ajouter ou enlever un agent et à faire en sorte que le système s'adapte à sa nouvelle composition

4. Vers un système ouvert

La plate-forme Synergic a été validée par une application en télémédecine : Télémac. Parce que les experts n'étaient pas toujours disponibles et parce que le volume de connaissances nécessaires était très important, la conception des modules de connaissances s'est réalisée de manière incrémentale. Le système était régulièrement enrichi par un nouveau module expert. Cette contrainte de développement s'est traduite par le besoin d'une dynamique importante au niveau du système : le système doit être ouvert [Costa, 1996] [Georgé, 2003b]. L'intégration d'un nouvel agent nécessite que les autres puissent interagir avec lui et donc qu'il soit informé de la présence de ces nouvelles compétences ou tout simplement de l'existence d'un autre agent. Pour cela, les connaissances sociales des agents doivent être enrichies de connaissances sur cet agent.

La société d'agents peut aussi être modifiée par le retrait d'un agent. Dans ce cas, le système multi-agents doit pouvoir continuer à fonctionner en mode dégradé. Il faut que les agents puissent s'adapter à cette nouvelle société. Dans ce cas aussi, les agents doivent mettre à jour leurs connaissances sociales de manière dynamique. En fait, la mise à jour

(1) Le terme probabilité est employé par les médecins, mais il ne correspond pas à la notion mathématique. C'est plutôt le degré d'évocation d'un diagnostic.

d'un agent dans la société nécessite l'adaptation des autres agents à cette nouvelle situation par la modification de leurs accointances. C'est une forme particulière d'apprentissage puisqu'elle est relative aux connaissances d'accointances. L'acquisition des accointances est guidée dans l'environnement Synergic par un outil: le générateur d'accointances développé dans le cadre de la thèse de Sylvie Trouilhet [Trouilhet, 1993].

4.1. Le générateur d'accointances

L'exploitation des accointances pour la mise en oeuvre des interactions permet d'avoir un système modulaire. Il n'est pas nécessaire pour concevoir une application de connaître dès le début tous les agents qui interviendront. Les agents sont autonomes dans le sens où pour exister ils n'ont pas besoin de l'existence des autres. La modularité du système permet d'ajouter un agent sans modifier la structure générale existante. Une mise à jour d'un agent (ajout / retrait / modification) demande non seulement la modification des connaissances de ce dernier, mais aussi la révision des interactions potentielles avec d'autres agents. Ces étapes d'acquisition et de mise à jour des accointances sont réalisées par le générateur d'accointances.

Pour un module de connaissance conçu avec Synergic, au cours de l'étape d'acquisition des accointances, le logiciel propose l'ensemble des interactions potentielles entre cet agent et la société existante. C'est aux concepteurs de valider les choix proposés pour une mise à jour définitive. Le générateur d'accointances permet aussi de créer une "interface" entre l'agent boîte noire et les autres agents du système. Cette interface met notamment en correspondance les paramètres formels et les éventuels paramètres effectifs dans le cas d'un logiciel.

Exprimer les accointances d'un agent consiste à isoler, parmi ses connaissances stratégiques, les problèmes qu'il peut résoudre, ainsi que les faits initiaux nécessaires à la résolution. Ces problèmes et conditions sont respectivement des antécédents et des conséquents de règles appartenant au niveau supérieur. Il convient ensuite de confronter ces accointances avec les connaissances des autres agents afin de vérifier s'ils ont des intersections de connaissances. En général, les agents désignent une entité par le même nom. Toutefois, les agents sont libres de s'exprimer avec leur propre vocabulaire. Nous n'excluons donc pas le cas où deux agents ont une terminologie différente pour une même entité.

Dans un premier temps, le générateur va donc examiner les agents deux à deux, afin de détecter les objets qu'ils ont en commun (et qu'ils désignent de la même manière). Pour un objet commun, il étendra l'examen à ses attributs pour obtenir le ou les entités communes.

Dans un second temps, ce sont les experts humains qui valident les règles construites et les informations communes trouvées. Eventuellement, ils demandent une rectification s'ils se rendent compte que deux agents désignent une entité commune différemment. Le générateur ajoute alors une règle de réécriture pour mettre en relation les deux termes.

4.2. Application à Télémac

Le principe de génération du savoir-faire a été appliqué à Télémac. Prenons trois règles appartenant au généraliste et traitant d'un patient malade (figure 7).

```
Règle 1
si l'etat du patient = malade
et la zone d'examen = abdomen
et symptome du patient  = nausee
alors examen du patient est effectue

Règle 2
si le diagnostic de doul-abdo
alors l'etude de doul-abdo est effectuee

Règle 3
si examen du patient est realise
et etude de doul-abdo est effectuee
alors diagnostic du patient est effectue
```

Figure 7 – Exemple de module de connaissances

A partir de ces règles, trois ensembles de connaissances sont construits :

G1: les informations figurant en conséquents de règles. Ce sont les conclusions déduites par le module à partir des faits initiaux et des faits déduits par le raisonnement. Ces résultats font partie du savoir-faire. Dans l'exemple, cet ensemble se réduit au fait `diagnostic du patient`.

G2: les informations utilisées en antécédents et en conséquents de règles. Ce sont des informations internes au module qui lui servent dans son raisonnement. Dans l'exemple, `etude de doul-abdo` et `examen du patient` sont des informations internes et donc réservées au module.

G3: les informations se trouvant uniquement en antécédents de règles. Ces faits sont utilisés pour répondre aux problèmes posés. Les conditions d'activation d'un module pour une tâche donnée font donc partie de cet ensemble. Il convient ensuite d'isoler pour chacun des résultats, l'ensemble des faits qui constituent les informations initiales minimales. C'est à l'auteur du module d'extraire de l'ensemble des faits les conditions d'activation. Il doit donc désigner les informations initiales nécessaires pour le généraliste. Le générateur doit soumettre chacun des faits trouvés et ne conserver que ceux qui ont été sélectionnés par l'expert du domaine. Dans l'exemple, les deux faits `l'etat du patient` et `la zone d'examen` sont des informations initiales.

La règle de savoir-faire écrite pour ce problème est la suivante:

```
Règle 4
si l'etat du patient de generaliste
et la zone d'examen du generaliste
alors le diagnostic du patient de generaliste
```

Dans cette règle, les experts ont désigné les faits `l'etat du patient` et `la zone d'examen` comme condition d'activation du module généraliste. Par contre, le fait `le symptome du patient` est demandé à l'utilisateur en cours de résolution.

La règle qui utilise `diagnostic de doul-abdo` apparaît au niveau le plus bas dans le module généraliste; elle dit que pour conclure sur `l'étude de doul-abdo`, il faut connaître `le diagnostic de doul-abdo`. Il s'agit d'une sous-tâche à résoudre qui peut être traitée par un agent du système.

Pour que cette tâche soit traitée, le module de savoir-faire doit connaître les agents qui manipulent ce concept, à savoir les modules généraliste et douleurs-abdominales. Pour cela, deux règles de réécriture sont nécessaires.

La règle 5 fait la correspondance entre le module généraliste et le module de savoir-faire, la règle 6, entre le module de savoir-faire et le module douleurs-abdominales. Elles sont de la forme:

```
Règle 5
si douleurs de l'abdomen
alors diagnostic de doul-abdo de generaliste = douleurs de l'abdomen
Règle 6
si diagnostic de l'abdomen de douleurs-abdominales
alors douleurs de l'abdomen = diagnostic de l'abdomen de douleurs-
abdominales
```

Le fait `diagnostic de l'abdomen` fourni par le module douleurs-abdominales est mis en correspondance avec le fait `diagnostic de doul-abdo` du module généraliste. La règle 5 sera mise dans le module douleurs-abdominales et la règle 6 dans le module généraliste.

Dans cette étape, le générateur peut aider les experts à trouver les liens entre agents. Il propose une liste d'objets utilisés dans un agent; lorsqu'un objet est commun à un autre agent, il présente les attributs de cet objet. Le générateur construit automatiquement les règles à partir des couples (attribut, objet).

Pour des agents programmes, le savoir-faire s'exprime de manière identique. Sa construction est plus aisée puisque les conditions d'activation d'un programme et les résultats fournis sont ses paramètres d'appel.

4.3. Contributions

L'approche distribuée permet une méthode de développement de projet totalement incrémentale, en effet, les agents d'une application sont ajoutés à la collectivité au gré des besoins des concepteurs sans une vue globale préalable. C'est ce qui a eu lieu lors de la conception des treize modules de connaissances de Télémac qui ont été développés par des experts différents à des moments différents.

L'utilisation du générateur d'accointances a permis aux concepteurs d'accélérer la création et la mise à jour des accointances dans le système. Mais ces opérations ne se font pas pendant l'exécution du système car les experts doivent valider les connaissances

expertes utilisées dans l'expression des accointances. Cette contrainte entrave la capacité d'adaptation du système pour prendre en compte l'apparition ou la disparition d'un agent dans le système sans l'intervention d'un humain. C'est pourquoi bien que modulaire le système obtenu n'est pas ouvert dans le sens où les agents ne peuvent pas être ajoutés en cours de fonctionnement. Ces travaux ont permis de renforcer ma conviction qu'il faudra pour la résolution de problèmes complexes avoir cette propriété d'ouverture.

5. Vers l'intelligence collective

Les interactions apparaissent comme l'élément nouveau dans les systèmes multi-agents par rapport aux logiciels classiques. C'est un concept clé car c'est par les interactions que le contrôle de la résolution est réalisé. Les travaux sur les accointances dans Synergic et Télémac réalisés par Sylvie Trouilhet durant sa thèse nous ont confortés pour faire des interactions notre objet d'étude.

Contrairement à l'Intelligence Artificielle classique, mais comme les réseaux neuronaux et la vie artificielle, l'IAD-SMA présuppose que l'intelligence peut être issue de l'agrégation de compétences simples. Dans le connexionnisme, un neurone n'a pas de tâche précisément assignée et n'a d'importance que dans la collectivité, hormis les neurones des couches d'entrée et de sortie. En vie artificielle, la population n'est là que pour faire émerger, après plusieurs générations, quelques individus mieux adaptés aux contraintes de l'environnement spécifié. La terminologie IAD possède encore en arrière-plan une vue centralisée de la décomposition d'une activité collective, alors que les travaux s'orientent beaucoup plus sur l'aspect d'autonomie de l'agent tout en maintenant la cohésion de la société par les interactions. En Intelligence Artificielle Collective, l'activité collective complexe est la résultante de traitements individuels plus simples réalisés de manière autonome chez les agents. Ce comportement collectif intelligent est issu de l'aptitude de l'agent individuel à raisonner avec et sur ses croyances (et donc sur ses interactions), pour acquérir de nouveaux savoirs et transformer l'organisation de la société d'agents. Cette approche conduit à une réutilisation de travaux développés en Intelligence Artificielle comme la fusion de données ainsi qu'à une extension des théories, telles que :

- Le comportement d'une société d'agents peut être vu comme une recherche de buts distribués dans un espace d'états. Avec des agents autonomes, l'espace global de recherche correspond à l'union des espaces locaux au sein desquels chaque agent travaille. Les algorithmes centralisés classiques de parcours d'espace d'états ne peuvent plus être appliqués intégralement sans y inclure une dimension de communication entre les agents pour gérer l'interdépendance entre les sous-buts.
- Le maintien de la cohérence est aussi un thème classique en Intelligence Artificielle. Des approches collectives ont été étudiées notamment par Lesser et Corkill avec la méthode FA/C [Lesser, 1981], [Durfee, 1985].

Le projet de recherche qui nous anime est donc de faire de la résolution de problèmes autrement qu'en suivant des schémas de décomposition de problèmes en sous-problèmes comme cela est préconisé en résolution distribuée de problèmes. Une nouvelle

voie consiste à explorer l'interaction. Une première étape a été de valider que l'interaction au sein d'un collectif avec des échanges de buts et des résolutions de conflits permet d'obtenir des performances supérieures aux résultats obtenus par un ensemble d'entités qui n'interagissent pas. Cette validation s'est faite par l'expérimentation sur un problème jouet : les loups et l'agneau.

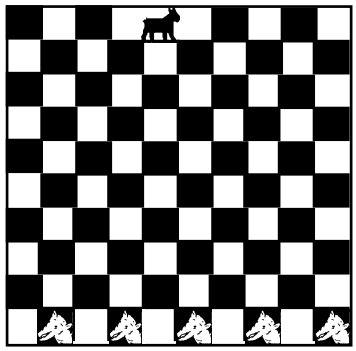


Figure 8 - Les loups et l'agneau Etat initial

5.1. Application : le jeu des loups et de l'agneau

Le jeu des loups et de l'agneau [Gleizes, 1993] a été implémenté avec Synergic dans le but d'étudier différentes stratégies d'interactions entre les loups.

Ce jeu est joué sur un damier de 100 cases et comporte six agents (voir figure 8). Les agents "loups" sont au nombre de 5 et ne peuvent se déplacer qu'en diagonale et que vers l'avant d'une case à chaque fois. L'agent "agneau" joue aussi sur les cases noires et se déplace d'une case en diagonale mais dans tous les sens. Dans la position initiale les cinq loups sont alignés au fond de l'échiquier et l'agneau est sur la ligne du haut sur la case noire du centre. Le but du jeu est que l'agneau se sauve sur une case de la ligne du bas (point de départ des loups), tandis que les loups doivent cerner l'agneau pour le dévorer. Un des loups joue alternativement avec l'agneau.

Pour montrer l'intérêt du savoir social dans une stratégie collective, le comportement de l'agneau est identique dans tous les cas et suffisamment élaboré pour qu'il ait des chances de survie non négligeables. Seuls les loups ont un savoir évolutif, ce qui nous permet de comparer les performances respectives dans leur aptitude collective à cerner l'agneau.

Les loups et l'agneau se communiquent seulement leurs positions respectives (abscisse et ordonnée). Hormis la première stratégie de base, les loups se transmettent aussi leurs intentions de déplacement. Mais leur savoir ne contient aucun envoi de message explicite: il n'existe pas d'ailleurs dans la grammaire de Synergic de prédicat permettant l'envoi de message. C'est au cours du raisonnement d'un agent que Synergic construit la liste des informations potentielles (données ou buts) à envoyer vers d'autres partenaires. Ce modèle de communication est fondé sur des actes de langage et des maximes de la conversation [Trouilhet 1993].

Les différentes stratégies.

Cas 1 : La coopération sans savoir social ni communication

Il n'y a donc pas de communication directe entre agents, mais seulement des actions perçues par autrui. Dans cette situation, les loups n'ont qu'une stratégie individuelle consistant en un simple instinct d'attirance vers l'agneau. Ils perçoivent l'occupation du damier par l'agneau et les autres loups. Un arbitre du jeu permet de suivre les règles précédemment énoncées, sans influencer les actions des autres agents. Nous constatons que

l'agneau arrive systématiquement à gagner car les loups jouent en ordre dispersé et donc laissent des espaces libres sur le damier, dans lesquels l'agneau peut se glisser.

Cas 2 : La coordination hiérarchique entre agents hétérogènes

Dans cette seconde situation, un loup (celui du milieu du damier) est apte à raisonner sur la collectivité. Les agents n'ont plus des compétences identiques car l'un possède une stratégie plus élaborée que ses partenaires. Il ne **donne pas d'ordre** aux autres loups, mais décide seulement en fonction des propositions. Il n'a pas qu'une stratégie individuelle car il sait si le déplacement qu'il propose est intéressant pour la stratégie de la société des loups. Rien qu'avec cette modification mineure, les loups sont arrivés à bloquer l'agneau une fois sur dix.

Cas 3 : La coopération avec savoir social sans communication

Tous les loups ont maintenant une stratégie identique à celle du précédent "loup social". Ils cherchent donc à avancer collectivement vers l'agneau sans négocier leurs éventuels conflits, mais en tentant de les éviter. Ils ont seulement la perception des autres joueurs comme dans la première situation et sans la hiérarchie de la seconde situation. L'organisation est simple et la stratégie de base de rapprochement est identique, mais les connaissances sont beaucoup plus élaborées car chaque loup analyse les propositions des autres en fonction de leur propre stratégie. On conçoit aisément que chacun jouant aussi bien que le chef précédent, ils arrivent à bloquer l'agneau dans plus de situations: quatre fois sur dix.

Cas 4 : La Coopération avec conflit et négociation

Malgré le comportement collectif précédent, il arrive que deux loups suggèrent des déplacements dans la même case. Dans ce cas, l'arbitre qui est neutre, choisit aléatoirement. L'intérêt est de minimiser ces conflits. Pour cela, chaque loup **communique aux autres ses intentions** pour voir si elles n'entrent pas en conflit. Si c'est le cas, il vérifie s'il ne peut pas modifier ses propres intentions pour éviter des conflits avec ses voisins.

L'intérêt de cette méthode est que la stratégie individuelle n'est pas plus élaborée que dans le cas précédent. Néanmoins l'agneau est cerné neuf sessions sur dix.

5.2. Résultats

En analysant les résultats de la société de loups selon les méthodes employées, on s'aperçoit qu'il y a un taux de réussite qui s'accroît au fur et à mesure que la stratégie des loups devient plus élaborée. La gradation des performances des loups va de pair avec une augmentation de leur comportement collectif, nettement visible dans leur répartition sur l'échiquier. Il y a deux changements très nets dans les performances :

- le premier intervient lorsque les loups prennent en compte l'existence de leur groupe dans leur stratégie qui de ce fait devient collective,
- le second intervient lorsque les loups négocient leur déplacement entre eux.

Dans ces deux cas, c'est le savoir social qui est à la source des échecs de l'agneau. Cela indique clairement que les actions personnelles bien que suffisantes dans certaines situations élémentaires sont tout de même moins performantes qu'un comportement collectif.

5.3. Contributions

Dans cette période la coopération est utilisée pour faire du partage de tâches comme dans le Contract Net de Davis [Davis, 1983], pour résoudre des conflits comme dans le système de Cammarata [Cammarata, 1983]. Thierry Bouron [Bouron, 1992] définit un ensemble d'indices de coopération permettant de la caractériser et de la quantifier. Dans notre approche la coopération est étudiée en lui accordant un sens qui va au-delà de travailler ensemble et qui s'intéresse à l'attitude coopérative que peut avoir un agent. La mise en œuvre de comportements coopératifs dans le cadre de l'application du loup et des agneaux a montré que mettre en œuvre la coopération pour guider le comportement des agents dans le système permettait d'améliorer les performances de la résolution. Ces résultats vont dans le sens des résultats obtenus un peu avant par Hogg and Huberman dans [Hogg, 1992] qui montrent que quand des agents coopèrent pour résoudre un problème de manière distribuée, ils peuvent le résoudre plus rapidement qu'un agent travaillant seul.

6. Enseignements de cette période

Les travaux entrepris lors de cette période ont eu le même objectif : expliciter et comprendre comment une fonction collective pertinente, telle que la résolution d'un problème, peut-être obtenue par l'interaction des composants du système et expérimenter les modèles construits. Les projets qui ont été menés pour mettre en œuvre les concepts élaborés dans les architectures constituent une méthode de travail très riche. D'un côté, elle permet la validation expérimentale des modèles. D'un autre côté, les contraintes des applications induisent de nouveaux modèles. Ainsi, la nécessité de concevoir Télémac de manière incrémentale a été un des éléments décisifs dans la décentralisation du contrôle. L'étude de la résolution distribuée de problèmes a permis de mieux définir notre objet d'étude, à savoir les systèmes multi-agents et les agents, et de poser de nouveaux problèmes.

6.1. Les systèmes multi-agents

A l'issue de cette période, les principales caractéristiques des systèmes multi-agents que nous souhaitions étudier, ont été déterminées. Dans ces systèmes, la connaissance du domaine et la connaissance du contrôle doivent être distribuées dans les agents du système. Les traitements effectués par les agents sont asynchrones et un agent n'a qu'une vue partielle du système et de son environnement.

De plus, les projets élaborés montrent que ces systèmes complexes doivent dans certains cas être ouverts et comporter des agents hétérogènes. Le système est ouvert car des agents peuvent être ajoutés ou enlevés du système en cours d'exécution. Le système est hétérogène dans le sens où ils ne sont pas conçus sur le même modèle. Par exemple dans Télémac, le système comporte des bases de connaissances et des programmes comme l'interface homme-machine.

Une conclusion importante de ces travaux est que l'interaction est porteuse du contrôle de résolution dans ces systèmes et établit l'organisation du système. Les moyens de mettre en œuvre l'interaction passent soit par l'environnement soit par une communication directe entre les agents. C'est pourquoi à ce moment-là, j'ai animé un groupe de travail de l'AFCET appelé « Interaction » dont l'interaction entre les agents était l'objet d'étude.

6.2. Les agents

A partir des modules de connaissances et des programmes qui ont été les premiers agents étudiés, nos travaux ont abouti à la définition d'un **modèle d'agent**. Un agent est composé :

- de compétences : les traitements qu'il sait effectuer, par exemple le raisonnement expert dans Télémac,
- de représentations des autres : les accointances dans Télémac. L'agent raisonne sur ses connaissances sociales et coordonne ses actions avec les autres agents,
- d'un langage de communication : dans Télémac, la constitution des informations à échanger à partir des règles d'accointances inférées et leur mise à disposition dans la base de faits de l'agent récepteur, l'utilisation des actes du langages de Searle et des maximes de Grice pour la constitution des messages. L'agent communique spontanément avec les autres agents ce qui facilite la mise en œuvre. Le programmeur n'a pas à gérer le moment de communiquer, avec qui, quel contenu doivent avoir les messages … L'agent est aussi capable d'interagir avec l'environnement du système.
- d'aptitude : le moteur d'inférences dans Télémac qui permet à l'agent de raisonner
- d'attitude sociale : les agents de Télémac sont coopératifs.

Il ressort de nos travaux que l'agent de Synergic a deux caractéristiques fondamentales :

- L'agent est autonome.

L'autonomie d'un agent relativement aux autres et à son environnement entraîne l'impossibilité de prédéfinir le choix des interlocuteurs, l'instant du dialogue et le contenu des messages. Il devient alors nécessaire d'employer des conventions acceptées par tous les agents afin d'obtenir des communications à la fois concises et pertinentes et de garantir un comportement collectif cohérent. Les agents de Synergic héritent des mécanismes de communication adéquats avec ces comportements, qui évitent au concepteur d'application de spécifier à quels moments l'agent doit communiquer et ce qu'il doit envoyer. Une démarche similaire était employée dans la même période par le système Grate [Jennings, 1992], [Wittig, 1992].

- L’agent est coopératif.

Il tente en permanence de satisfaire autrui, compte tenu de ses compétences et de ses croyances. L'activité coopérative décrite dans les cinq points suivants est dictée par ses compétences et croyances instantanées et passe toujours par le biais de communications directes ou indirectes :

- L'agent accepte des données sans condition, mêmes celles qui mettent en cause ses connaissances propres. L'agent sait qu'il est dans un monde non monotone et que l'agent émetteur est lui-même coopératif.
- L'agent accepte un nouveau but sollicité par autrui si et seulement si il considère qu'il peut atteindre un état du monde dans lequel tous ses buts courants ainsi que le nouveau sont vérifiés.
- L'agent persiste dans ses buts implicitement, sauf s'il considère que ses compétences ne lui permettent plus d'obtenir un état ultérieur du monde dans lequel ce but puisse être vérifié.
- L'agent communique spontanément une donnée à autrui s'il croit qu'un autre agent partage cette donnée mais avec un état différent. Cet acte de communication s'effectue dans un état stable : l'agent est bloqué dans son raisonnement (but atteint ou nouvelle tâche pour autrui).

6.3. De nouvelles problématiques

Pour la résolution de problèmes avec SYNERGIC, les agents représentent en général le savoir-faire d’un expert. Dans la problématique de recherche de conception de systèmes complexes, il semblait alors nécessaire de trouver des mécanismes pouvant s’appliquer de manière plus générale à des systèmes où la connaissance était détenue par des experts ainsi qu’à des systèmes nécessitant des compétences plus simples. Quelle architecture donner à des agents non cognitifs ?

Les travaux précédents nous ont permis de définir une architecture d’agent. Mais rien ne permettait de guider le concepteur pour construire son agent « en dynamique » c’est-à-dire comment devait se comporter l’agent. Ce qui a motivé nos recherches était de trouver un mécanisme pour guider les agents au cours de la résolution.

Etudier un système ouvert c’est étudier la dynamique du système (en son sein) et lui permettre de continuer à fonctionner. Cette problématique est très proche de l’étude d’un système devant évoluer dans un environnement dynamique qui suscite la question suivante : comment s’adapter à un environnement ?

A cette époque, nous étions convaincus que l’apport des systèmes multi-agents était une technique de conception de systèmes que le client ne pouvait pas spécifier complètement au départ et qui devaient s’adapter pour réaliser ce pourquoi on les concevait. C’est dans cette direction que j’ai continué mes travaux.

7. Bibliographie relative au chapitre 1

[Agha, 1985] Agha G., Hewitt C., *« Concurrent Programming Using Actors: Exploiting Large-Scale Parallelism »,* Readings in Distributed Artificial Intelligence - Bond A.H. and Gasser L Eds Morgan Kaufmann Publishers, 1985

[Avouris, 1992] Avouris N., Gasser L., *« Distributed Artificial Intelligence: Theory and Praxis »,* Computer and Information Science, volume 5 Kluwer Academic Publishers, 1992

[Barthes, 1987] Barthes C., Frontin J., Glize P., *« EURISKO: An artificial intelligence tool for automatic online information retrivial »,* 11th International Online Information Meeting, London, December 1987

[Benchimol, 1986] Benchimol G., Lévine P., Pomerol J-C., *« Systèmes experts dans l'entreprise ».* Gestion et productique. Paris: Editions Hermès, Mai 1986

[Bond, 1988] Bond A. H., Gasser L. *« Readings in Distributed Artificial Intelligence »,* Edited by Alan H. Bond and Les Gasser, Morgan Kaufman Publishers, Inc, San Mateo California, 1988

[Bouron, 1992] Bouron T., *« Structures de communication et d'organisation pour la coopération dans un univers multi-agents »,* thèse de l'université de Paris VI, Novembre

[Bratman, 1987] Bratman M., *« Intentions, Plans, and Practical Reasoning »,* Harvard Univ. Press, Cambridge MA

[Brooks, 1986] Brooks R. A., *« A robust control system for a mobile robot »,* IEEE Journal of Robotics and Automation, 2 (1) :14-23, 1986

[Bura, 1991] Bura S., Drogoul A., Ferber J., Jacopin E., *« Eco-résolution : un modèle de résolution de problèmes par interactions »* Congrès RFIA 1991

[Burg, 1994] Burg B., Arlabosse F., *« ARCHON : une plate-forme industrielle pour l'intelligence artificielle distribuée »,* Architecture et Environnement. Deuxièmes Journées Francophones : IAD & SMA - Voiron, 1994

[Cammarata, 1983] Cammarata S., Mc Arthur and Steeb R., *« Strategies of cooperation in Distributed Problem Solving »,* Proceedings of the International Joint Conference on Artificial Intelligence 1983

[Cannizzo-Puyet, 1991a] Cannizzo-Puyet P., Carpuat B., Gleizes M.P., Glize P ., *« TELEMAC : système multi-agent en télémédecine »,* Convention Intelligence Artificielle - Paris, 1991

[Cannizzo-Puyet, 1991b] Cannizzo-Puyet P., *« TELEMAC: système multi-expert en télémédecine »,* Mémoire CNAM, Toulouse, Février 1991

[Castelfranchi, 1992] Castelfranchi C., Werner E., *« Artificial Social Systems »,* Proceedings of the fourth European Workshop MAAMAW S Martino al Cimino Italy, July 1992, Lectures Notes in AI 830, Springer Verlag 1994

[Chevallier, 1988] Chevallier J., Cohen P., Gleizes M-P., *« La multi-expertise et l'analyse de la valeur au service de l'innovation »,* Congrès International La maîtrise de la valeur. 27-28 Avril 1988 Paris.

[Cohen, 1986a] Cohen P., Gleizes M-P., Soubie J-L., *« SATIN: L'intelligence Artificielle au service de l'Innovateur »*, Génie industriel Juin 1986

[Cohen, 1986b] Cohen P., Glize P, Soubie J-L., *« Prendre en compte le processus de transfert technologique dans l'intelligence artificielle »*, Symposium Européen sur l'utilisation des résultats de recherche et du développement publics. Luxembourg Septembre 1986

[Conte, 1990] Conte R., Miceli M., Castelfranchi C., *« Limits and levels of cooperation : disentangling various types of prosocial interaction »*, in Demazeau Y., Müller J-P. Decentralized A.I.2 Proceedings of the Second European Workshop on Modelling Autonomous Agents in a Multi-Agent World, Saint-Quentin en Yvelines, France August 13-16 1990, North-Holland 1991

[Davis, 1983] Davis R., Smith R.G., *« Negociation as a metaphor for distributed problem solving »*, Artificial intelligence 20 1983

[Debord, 1982] Debord P., *« Contribution à la définition du système d'analyse de scènes SACSO et réalisation de l'interpréteur minimal »*, Thèse de 3ieme cycle Informatique n° 2722 Octobre 1982 LSI UPS Toulouse

[Demazeau, 1990] Demazeau Y., Müller J-P., *« Decentralized A.I. »*, Proceedings of the First European Workshop on Modelling Autonomous Agents in a Multi-Agent World, (MAAMAW'89), Cambridge, England August 16-18 1989, Elsevier Science Publishers B.V. : Amsterdam, The Netherlands 1990

[Demazeau, 1991] Demazeau Y., Müller J-P., *« Decentralized A.I.2 »*, Proceedings of the Second European Workshop on Modelling Autonomous Agents in a Multi-Agent World, (MAAMAW'90), Saint-Quentin en Yvelines, France August 13-16 1990, Elsevier Science Publishers B.V. : Amsterdam, The Netherlands 1991

[Durfee, 1985] Durfee E.H., Lesser V.R., Corkill D.D., *« Increasing coherence in a distributed problem solving network »*, 9th IJCAI, August 1985, Los Angeles, California

[Engelmore, 1988] Engelmore R., Morgan T., « Blackboard systems », Addison-Wesley Publishing Compagny 1988

[Erceau, 1991] Erceau J., Ferber J., *« L'intelligence artificielle distribuée »*, La recherche 233, Juin 1991 Volume 22

[Erman, 1975] Erman L. D., Lesser V.R., *« A multi-level organization for problem solving using many, diverse, cooperating sources of knowledge »*, International Joint Conference on Artificial Intelligence, Septembre 1975 Vol 2, pages 483-490

[Féliu, 1994] Féliu C., *« Intégration d'un système multi-agent dans un réseau d'informations hydrologiques »*, Rapport de DESS-ASIC - Université Paul Sabatier - Juin, 1994

[Gasser, 1987] Gasser L., Braganza C., Herman N., *« Implementing Distributed AI Systems Using MACE »*, Proceeding of the Third IEEE Conference on Artificial Intelligence Applications 1987, pages 315-320

[Gasser, 1989] Gasser L., Huhns M.N., *« Distributed Artificial Intelligence »*, Volume II Pitman London Morgan Kaufmann Publishers, Inc., San Mateo, California, 1989

[Gasser, 1991] Gasser L., *« Social Conceptions of Knowledge and Action : DAI foundations and open systems semantics »*, Artificial Intelligence 47 107-138

[Georgeff, 1987] Georgeff M.P., Lansky A.L., *« Reactive reasoning and planning »,* Proceedings of the sixth National Conference on Artificial Intelligence, pages 77-682

[Gleizes, 1990] Gleizes M-P., Glize P., *« Les systèmes multi-experts »,* Collection Technologie de Pointe, Editions Hermès

[Gleizes, 1992] Gleizes M-P., Glize P., Trouilhet S., *« Le rôle du savoir-faire dans un univers multi-agent »,* 3ièmes journées "symboliques-numériques" (291-302) Paris 14-15 mai 1992

[Gleizes, 1993] Gleizes M-P., Glize P., Trouilhet S., *« Social knowledge of an agent. Une étude des potentialités du savoir social chez l'agent »,* EXPERSYS-93, Paris 6-7 December 1993, IITT-International. Pages 263-272.

[Grice, 1975] Grice, H.P., *« Logic and conversation »,* Syntax and semantics Vol3 - P.Cole & J.L.Morgan Editeurs New York Academic Press 1975

[Haton, 1989] Haton J-P., *« Panorama des systèmes multi-agents »,* Onzièmes journées francophones sur l'informatique. Architectures Avancées pour l'Intelligence Artificielle. Postes de travail, parallélisme, connexionnisme, outils logiciels. Nancy, 12-13 Janvier, 1989, Editeur EC2, pages 247-261

[Hayes-Roth, 1985] Hayes-Roth B., *« A Blackboard Architecture for Control »,* Artificial Intelligence, Vol 26 1985 and in Readings in DAI in Bond 1988 pages 505-540

[Hayes-Roth, 1986] Hayes-Roth B., Johnson M.V., Garvey A., Hewett M., *« Application of the BB1 blackboard control architecture to arrangement-assembly tasks »,* Artificial Intelligence Vol 1 n°2 1986

[Hogg, 1992] Hogg T., Huberman B., *« A better than the best: The power of cooperation »,* Lectures notes in complex systems - Addison Wesley, 1986

[Huhns, 1987] Huhns M.N., *« Distributed Artificial Intelligence »,* Research Notes in Artificial Intelligence. Morgan Kaufmann Publishers, Inc., Los Altos, California 1987.

[Jennings, 1992] Jennings N.R., Wittig T., *« ARCHON : Theory and Practice. DAI : Theory and Praxis »,* N.M. Avouris and L. Gasser Editors - ECSC, EEC, EAEC, 1992

[Laasri, 1987] Laasri H., Maître B., Haton J-P., *« ATOME: Another Tool for developing Multi-Expert Systems Outil d'aide au développement de systèmes multi-experts »,* 6ième Congrès AFCET RFIA, Antibes, 16-20 Novembre 1987

[Laasri, 1989] Laasri H., Maître B., *« Organisation du contrôle dans les architectures blackboard »,* Revue d'IA Volume 3 n°1 1989

[Lenat, 1975] Lenat D.B., *« BEINGS: knowledge as interacting experts »,* Proc. International Joint Conference on Artificial Intelligence 1975, pages 126-133, in Readings in DAI de Bond 1988 pages 161-168

[Lesser, 1977] Lesser R., Ermlan L.D., *« A retrospective view of the HEARSAY II architecture »,* International Joint Confernce on Artificial Intelligence 77, pages 790-800

[Lesser, 1980] Lesser V.R., Erman L.D., « Distributed interpretation: a model and experiment », IEEE Transactions on computers Vol. C-29 N°12 Dec. 1980

[Lesser, 1981] Lesser V.R., Corkill D. D., « Functionally, Accurate, Cooperative Distributed Systems », IEEE Transactions on Systems, Man and Cybernetics, Vol SMC 11, n°1 January 1981, pages 81-96, and in Readings in DAI in Bond 1988 pp 294-310

[Lévine, 1986] Lévine P., Maillard J-C., Pomerol J-C., « DECIDEX, un système intelligent pour l'aide à la décision stratégique », Economics and Artificial Intelligence. Aix-en-Provence Septembre 1986

[Magues, 1992] Magues J.P., Pujos M., Carpuat B., Glize P., Puyet P., Lareng L., *« TELEMAC: A multi-agent system for aided-decision and diagnosis approach »,* AAAI Spring symposium on A.I. in medecine - pp 77-78 - Mars 1992, Standford

[Minsky, 1988] Minsky M., *« La société de l'esprit »,* Intereditions - Paris, 1988

[Montgénie, 1994] Montgénie E., *« Conception de l'interface utilisateur du système multi-expert pour la prévision des crues SEC »,* Rapport de stage DUT - Université Paul Sabatier - Juin, 1994

[Nitas, 1986] Nitas, *« A reasoning system for the guidance of technological transfer »,* Economics and artificial intelligence, Septembre 1986

[Nitas, 1987] Nitas, *« Control structure of a general inference mechanism »,* IASTED Grindelwald, February 1987

[O'Hare, 1996] O'Hare G.M.P, Jennings N.R., *« Foundations of distributed Artificial Intelligence »,* Sixth Generation Computer technology Series Wiley InterScience 1996

[Piquemal-Baluard, 1992] Piquemal-Baluard C., Glize P., *« Explanation constraints in a multi-agent environment »,* Conférence Latino-américaine d'informatique - Panel92 - Septembre 1992, pages 969 - 979

[Searle, 1979] Searle J. R., *« A taxonomy of illocutionary acts. Expression and Meaning, Studies in the Theory of Speech Acts »,* Cambridge University Press 1979 (chapter 1, p. 1-29)

[Smith, 1981] Smith R.G., Davis R., *« Frameworks for cooperation in Distributed Problem Solving »,* IEEE Transactions on Systems, Man and Cybernetics – Vol 11 n°1, 1981

[Soubie, 1996] Soubie J-L., *« Coopération et systèmes à base de connaissances »,* HdR Université Paul Sabatier, Novembre 1996

[Terry, 1988] Terry, A., *« Using Explicit Strategic Knowledge to Control Expert Systems »,* originally published in 1985, reproduced in Blackboard Systems, Engelmore, R. and T. Morgan (Eds), Blackboard Systems, Addison-Wesley, 1988 pages 159-188

[Trouilhet, 1993] Trouilhet S., *« Représentation et traitement des connaissances sociales chez l'agent : Application à l'environnement multi-agent SYNERGIC »,*Thèse de Doctorat de l'Université Paul Sabatier, 8 Juillet 1993.

[Vailly, 1987] Vailly A., *« MEDIA Expérimentation de systèmes experts coopérants »,* Journées PRCIA, Pôle 4 11-13 Novembre 1987, Beaugency

[Weiß, 1999] Weiß G., *« Multiagent Systems, A modern Approach to Distributed Artificial Systems »,* MIT Press 1999

[Werner, 1992] Werner E., Demazeau Y., *« Decentralized AI 3 »,* Proceedings of the Third European Workshop on Modelling Autonomous Agents in a Multi-Agent World, (MAAMAW'91) Kaiserslautern, Germany, 1991, Elsevier Science Publishers B.V. : Amsterdam, The Netherlands 1992

[Wittig, 1992] Wittig T., *« ARCHON, an architecture for multi-agent systems »,* Thies Wittig Editor – Ellis Horwood Limited, 1992

Chapitre II. La coopération et l'auto-organisation

1. Objectifs - contexte

Avant de situer mes travaux de recherche dans le domaine de l'intelligence artificielle distribuée et des systèmes multi-agents, je décrirai mes objectifs de recherche ainsi que mes motivations dans la période 1994 – 1998.

1.1. Objectifs

Les résultats obtenus précédemment ont montré que l'interaction est un concept essentiel des systèmes multi-agents. Parce que notre objet d'étude correspond aux systèmes composés de plusieurs agents en interaction dans le but de réaliser une tâche globale, la coopération devient l'un des thèmes central dans nos travaux. Effectivement, les agents ne sont pas dans un même système pour atteindre uniquement leur objectif individuel comme cela est le cas lors d'une vente aux enchères classique [Workshops on Agent Mediated Electronic Commerce] ou lorsqu'un agent assistant part sur le Web pour rechercher de l'information pour un utilisateur [Workshops on Cooperative Information Agents]. Les agents doivent travailler ensemble pour permettre au collectif d'atteindre son but, comme une équipe de robots footballeurs joue et gagne un match (www.robocup.org/). Cela n'est pas contradictoire avec le fait que les agents peuvent avoir des buts individuels. Un de nos objectifs est donc de comprendre et de définir ce qu'est la coopération pour un agent dans un système multi-agent plongé dans un environnement et de répondre aux questions suivantes : qu'apporte la coopération à la résolution ? Peut-on généraliser l'attitude coopérative des agents ? La coopération est prise en compte comme une attitude sociale d'un agent, elle a été étudiée dans le cadre de la thèse de Valérie Camps [Camps, 1998a] et de divers stages d'IUP [Heurtevin, 1996], [Carré, 1998], [Topin, 1998], ou de DEA [Dotto, 1996], [Morisset, 1997], [Topin, 1999a] que j'ai encadrés.

En 1993, le générateur d'accointances est opérationnel mais l'acquisition de la connaissance sur ce que savent faire des agents ne peut être que semi-automatique car la présence d'un expert est nécessaire pour la valider. En effet, il n'existe pas de moyen automatique pour la validation sémantique d'une règle de production. Le système obtenu est bien modulaire mais pas ouvert dans le sens où l'ajout d'agent ne se fait pas en cours

d'exécution. L'ouverture d'un système est une caractéristique fondamentale pour les futurs systèmes notamment avec le succès du Web en 1994. N'importe quel utilisateur peut apparaître et disparaître sur la toile. Cette dynamique en cours de fonctionnement au sein d'un système devient donc de plus en plus nécessaire. Il est évident pour nous que les systèmes multi-agents doivent pouvoir répondre à ce besoin. De plus, dans ce type de système, les interactions entre un système et son environnement ne sont pas toutes prévisibles. Cette dynamique induite par l'environnement du système ne doit pas entraîner un arrêt du système : le système doit être robuste. Il doit continuer à fonctionner même en mode dégradé et pour cela il doit pouvoir s'adapter. Il paraît donc fondamental, pour nous concepteurs de systèmes artificiels, de trouver de nouvelles méthodes de conception de ces systèmes correspondant à un besoin croissant d'autonomie, de robustesse et de complexité. Les questions auxquelles nous allons devoir répondre sont : quel doit être le modèle d'agent permettant l'adaptation ? Quel doit être le moteur du comportement d'un agent ? Comment concevoir des systèmes qui, plongés dans leur environnement, réalisent la fonction pour laquelle ils ont été conçus ?

C'est pourquoi dans la suite de nos travaux, nous nous intéressons à la notion de système et à l'adaptation des systèmes. Dans un environnement donné, ces systèmes ont à remplir une fonction qui correspond aux tâches que le système est capable d'exécuter. L'adéquation du système est jugée par un observateur extérieur au couple système-environnement et correspond au fait que le système réalise bien la fonction attendue par son environnement. Par hypothèse, ces systèmes évoluent dans des environnements dynamiques et doivent donc continuer à être fonctionnellement adéquats. Les systèmes doivent de manière autonome s'adapter : ils sont donc adaptatifs et pas uniquement adaptables. L'autonomie (la spécification par un système de ses propres lois) n'est pas spécifique aux systèmes cognitifs, mais ce sont eux qui sont le cadre de nos travaux. Nous situons notre approche de la connaissance en ligne directe avec des travaux tels ceux de Maturana et Varela [Maturana, 1980], qui mettent au centre l'idée que la signification émerge de l'histoire du couplage d'un système biologique avec son environnement. Ceci est en opposition avec les vues plus classiques du cognitivisme pour lequel les symboles manipulés sont porteurs de sémantique, celle-ci étant donnée par le concepteur. Notre approche s'inscrit dans l'approche cybernétique dans laquelle les informations manipulées n'ont pas le statut de symbole. Le sens sera donné par l'observateur du système, il n'est pas pré-existant dans le système mais émergent.

Notre objectif est donc de concevoir des systèmes complexes adaptatifs en utilisant le concept de systèmes multi-agents qui adapte son comportement pour réagir à la dynamique de son environnement. Un système complexe est défini comme un système composé d'entités en interaction et dont le comportement est non déterministe (www.wordiq.com/definition/Complex_systems). Il faut entendre par systèmes complexes, les systèmes étudiés en théorie du chaos, vie artificielle, calcul évolutionnaire et algorithme génétique. L'adaptation du système se fait par auto-organisation et la fonction adéquate du système émerge. Le problème est donc de savoir comment concevoir un système auto-organisé qui soit fonctionnellement adéquat, quel est le mécanisme d'auto-organisation ? Les travaux de l'équipe SMAC vont donc essayer d'apporter une réponse en élaborant une théorie des systèmes multi-agents adaptatifs : la théorie des AMAS[7] [Glize, 2000]. Nous ne nous intéressons pas à l'adaptation d'un agent mais nous nous plaçons au niveau de l'adaptation du système multi-agent à un environnement. A cette période de nos travaux,

[7] AMAS : Adaptive Multi-Agent Systems

nous avions l'intuition que la coopération était le concept fort pour concevoir des systèmes multi-agents fonctionnellement adéquats et pour optimiser la résolution.

Le terme théorie que nous employons n'est pas relatif au concept de théorie mathématique qui en général est constituée de formules bien formées, d'axiomes et de tous les théorèmes prouvable à partir de ces axiomes (www.Wordiq.com/definition) mais relatif à une théorie scientifique. La théorie scientifique est définie en épistémologie comme un système de lois et de concepts visant à rendre compte de certains phénomènes expérimentalement établis. Elle a pour but de représenter aussi simplement que possible, aussi complètement et aussi exactement que possible, un ensemble de lois expérimentales. Habituellement en science, une théorie est une explication ou un modèle basé sur l'observation, l'expérimentation et le raisonnement. Ce modèle est testé et confirmé, et permet d'expliquer les données utilisées et offre des prédictions valides qui peuvent être testées.

Les applications comme le projet Télémac, les loups et l'agneau menés en parallèle de nos études théoriques ont permis d'expérimenter et de valider les modèles proposés. Dans nos objectifs de travail, le volet expérimental continue de tenir une place importante. En effet, les applications et l'industrialisation d'applications basées sur les systèmes multi-agents est une garantie pour les recherches menées dans ce domaine. Dans la période décrite ci-après, j'ai participé au projet GIS ANTS et au projet CTI-CNET ARCADIA qui a constitué la partie applicative de la thèse de Valérie Camps. J'ai aussi eu des responsabilités dans le cadre de projets au niveau national (Responsable du Projet région ABROSE) et Européen (responsable pour le site IRIT et responsable du "workpackage System Design and Implementation" dans le projet ABROSE).

Ainsi dans la période allant de 1994 à 1999, mes travaux ont concerné deux axes :

- l'étude de la coopération comme moteur de la résolution,
- l'étude de la coopération comme moteur de l'adaptation.

Dans le cadre de cette activité de recherche, j'ai animé deux groupes de travail. De 1994 à 1997 au sein du groupe de travail de l'AFCET-AFIA sur les interactions dans les systèmes multi-agents, le travail a consisté à organiser des séminaires tous les semestres et à élaborer collectivement un rapport interne au groupe concernant la définition et le rôle des interactions dans un système multi-agent. De 1995 à 1997 avec Vincent Chevrier (LORIA – Nancy), nous avons eu la responsabilité du groupe MARCIA du PRC-GDR. Ce groupe a eu pour objectif de réfléchir sur les concepts d'auto-organisation et d'émergence. Les résultats des réflexions ont fait l'objet d'un article pour les journées du PRC GDR de Grenoble en 1997 [MARCIA, 1997]. Dans le cadre de ces mêmes sujets de réflexion, j'ai participé au groupe de travail de l'AFCET-AFIA sur Emergence dont les travaux ont donné lieu à un article à la conférence JFIADSMA en 1997 [M. R. Jean, 1997].

1.2. Contexte

A partir de 1994, le Web connaît un succès grandissant. Dans ce contexte, les agents autonomes deviennent une technique pertinente pour aller chercher de l'information sur

l'Internet. C'est la période où la société General Magic propose le langage Telescript [White, 1994]. Les chercheurs en IAD-SMA ont jusqu'à présent proposé de nombreuses architectures d'agent et de nombreuses plates-formes, un effort de standardisation est nécessaire, il sera entrepris par le groupe FIPA qui propose en 1997 le langage ACL (Agent Communication Language) de FIPA (www.fipa.org) qui est dans la même lignée que les travaux sur KQML (Knowledge Query Manipulation Language) [Finin, 1994]. En effet, un des axes principaux de FIPA, qui a encore aujourd'hui un impact, concerne la communication entre agents. Ces langages fournissent les principaux actes de langages tels que : affirmer, infirmer, refuser, ne pas comprendre …., que les agents peuvent utiliser au sein de protocoles. La normalisation des ACL a pour objectif de faire interopérer différentes plates-formes agents. De plus, il est communément accepté à cette période, qu'un système multi-agent est un logiciel dans lequel des agents interagissent et travaillent ensemble pour exécuter un ensemble de tâches ou satisfaire un ensemble de buts. Les agents peuvent être homogènes ou hétérogènes, ils peuvent avoir des buts communs ou différents et des humains peuvent avoir à interagir au sein de ces systèmes.

Le Web amène d'un côté de nouvelles applications telles que :

- la recherche d'information Retsina, Amalthea [Moukas, 1996], Infosleuth [Nodine, 1998], ARCADIA [Camps, 1997a],… et son workshop on Cooperative Information Agents
- le commerce électronique ABROSE [Camps, 1998b], Kasbah [Chavez, 1996]…
- la vente aux enchères

mais aussi de nouvelles problématiques telles que :

- la mobilité,
- l'autonomie,
- les applications à grande échelle caractérisées par un grand nombre d'agents
- l'ouverture des systèmes [Hewitt, 1991], [Costa, 1996]
- un besoin d'adaptation et d'apprentissage [Sen, 1995], [Weiβ, 1996],…

Le nombre d'applications grandissant, de nombreuses plates-formes de développement sont conçues dans le but d'aider les concepteurs : AALADIN & Madkit [Gutknetcht, 1998] (www.madkit.org/), DIMA [Guessoum, 1999], JADE [Pitt, 1999] (jade.tilab.com/), ORIS [Harrouet, 1997], PACO [Demazeau, 1993],…Nous avions développé la plate-forme Synergic et l'objectif principal de nos travaux allait être de chercher à proposer dans le cadre d'applications réelles des solutions pour la résolution de problèmes.

C'est donc une période riche et les manifestations concernant les systèmes multi-agents se multiplient avec la création :

- de workshops dédiés qui augmentent énormément en peu de temps : ATAL (Agent Theories Architectures and Languages) en 1994, MABS (Multi-Agent Systems and Agent-Based Simulation) en 1998, CIA (Cooperative Information Agents) en 1997, AMEC (Agent-Mediated Electronic Commerce) en 1998, …. Chacun traitant une ou plusieurs des problématiques exposées précédemment.
- des conférences telles que : ICMAS (International Conference on Multiagent Systems) en 1995 [Lesser, 1995], en 1996 [Tokoro, 1996] et 1998 [Demazeau, 1998],

CKBSe (International Working Conference on cooperating Knowledge Based Systems) en 1994

- de groupes d'échanges par mail (MAAMAW blackboard, DAI Discussion List…)
- de numéros spéciaux de journaux tels que : Communications of the ACM [Riecken, 1994], dans Artificial Intelligence [Agre, 1995], dans Applied Artificial Intelligence Journal [Wooldridge, 1996]…
- d'un réseau d'excellence au niveau Européen : AgentLink I (1998-1999) (www.agentlink.org/).

Les recherches en IAD-SMA sont clairement établies dans cette période avec des sessions à l'ECAI, à l'IJCAI et la parution des premiers livres sur le sujet : celui de Jacques Ferber en français [Ferber, 1995], celui de O'Hare et Jennings en 1996 [O'Hare, 1996] et de Weiß en 1999 [Weiß, 1999]. Dans cette période, sous le terme systèmes multi-agents deux types de systèmes font l'objet de recherches qui peuvent être complémentaires : les systèmes à agent et les systèmes multi-agents. Les premiers types de systèmes concernent les systèmes où la notion d'agent est prépondérante. L'agent est une notion ancienne en Intelligence Artificielle [Russel, 1995] et représente un système intelligent ; l'objet d'étude est l'agent en tant qu'entité qui raisonne et cherche à atteindre ses buts. Ces travaux reprennent des travaux de l'Intelligence Artificielle classique sur le raisonnement et cet agent est en interaction avec d'autres agents. Pour ma part, ces systèmes sont des systèmes à agents et sont à différencier des systèmes multi-agents. En effet, dans ces derniers c'est le système qui est l'objet de l'étude, il a un but à atteindre et est constitué d'agents en interactions. Mes travaux se situent exclusivement dans le champ des systèmes multi-agents. Dans ces systèmes, il existe une gradation dans l'intelligence d'un agent que l'on classe habituellement en trois catégories [Wooldridge, 1995] :

- réactivité : l'agent est uniquement dirigé par les événements perçus dans l'environnement. Il réagit de manière opportuniste à ces changements.
- pro-activité : l'agent n'agit pas simplement en réponse à des changements de l'environnement, mais est aussi capable de s'assigner des buts et de prendre des initiatives pour les atteindre;
- socialité : l'agent interagit avec les autres agents – et les humains.

Les problèmes clés qui se posent sont les suivants :

- quand et comment les agents doivent-ils interagir ? quand et comment doivent-ils coopérer ou être en compétition pour atteindre leurs objectifs ?
- comment développer des stratégies de coordination pour permettre la résolution collective de problèmes ?
- quels sont les mécanismes de négociation que doivent utiliser les agents ? Comment détecter et résoudre les conflits ?
- comment résoudre les problèmes soulevés par la mobilité des agents ?
- quels sont les protocoles de communication disponibles pour des agents ?
- quels standards de communication KQML [Finin, 1994] et KIF (1992), FIPA (www.fipa.org) (1995- 2004) définir pour favoriser l'interopérabilité ?
- la relation local-global : comment passer de comportements locaux à un comportement global ?

- comment l'activité individuelle au niveau de l'agent et les structures et règles sociétales au niveau du groupes sont-elles reliées les unes aux autres ? Ce problème est le micro-macro problème en sociologie. [Weiß, 1999]
- quels sont les mécanismes qui permettent aux agents de garder leur autonomie tout en ayant un système global efficace ?

Mes travaux apportent essentiellement une contribution aux quatre derniers points. Les trois premiers ont déjà été relativement approfondis dans mes travaux de la première période (cf. chapitre2).

Les recherches et les développements en IAD & SMA de cette période peuvent être classés en trois grandes catégories :

- les systèmes de simulation. Les systèmes multi-agents modélisent et reproduisent des phénomènes du monde réel, afin de comprendre et d'expliquer leur comportement. C'est le cas par exemple de systèmes de modélisation de phénomènes naturels tels que le ruissellement de l'eau, l'embroussaillement…, de modélisations de systèmes sociaux tels que le phénomène de troc entre les agriculteurs et les chasseurs au Cameroun ou la gestion de l'arrosage de parcelles de terrain…
- les applications modélisant ou intégrant un utilisateur humain dans le système. Les agents du système multi-agent sont les représentants des utilisateurs comme dans les applications de commerce électronique où les agents représentent les clients et les fournisseurs de services. En général les agents utilisés dans ce type d'applications sont des agents dits cognitifs.
- la résolution de problèmes dont la solution globale est apportée par le comportement du collectif. L'acheminement d'informations dans un système de routage tel un réseau téléphonique ou la gestion d'agendas, sont des problèmes typiques de cette catégorie dans lesquels les agents peuvent être cognitifs ou réactifs.

Nous situons nos travaux dans cette dernière catégorie de systèmes et plus particulièrement nous utilisons l'auto-organisation comme moyen d'adaptation. L'auto-organisation est définie en biologie comme une structure ou une fonction au niveau global du système qui émerge des interactions entre les composantes du système [Camazine, 2001]. Une autre définition par le groupe de travail d'AgentLink est la suivante : « l'auto-organisation est le mécanisme ou le processus qui permet à un système de changer son organisation sans contrôle explicite externe durant le fonctionnement du système », (www.irit.fr/TFGSO). En IAD SMA, nous pouvons voir l'auto-organisation comme la transformation de manière autonome de la topologie du système (le système pouvant être vu comme un réseau d'agents) par ses agents [Camps, 1997b]. Un système auto-organisé est un système dans lequel une collection d'éléments en interaction produit un comportement que les éléments de manière individuelle ne sont pas capables de produire.

Les recherches sur les simulations de comportements de sociétés animales, de phénomènes naturels… Drogoul [Drogoul, 1993], Van Parunak [Parunak, 1997b] …abordent la problématique de l'auto-organisation par le fait que des règles locales sont données aux agents et qu'un observateur au niveau global analyse le résultat du collectif. La différence essentielle avec nos travaux provient du fait que le collectif n'a pas à

résoudre un problème. Ce qui intéresse les éthologues c'est de pouvoir faire varier certains paramètres dans la simulation pour observer leur influence sur le global.

Concernant les travaux sur l'auto-organisation, peu de chercheurs du domaine IAD-SMA s'intéressent à l'utilisation de l'auto-organisation pour la conception de systèmes multi-agents. Durant la période concernée, en France, Rémi Foisel se préoccupe du processus de réorganisation au sein d'un système [Foisel, 1996], [Foisel, 1997]. Les travaux les plus représentatifs sont ceux de Van Dyke Parunak qui propose de concevoir les systèmes artificiels en s'inspirant des systèmes naturels et notamment des insectes sociaux [Parunak, 1997a].

2. La coopération pour la résolution

L'hypothèse forte dans un système multi-agent est qu'il est nécessaire d'avoir un collectif d'agents en interaction pour résoudre la tâche globale. Les interactions entre deux agents peuvent être définies comme toute forme d'action exécutée au sein de la société ayant pour effet de modifier le comportement d'un des deux agents. Elles donnent la possibilité aux agents d'agir ou de réagir par une modification de leur comportement et donc de participer au collectif. Cette participation permet au système d'atteindre un de ses objectifs. Nos travaux concernent donc l'étude du comportement d'un agent au sein d'une société et la définition de son architecture.

Nous ne nous situons pas dans le cadre d'une simulation comportementale mais dans le cadre de la résolution de problèmes ; ainsi nos agents n'ont pas à avoir un comportement contraint par le système réel. Nous adoptons les deux postulats suivants : les agents n'induisent pas intentionnellement les autres en erreur et un acte individuel est guidé par une recherche permanente de coopération avec les autres. Le comportement d'un individu au sein d'un collectif est dépendant de ses attitudes sociales. Ces dernières peuvent être à des degrés divers : la sincérité ou le mensonge, l'altruisme ou l'égoïsme... Nous préconisons la coopération comme attitude sociale dans les systèmes multi-agents et nous définissons les caractéristiques de la coopération idéale comme suit :

$C_{perception}$ - Un signal perçu doit être interprétable par un système coopératif. La compréhension mutuelle n'a pas à être postulée mais doit émerger de l'ajustement mutuel entre le système et son environnement.

$C_{décision}$ - Toute information (un signal interprété) doit avoir des conséquences logiques dans le système. En d'autres termes, une information doit apporter de la nouveauté : une différence avec les informations actuellement mémorisées. Un raisonnement est possible.

C_{action} - Les conclusions du processus de raisonnement (le résultat de la fonction réalisée par le système) doivent être utiles à l'environnement du système coopératif.

Toute autre situation sera jugée comme non coopérative, par exemple lorsque une situation jamais observée survient ou lorsqu'une activité antérieure « erronée » du système entraîne un retour de son environnement. Il est à noter qu'un agent coopératif n'est pas un agent altruiste car il va œuvrer pour atteindre son but individuel.

La coopération a donc été étudiée dans l'équipe au travers de différents stages que j'ai encadrés : DEA de Dotto en 1996, de Morisset, en 1997 et de Topin en 1999 et dans le cadre de la thèse de Valérie Camps. Les résultats sont présentés dans les applications dans les parties 2.3 et 2.4 du chapitre II. L'architecture d'un agent est issue de nos travaux précédents et est présentée dans la partie 2.2 du chapitre II.

2.1. Les situations non coopératives

Un agent doit tant que c'est possible pour lui suivre les règles de coopération c'est-à-dire être coopératif. Malgré cette recherche permanente de coopération par les agents, plusieurs situations non coopératives comme l'ambiguïté, la concurrence, le conflit et l'incompréhension peuvent survenir. Elles se produisent fréquemment parce que les agents n'ont qu'une connaissance incomplète/erronée du monde et des autres. Elles sont qualifiées de non coopératives car elles dégradent l'activité collective de la société, on les appelle aussi des échecs de coopération. Elles peuvent être supprimées ou utilisées grâce à la connaissance possédée par le collectif et par la coopération entre les agents. Chaque acte individuel est alors simultanément guidé par l'objectif propre d'un agent et par sa coopération avec les autres.

Plusieurs exemples peuvent illustrer la notion de coopération pour un agent :

- envoyer spontanément une information aux autres agents, s'il croit qu'elle peut leur être utile,
- accepter de satisfaire une demande explicite si elle n'est pas en contradiction avec son activité courante ou ses objectifs,
- agir sur le monde s'il croit qu'il peut favoriser les autres, même s'il n'en retire pas de satisfaction personnelle.

D'une manière plus générale, un agent a un comportement coopératif :

- s'il peut reconnaître et agir de manière appropriée lorsqu'il constate une situation non coopérative,
- s'il n'agit pas intentionnellement pour en créer.

A chaque instant son comportement coopératif (de son point de vue) est guidé par ses compétences et croyances courantes. Les conditions de non coopération conduisant au processus de réorganisation dérivent immédiatement de la définition de la coopération idéale :

$$\text{Coopération} = C_{\text{perception}} \textbf{ et } C_{\text{décision}} \textbf{ et } C_{\text{action}}$$

Non $C_{\text{perception}}$ (incompréhension, ambiguïté) : un signal perçu par un agent est incompris ou interprété de différentes façons. Dans ce cas, un agent coopératif ne va pas ignorer le signal car il le considère nécessaire à l'activité du système. Il va donc tenter de le transmettre à d'autres agents qu'il estime plus compétents et va se faire aider afin de lever les ambiguïtés,

Non $C_{décision}$ (improductivité) : l'information que l'agent a reçue est déjà connue par lui ou n'a aucune conséquence logique. L'agent coopératif ne peut tirer profit de cette information et va donc chercher d'autres agents qui pourraient en bénéficier.

Non C_{action} (inutilité ou conflit) : l'agent considère, compte tenu des croyances qu'il possède sur ses congénères, que l'action qu'il peut opérer n'est pas bénéfique à autrui. Cette situation englobe les notions de conflit et de concurrence qui sont fréquemment étudiées dans le domaine. Par exemple, un conflit de résultat peut survenir si l'agent aboutit à une conclusion opposée à celle d'un autre, un conflit de ressource peut survenir lorsque deux agents veulent accéder à une ressource limitée et que leurs demandes conjuguées excèdent l'offre... La concurrence de résultat est détectée si l'agent aboutit à des conclusions identiques à celles d'un autre. Face à de telles situations, l'agent doit agir de manière pertinente, en fonction du contexte, pour revenir à un état coopératif. Par exemple, dans le cas d'un conflit de ressource, les agents impliqués devront relâcher des contraintes et tenter de se répartir au mieux la ressource

L'agent réalise en permanence sa fonction partielle, mais il agit simultanément sur l'organisation interne du système s'il détecte des situations non coopératives. La conception d'un système multi-agent coopératif demande de définir pour chaque composant ou agent pris isolément, tous les états non coopératifs et les activités associées pour les supprimer. Quand le système est plongé dans un environnement dynamique, l'observateur peut analyser un processus interne au système conduisant à la modification des relations entre les agents. Ainsi, la recomposition des fonctions partielles réalisées par chaque agent amène une transformation de la fonction globale du système et les états non coopératifs dus aux situations imprévues sont progressivement supprimés. L'organisation d'un système est décrite par les liens d'interaction entre agents. Dans notre approche, un agent autonome considère qu'il a trouvé la bonne place au sein de l'organisation s'il interagit coopérativement avec autrui ; dans le cas contraire, il agira pour chercher une place plus adaptée.

2.2. Les composantes d'un agent coopératif

L'organisation qui émerge au cours de la résolution est une organisation observable non prédonnée par le concepteur du système. Mais ce qui nous intéresse le plus c'est l'émergence de la fonction du système qui est produite par l'organisation entre les agents à un instant donné. Pour réaliser cela, les agents sont programmés pour être en situation coopérative avec les autres agents du système. Il est important que l'agent puisse décider de manière locale s'il est en situation coopérative. En effet, par rapport aux croyances qu'il a sur lui-même, il peut localement déterminer si ce qu'il reçoit est compréhensible et lui permet de réaliser une action. De la même manière, en fonction de ses perceptions, il pourra juger localement si ses actions ont été utiles. D'une manière générale, cinq parties sont indispensables à un agent coopératif pour qu'un comportement collectif cohérent puisse être observé à partir de l'agrégation de comportements individuels.

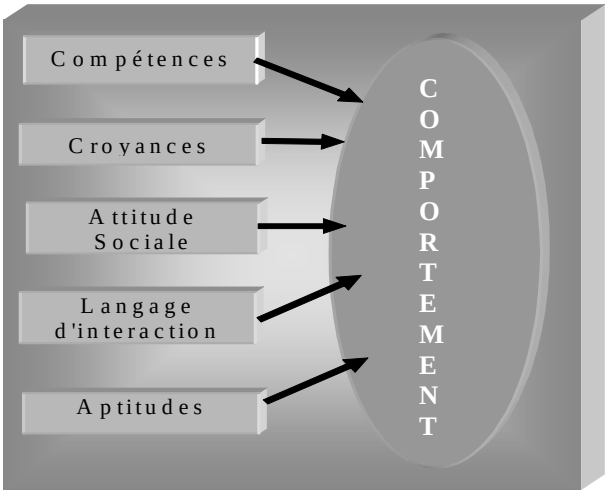


Figure 9 - Composantes d'un agent

Le premier module concerne les compétences définies comme des connaissances d'un domaine particulier qui permettent à l'agent de réaliser la fonction partielle qui lui est assignée. Aucune contrainte technique n'est imposée pour le développement (système de production, méthode objet,…).

Le deuxième module correspond à la représentation de lui-même, des autres agents et de l'environnement. Il confère à l'agent une croyance sur ce qu'il sait de lui-même, des autres et de son environnement. Les croyances peuvent être implicites ou explicites.

L'attitude sociale appelée coopération se situe dans le troisième module. Elle permet de définir des critères locaux qui vont permettre à l'agent de décider de son comportement et de se réorganiser avec les autres agents en modifiant ses liens avec les autres agents comme cela a été présenté dans la partie 2.1 du chapitre II.

Le quatrième module est relatif au langage d'interaction qui permet à l'agent de communiquer soit directement par envoi de messages soit indirectement par l'environnement.

Les aptitudes sont regroupées dans le cinquième module. Elles correspondent aux capacités que l'agent possède pour raisonner sur ses représentations et sa connaissance.

Ce travail donne un modèle de l'agent, il spécifie l'architecture des agents à développer. Ainsi, notre travail de concepteur nous amène à avoir une approche de conception ascendante. Les applications traitées dans cette période (décrites ci-dessous) sont caractérisées par le fait que les agents du système à concevoir sont des entités physiques distribuées (les robots dans le Tileworld, les fourmis dans ANTS…) et la conception ascendante est incontournable. La coopération est un moyen de passer de la conception des entités au système global et d'obtenir de bonnes performances globales.

2.3. Application au Tileworld (1996)

Le Tileworld est une application développée par les doctorantes Christine Piquemal-Baluard, Valérie Camps, ainsi que par Pierre Glize et moi-même en 1996 [Camps, 1996]. Cette application est présentée dans ce mémoire car elle a permis de montrer que la coopération est un moyen efficace pour résoudre un problème par un système composé d'agents. La présentation est axée sur la description des situations non coopératives qu'un agent peut détecter localement.

2.3.1. Objectifs

L'objet de ce travail était de montrer que des agents coopératifs pouvaient simultanément satisfaire la tâche individuelle qui leur était assignée tout en ayant une activité collective meilleure que celle d'agents individualistes. Les agents de la société considérée ont des compétences homogènes.. Nous avons choisi comme domaine d'application le Tileworld car ce jeu a précédemment été étudié [Pollack, 1990], [Goldman, 1994] et, par conséquent, nos résultats peuvent être comparés avec ceux déjà obtenus.

2.3.2. Description du Tileworld

Les agents se déplacent verticalement ou horizontalement sur un damier 10*10 sur lequel se trouvent des pavés, des trous et des obstacles. Le but du jeu est de combler si possible tous les trous à l'aide de pavés. L'objectif premier de chaque agent est de pousser le pavé le plus proche dans le trou le plus proche. La notion de proximité n'est autre qu'un calcul de distance euclidienne qui ne tient pas compte des obstacles, des autres agents et des pavés. Les agents se déplacent ensuite en contournant les obstacles. Tous les agents possèdent les mêmes compétences et ont des croyances sur les autres : ils croient que les autres ont les mêmes compétences qu'eux. L'interaction s'effectue par des perceptions et des actions sur le monde. Chaque agent observe le résultat du raisonnement individuel des autres agents à partir de ses croyances avant de se déplacer. Il présuppose ainsi le comportement des autres agents. La transmission de messages entre agents étant impossible, les agents ne peuvent donc ni négocier lorsque des conflits surviennent, ni informer les voisins en cas de concurrence ou d'ambiguïté.

Un agent perçoit tous les événements qui peuvent survenir sur la grille, il connaît, par conséquent l'existence de tous les autres agents. Il va donc améliorer son comportement en tenant compte de celui des autres.

Comme dans la simulation de Goldman, les agents implémentés ont une stratégie identique. Le fonctionnement de base d'un agent est le suivant :

- il se dirige vers le pavé le plus proche,
- il pousse alors ce pavé vers le trou le plus proche.

2.3.3. Détection et traitement des situations non coopératives

Étant donné l'aspect distribué du système multi-agent, un agent individuel (avec éventuellement l'aide des autres) doit avoir la capacité de détecter et de traiter ces situations. Pour cela, les agents sont dotés d'attitudes sociales qui sont des comportements indépendants du domaine d'application. Elles leur permettent d'agir en toutes circonstances de manière coopérative.

Dans les paragraphes suivants, trois situations non coopératives sont détaillées : le conflit, l'ambiguïté et la concurrence. Nous ne pouvons pas exhiber des comportements dans le cas d'incompréhension pour le domaine d'application choisi car tous les agents connaissent parfaitement le monde. Nous présentons comment un agent détecte chaque situation non coopérative, puis la manière de l'instancier dans le domaine d'application du Tileworld.

La situation non coopérative d'ambiguïté

L'ambiguïté est formulée ainsi : un agent est à la même distance de deux pavés ou de deux trous.

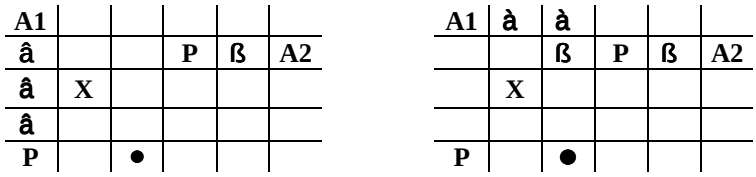


Figure 10 - Exemple de situation de coordination

Légende : P pavé ● trou → sens de déplacement
X obstacle Ai agent i

Dans la figure 10, l'agent A1 a deux pavés qui sont à la même distance. Il détecte donc une ambiguïté. La grille de gauche indique le comportement d'agents individualistes qui ne se soucient pas des situations ambiguës. Comme ils ont des stratégies d'action identiques, ils vont choisir arbitrairement de se diriger vers le premier pavé le plus proche. Ils sont alors amenés à pousser deux pavés dans le même trou ce qui provoque un conflit de trou.

La grille de droite montre le comportement d'agents coopératifs dans la même situation. A1 va résoudre son ambiguïté en choisissant le pavé partagé avec A2 pour coopérer avec ce dernier et travailler plus efficacement. L'agent A2 va alors pousser le pavé pour l'amener dans une position telle que l'agent A1 n'a plus qu'à le pousser directement vers le trou.

La résolution de l'ambiguïté consiste à choisir avant tout les pavés convoités, c'est-à-dire ceux qui constituent les objectifs des autres, ceci afin de coopérer.

La situation non coopérative de concurrence

La concurrence se traduit par : deux agents sont à la même distance d'un même pavé.

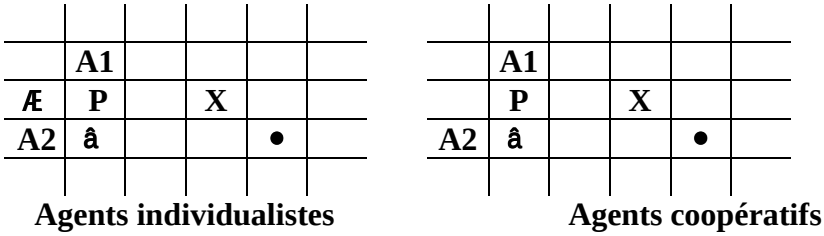


Figure 11 - Résolution d'une situation de concurrence.

Dans la figure 11, les agents A1 et A2 sont en concurrence : ils veulent pousser le même pavé dans le même trou.

Lorsque les agents sont individualistes, ils vont choisir de se diriger vers le pavé le plus proche d'eux, en l'occurrence ici l'unique pavé. Ils vont ainsi être amenés à vouloir pousser le même pavé dans le même trou. Cela conduit à de nombreux déplacements inutiles voire des gênes mutuelles.

Lorsque les agents sont coopératifs et sont tous deux en situation de concurrence, ils vont tenter de s'aider réciproquement en coopérant. Pour cela A1 va amener le pavé dans une position telle que A2 n'a plus qu'à le pousser directement vers le trou.

La résolution de la concurrence consiste à se coordonner pour atteindre le but plus efficacement.

La situation non coopérative de conflit

On distingue trois types de conflits dans le domaine du Tileworld.

1- Les conflits de trou : deux agents désirent amener deux pavés différents dans un même trou.

2- Les conflits de case : deux agents prévoient de se déplacer sur la même case.

3- Les conflits de case et de trou : deux agents prévoient de se déplacer sur la même case afin d'amener deux pavés différents dans un même trou.

Lorsque les agents sont individualistes et ont des stratégies d'action identiques, ils vont choisir de se diriger vers le pavé le plus proche d'eux et d'amener ce dernier dans le trou le plus près. Ils vont ainsi être amenés à vouloir pousser leur pavé respectif dans le même trou. Comme dans la situation de concurrence, vue précédemment, cela conduit à de nombreux déplacements inutiles et parfois même à des gênes mutuelles.

Lorsque les agents sont coopératifs et tous deux en situation de conflit, ils vont tenter de s'aider réciproquement. Pour cela A1 et A2 vont tout d'abord essayer de changer de trou : impossible car, dans le cas présent, le trou est unique. Ils essaient alors de changer de pavé. A2 ne peut pas, A1, lui, détecte une ambiguïté. Comme dans la figure 10, A1 va choisir le pavé qui est convoité par A2. Il choisit donc parmi les plans qu'il peut appliquer, celui qui le coordonne avec l'agent A2.

La résolution de conflit se résume à deux actions : éviter le conflit, ce qui signifie changer d'objectif (trou, pavé ou chemin) ou ignorer le conflit ce qui signifie laisser l'agent le plus rapide exécuter son plan. Ceci est possible car les agents ayant tous des compétences identiques, ils savent décider si un autre agent est plus efficace en nombre de déplacements que lui-même pour pousser un pavé.

2.3.4. Résultats et analyse

Afin d'étudier les avantages de la coopération, nous avons calculé le nombre d'actions réalisées par les agents en tenant compte de toutes les situations non coopératives. Ce processus de raisonnement, basé sur les croyances, est totalement distribué à l'intérieur des agents. Il n'y a pas d'analyse globale du travail.

La durée d'une session est mesurée en nombre de cycles. Un cycle est l'unité minimale de temps pendant laquelle plusieurs agents peuvent se déplacer en parallèle d'une case à une autre. Tous les agents ayant la même vitesse de mouvement, l'application considère des agents synchrones : à chaque cycle, chaque agent effectue un déplacement élémentaire (s'il est possible compte tenu des contraintes de l'environnement). La session est terminée lorsqu'il ne reste plus de pavé ou de trou, ou lorsque le temps alloué à la session est terminé. Le nombre total de déplacements effectués ainsi que le nombre de cycles sont calculés au fur et à mesure de la session.

Nous avons étudié notre méthode sur 5 scénarios.

- Scénario individualiste. Le niveau de coopération de chaque agent est 0. Un agent est seulement intéressé par ses objectifs individuels. Ceci est possible parce qu'un agent peut faire seul tous les travaux pour terminer la session.
- Scénario non ambigu. Un agent n'est capable de traiter que les ambiguïtés.
- Scénario non concurrent. Un agent n'est capable de traiter que les situations de concurrence.
- Scénario non conflictuel. Un agent n'est capable de traiter que les conflits.
- Scénario coopération totale. Le niveau de coopération de chaque agent est optimal. Chaque agent est capable de considérer et de résoudre l'ambiguïté, la concurrence, et les situations de conflit. En fait, les agents essaient, s'ils le peuvent, d'aider ou d'être aidés par le collectif.

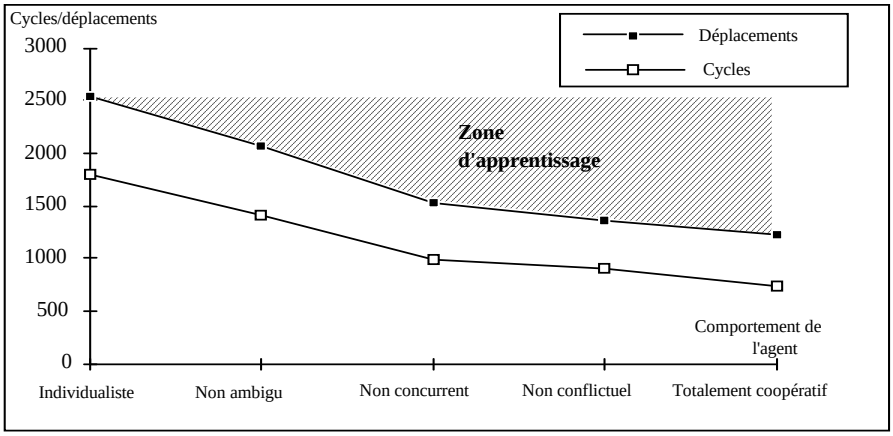


Figure 12 - Nombre de cycles/déplacements selon le comportement de l'agent

Nous avons tout d'abord remarqué que la terminaison a toujours été obtenue avec des stratégies de coopération totale, ce qui n'est pas le cas avec des attitudes de coopération partielle et a fortiori lorsque les agents sont totalement individualistes.

Dans toutes les sessions, nous avons également noté que la coopération totale (résolvant successivement l'ambiguïté, la concurrence et le conflit) donnait de meilleurs résultats en nombre de déplacements et de cycles que les agents individuels (Cf. courbes de la figure 12).

Les résultats obtenus montrent aussi qu'une attitude coopérative basée sur l'aide mutuelle n'est intéressante que si les agents sont capables de se coordonner. Les résultats

sont alors meilleurs que les résultats obtenus avec les agents individuels. La caractéristique générale qui émane de ces sessions et mise en évidence dans les courbes de la figure 12 est : plus les agents sont coopératifs, meilleure est l'organisation de la société et plus les agents sont capables de faire face à des situations imprévues donc de s'adapter. La coopération est donc une bonne heuristique pour la résolution de problèmes.

La courbe des cycles indique clairement que la coopération peut s'adapter à un raisonnement totalement distribué chez les agents, car les agents ont une activité parallèle. Si ce n'était pas le cas, ils auraient fréquemment des interblocages dans leur déplacement et le nombre de cycles ne serait pas aussi faible.

Les agents n'étaient pas "programmés" pour le but collectif : c'est une fonction émergente, car non réductible à la dynamique individuelle. Cette application du Tileworld illustre donc tout le profit que l'on pouvait tirer d'attitudes coopératives pour résoudre et pour augmenter la performance globale d'un système. Dans la même période, les travaux de Sekaran et Sen [Sekaran, 1995] mettent en évidence le fait que si les agents coopèrent, la performance générale du système est meilleure. Leur but est d'étudier si, dans une situation particulière, un agent doit accepter ou refuser une demande de coopération. Leur système est basé sur le concept de comportement réciproque.

2.4. ANTS (1997 - 1998)

ANTS est un projet GIS Science de la Cognition au niveau national dont les partenaires sont l'IRIT, le LECA[8] de l'Université Paul Sabatier, et a donné lieu à deux projets financés par la région Midi-Pyrénées en 1997 et 1998. De nombreux stagiaires ont participé à l'élaboration de la plate-forme tels que les IUT Nicolas Salle en 1997 [Salle, 1997], Brice Kuhn en 1998 [Kuhn, 1998] et David Pelletier-Proulx en 1999 [Pelletier-Proulx, 1999]. Benoît Morisset en 1997 (DEA) [Morisset, 1997] et Xavier Topin (stage IUP en 1998 et DEA en 1999) [Topin, 1998], [Topin, 1999a] ont plus particulièrement étudié et mis en œuvre la coopération. Les permanents ayant en charge ce projet, étaient pour l'IRIT Pierre Glize, Christine Régis et moi-même et pour le LECA Vincent Fourcassié et Guy Théraulaz.

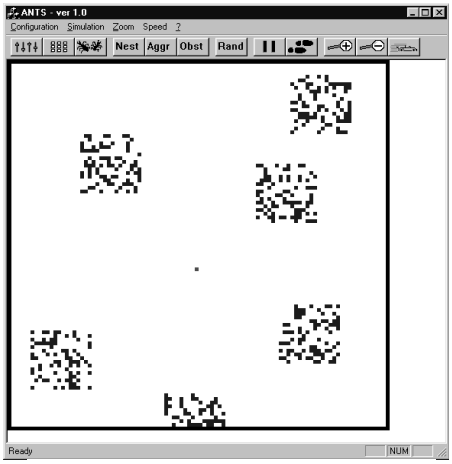


Figure 13 - La plate-forme de simulation d'ANTS

Dans ce mémoire, le projet ANTS [Topin, 1999b] est décrit pour illustrer les différentes composantes de l'architecture d'un agent et les situations non coopératives. Les résultats obtenus ont été très importants car ils ont renforcé l'idée que la coopération était une heuristique de résolution permettant d'optimiser les résultats et permettait de s'adapter à un environnement évolutif.

[8] LECA Laboratoire d'Ethologie et de Cognition Animale

2.4.1. Objectifs

L'objet de ce projet était de fournir une plate-forme de simulation du fourragement de fourmis c'est-à-dire concevoir un collectif de fourmis logicielles chargées d'explorer un environnement et de collecter des ressources. L'objectif pour les éthologues était de disposer d'un outil de travail pour expérimenter le modèle de fourmi oeccophylle sur lequel ils travaillaient. L'objectif pour notre équipe était de mettre en œuvre des robots-fourmis ayant un comportement coopératif et de pouvoir comparer les résultats de ce collectif avec les résultats des fourmis ayant le même comportement que les fourmis oecophylles naturelles. Le comportement des robots-fourmis s'inspire du comportement des fourmis oecophylles tout en étant différent, dans le sens où il n'est pas fidèle au comportement des fourmis naturelles observé par les éthologues.

C'est pourquoi le système de robots-fourmis développé n'est pas une simulation de fourmis en train de fourrager mais un système de robots-fourmis autonomes qui s'adaptent à un environnement dynamique, pour exploiter des ressources.

2.4.2. Description de la plate-forme ANTS

Pour observer et faire des mesures sur le comportement des fourmis oecophylles, un environnement de simulation a été développé (voir figure 13). Il est constitué d'une arène de taille variable entre 1 et 10000 pixels qui peut éventuellement contenir des obstacles, c'est-à-dire des zones inaccessibles par les fourmis.

Des agrégats de nourriture sont matérialisés par des amas de points sur la figure 13. Un agrégat a une certaine densité et est constitué d'unités de nourritures où chaque unité de nourriture est représentée par un pixel. Les agrégats peuvent être disposés de manière aléatoire ou bien par l'utilisateur de la plate-forme. L'arène comporte au minimum un nid de fourmi, et chaque fourmi a une taille d'un pixel.

2.4.3. Un agent coopératif dans ANTS

Un robot-fourmi a comme *compétences* celles qui correspondent aux aptitudes de base des fourmis appartenant à l'espèce des oecophylles et à leurs caractéristiques physiologiques. Ainsi, un robot-fourmi perçoit les obstacles, les ressources, les autres robots-fourmis et la phéromone. Il est capable de se déplacer, de prélever des ressources, de retourner à son nid, de se reposer et de retourner sur le dernier lieu où il a trouvé des ressources.

Son seul *moyen de communication* consiste à déposer de la phéromone.

Chaque robot-fourmi a une *représentation (ou croyance)* implicite sur les autres robots-fourmis appartenant au même collectif : il sait que les autres robots-fourmis sont comme lui (même compétence, même aptitude, mêmes croyances, même attitude sociale, même langage de communication).

L'aptitude d'un robot-fourmi est codée dans le raisonnement dont le concepteur l'a doté. Elle correspond au comportement d'une fourmi oeccophylle qui a été observé par les éthologues augmenté de l'attitude coopérative qui consiste à être coopératif.

Le comportement d'une fourmi oeccophylle durant la phase de fourragement consiste à passer par différents états tels que :

- le déplacement. Il consiste à choisir une des huit directions autour de la position courante de la fourmi.
- l'exploitation. Elle consiste à prélever de la nourriture quand la fourmi est positionnée sur un item de nourriture
- le repos. Une fourmi peut fourrager pendant un certain temps. Ensuite elle rentre au nid pour se reposer.
- le retour au nid. Une fourmi a une mémoire de la position de son nid pour pouvoir y revenir.
- le retour à la dernière source de nourriture. Une fourmi retournant au nid chargée repart explorer vers le dernier lieu où elle avait trouvé de la nourriture.

La modélisation du comportement de la fourmi consiste à choisir un nouvel état possible en fonction de l'état dans lequel se trouve la fourmi et de sa perception de l'environnement. Au début de la simulation les fourmis sortent du nid et passent en mode exploration, pour tenter de trouver de la nourriture. Elles sont alors attirées par la nourriture, si celle-ci est dans leur champ de vision. Si elles perçoivent de la phéromone, elles auront tendance à suivre cette piste de phéromone. L'algorithme du déplacement d'une fourmi a été développé dans le cadre d'un stage de DEA par Xavier Topin [Topin, 1998].

Si elles sont sur une unité de nourriture, alors elles vont l'exploiter, c'est-à-dire la prélever, puis continuer à explorer. Une fois la limite de nourriture transportable atteinte, la fourmi rentre au nid en laissant derrière elle une substance chimique, la phéromone, afin de relier le nid à la dernière source de nourriture trouvée. Une unité de nourriture se trouve rarement isolée, donc quand une fourmi chargée rentre au nid, il restait certainement de la nourriture à proximité. La phéromone permet ensuite aux fourmis suivantes d'aller vers cette source de nourriture découverte et ainsi de continuer à l'exploiter.

Ces caractéristiques font qu'une fourmi a une perception très réduite de l'ensemble de l'environnement, qu'elle peut perdre une piste de phéromone si la densité est faible et qu'elle peut aussi oublier de prélever la totalité d'un agrégat de nourriture lorsque les unités de nourriture qui restent dans l'environnement, sont légèrement dispersées.

L'*attitude sociale* du robot-fourmi est décrite dans la section suivante.

2.4.4. Détection et traitement des situations non coopératives dans ANTS

Etre coopératif pour un robot-fourmi signifie avoir toujours une attitude sociale coopérative. Dans un environnement dynamique et imprévu, les situations non coopératives sont très fréquentes, ceci implique que le robot-fourmi sera très souvent en situation non coopérative. Le comportement du robot-fourmi lui permet de revenir à des interactions coopératives. En fonction de ses croyances sur les autres, de ses compétences, de ses moyens de communication, de son attitude sociale, de ses aptitudes et de ses perceptions des autres et de l'environnement "physique simulé", il essaye de revenir vers des interactions coopératives, s'il détecte des situations non coopératives.

Tous nos efforts de spécification sont donc basés sur la définition de ce qui peut être une situation non coopérative pour un robot-fourmi et comment agir ensuite pour revenir à un état coopératif. Cette détection se fait de manière locale au robot-fourmi en fonction de

ses capacités de perception de l'environnement. L'action spécifiée pour réagir à cette situation dépend aussi des aptitudes et des compétences du robot-fourmi. Pour chacune des situations décrites ci-dessous, nous donnons en parallèle l'action qu'une fourmi oeccophylle entreprendrait dans des conditions identiques.

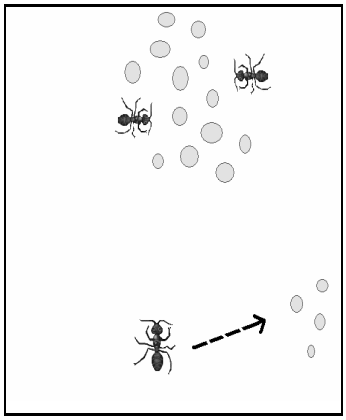


Figure 14 - Concurrence

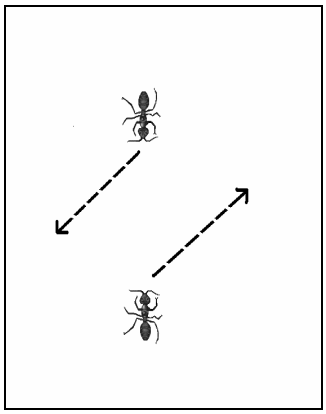


Figure 15 - Inutilité

Cas de concurrence n°1

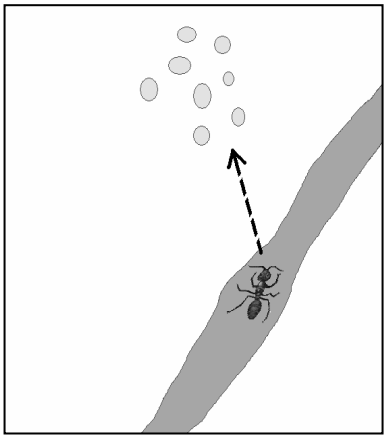
Un cas de non coopération survient lorsqu'une fourmi voit une ressource non exploitée et une ressource exploitée. Une fourmi oeccophylle est attirée proportionnellement par la ressource, et ira plutôt là où il y a le plus de ressources. Pour le robot-fourmi cette situation correspond à une détection de concurrence qui en l'occurrence est une situation non coopérative de son point de vue. En général, deux agents sont en concurrence s'ils veulent obtenir un même état du monde : ici, deux robots-fourmis veulent ramener la même quantité de ressources au nid. Le robot-fourmi a une représentation des autres robots-fourmis. En effet, un robot-fourmi sait implicitement que tous les robots-fourmis du système ont les mêmes comportements que lui. Il sait donc que les robots-fourmis qu'il perçoit proche de ressources ont de grandes chances de prélever ces ressources. Aussi pour ne pas être en situation de concurrence, le robot-fourmi sera d'autant moins attiré par une ressource qu'elle est exploitée par un nombre de congénères important (figure 14).

Cas de concurrence n°2

Un deuxième cas de non coopération est détecté lorsqu'une fourmi en mode exploration voit une autre fourmi. Une fourmi oeccophylle ira indifféremment vers la fourmi ou ailleurs. Pour un robot-fourmi, il se trouve dans une situation où il rencontre un autre robot-fourmi qui ne lui transmet pas d'information (pas de phéromone). Donc, s'il se dirige vers ce robot-fourmi, il entrera en situation de concurrence, car il visitera une zone déjà couverte par l'autre robot-fourmi. Il évite donc cette zone et il n'ira pas dans la direction du robot-fourmi perçu (figure 15).

Cas de concurrence n°3

Une autre situation non coopérative survient quand une fourmi qui suit sa piste de phéromone voit des ressources. Une fourmi oeccophylle est attirée proportionnellement par la phéromone et la ressource, elle a donc plus de chance de suivre la piste, qui comporte a priori une grande concentration de phéromone. Un robot-fourmi risque de se retrouver en situation de concurrence s'il se dirige vers des ressources exploitées par d'autres agents. Pour éviter cette concurrence potentielle, il ira s'il le peut vers des ressources nouvelles, même si la phéromone est présente en grande quantité (figure 16).



Cas de coopération n°1

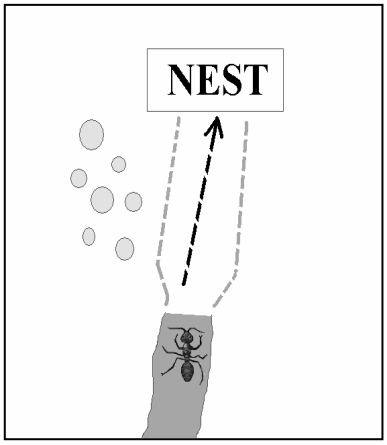


Figure 17 - Communication spontanée

Parce qu'un robot-fourmi est coopératif, il donne spontanément des informations aux autres robots-fourmis s'il croit que ces informations peuvent leur être utiles. C'est ce que l'on appelle la communication spontanée. Ainsi, une fourmi qui se trouve dans la situation suivante : elle rentre au nid et elle voit en passant des ressources, détient une information sur la localisation de ressources. Une fourmi oeccophylle ne prend pas en compte l'information liée à la perception de cette ressource. Par contre, le robot-fourmi continue son chemin vers le nid en déposant plus de phéromone dans le but de donner des informations plus exactes sur l'environnement tel qu'il le perçoit (figure 17).

Cas de coopération n°2

Une fourmi oeccophylle dépose de la phéromone en retournant au nid quand elle est chargée. Donc une fourmi qui retourne au nid en n'étant pas chargée au maximum dépose aussi de la phéromone. Le fait qu'elle ne soit pas chargée au maximum signifie qu'elle a récolté les dernières ressources là où elle était et qu'elle a écoulé son temps maximum autorisé hors du nid. Le robot-fourmi par souci de bien informer ces partenaires ne laisse pas de trace de phéromone. Ici aussi, l'agent dispose d'une information supplémentaire : il n'y a plus de ressource là où il se trouve. Pour coopérer, il communique cela, en ne déposant pas de phéromone. Si par la suite l'agent découvre des ressources, il déposera à nouveau de la phéromone.

Cas de coopération n°3

Une fourmi oeccophylle qui retourne au nid en ayant trouvé des ressources dans l'environnement, recrute d'autres fourmis. Elle recrute en fonction de la quantité de nourriture ramenée au nid. Toujours par souci d'informer de manière la plus juste possible, le robot-fourmi recrute en fonction de la dernière quantité de phéromone déposée. En effet, il dispose d'une information supplémentaire : la somme des ressources détectées pendant le trajet. Ainsi, le robot-fourmi donne une représentation plus exacte de l'environnement que celle donnée par la fourmi oeccophylle.

2.4.5. Résultats et analyse

De nombreuses mesures ont été réalisées, nous présentons ici un type d'environnement dans la figure 18 où 2500 ressources ont été réparties aléatoirement dans l'arène dans 5 agrégats. Dans la figure 19, 2500 ressources ont été réparties aléatoirement dans l'arène dans 100 agrégats. Les agrégats ont été positionnés au hasard dans l'aire de fourragement mais la densité de ressources par agrégats est la même dans les deux types d'environnements. Le nombre de ressources dans l'arène est fixe au départ, il diminue par le fait que les fourmis prélèvent les ressources. L'arène a une dimension de 500*500 pixels.

Chaque simulation est réalisée pour 10000 cycles et l'efficacité de la collecte de ressource est calculée en comptant tous les 100 cycles le nombre cumulé de ressources ramenées au nid. Pour chaque ensemble de paramètres, le nombre cumulé de ressources ramenées au nid est la moyenne de 50 simulations. Pour chaque nouvelle simulation, un nouvel environnement de départ est créé toujours avec les mêmes contraintes, soient des agrégats de nourriture et des obstacles identiques en taille et positionnés aux mêmes endroits dans l'arène.

Les résultats des robots-fourmis sont représentés par les courbes ayant les traits les plus épais. Pour montrer les capacités d'adaptation de ces entités logicielles, les mesures ont été effectuées avec des environnements différents. Le comportement de ces robots-fourmis avec des espèces des fourmis oecophylles ayant des caractéristiques physiologiques différentes : les fourmis oecophylles 1 et 2 sur les courbes ci-dessous. Les caractéristiques physiologiques des robots-fourmis ont été modifiées en conséquence mais pas leur comportement.

Les robots-fourmis ont des résultats meilleurs ou aussi bons que les fourmis oecophylles. Ceci est obtenu par une évaluation directe et locale de l'environnement par les robots-fourmis. Quand ils trouvent au moins un comportement coopératif à avoir, ils l'adoptent, s'il y en a plusieurs ils en choisissent un au hasard.

Les robots-fourmis de par leur comportement coopératif ont tendance à explorer de manière plus optimale l'environnement. Ils se repoussent, ce qui a pour conséquence de créer un meilleur " éparpillement " dans les premières secondes, et ainsi une meilleure occupation de l'espace, pour trouver les ressources plus rapidement.

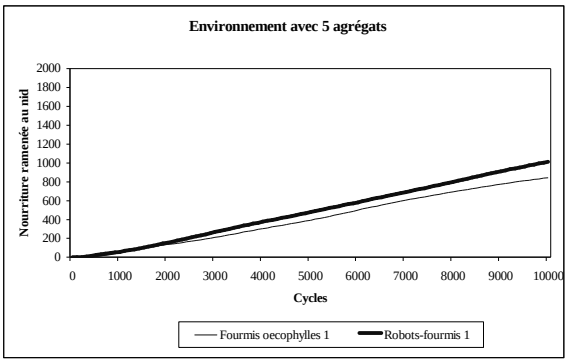


Figure 18 - Fourmis oecophylles 1 et robots-fourmis 1 dans un environnement de 5 agrégats

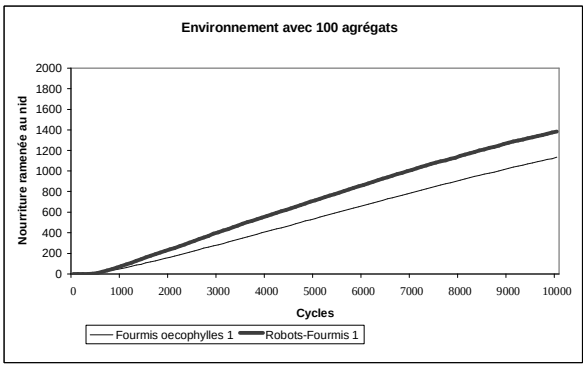


Figure 19 - Fourmis oecophylles 2 robots-fourmis 1 dans un environnement de 100 agrégats

2.5. Contributions

Nos résultats montrent que les systèmes coopératifs sont efficaces pour résoudre des problèmes et que la coopération est une heuristique de résolution. Au début la coopération était classique comme le partage de tâches ou de résultats (communication spontanée) entre des agents et était utilisée comme moyen de résolution par exemple dans Télémac. C'est la coopération classique dans un système multi-agent telle que Ferber [Ferber, 1995] la définit : il y a coopération entre agents si ils n'ont pas des buts incompatibles et s'il est nécessaire qu'ils partagent des compétences ou des ressources. Le résultat de l'étude de cette notion au cours de nos travaux a abouti à une définition plus large et plus générique de ce qu'est la coopération entre les agents. On parle alors d'attitude sociale de coopération qui englobe la coopération au sens classique mais aussi le traitement de situations non coopératives. Dans cette période, la coopération a été clairement spécifiée et ce toujours dans le but de concevoir des systèmes fonctionnellement adéquats.

Le traitement des situations non coopératives par les agents en cours de résolution permet de faire émerger la fonction globale du système. Dans l'application ANTS par exemple, la fonction globale du système émergente est l'optimisation du fourragement. Le concepteur n'a nullement programmé dans les fourmis la manière d'optimiser le fourragement du collectif. Ce résultat est extrêmement important car ainsi la coopération devient le moteur du comportement des agents.

Les systèmes construits sont ouverts dans le sens où l'ajout ou le retrait d'un agent est directement pris en compte par les facultés de perception des agents.

Les agents que nous avons modélisés dans le Tileworld sont adaptatifs mais non auto-organisateurs : ils ne modifient ni leurs compétences, ni leurs croyances, ni leurs attitudes sociales. De même, les robots-fourmis n'apprennent rien c'est-à-dire qu'ils ne modifient pas leurs croyances ni leurs compétences. Ils sont adaptatifs dans le sens où sans avoir programmé de manière exhaustive toutes les situations auxquelles ils pourront être

confrontés, ils décident d'un comportement à avoir en fonction d'un certain état de leur environnement et de leur attitude coopérative. La dynamique de l'environnement d'un agent est réalisée par le fait que les agents se déplacent, que les ressources sont prélevées. Seule l'attitude sociale coopérative permet aux agents de modifier leurs interactions dans un environnement dynamique. Ceci entraîne l'adaptation du système à l'environnement dynamique.

La capacité d'adaptation des agents n'étant que momentanée et non mémorisée, la suite de nos travaux doit donc concerner l'apprentissage de l'organisation. Il doit être décidé de manière autonome dans un système multi-agent coopératif avec un principe et une méthode précise. Le principe est le suivant : chaque agent doit se positionner dans l'organisation de manière à être en situation permanente d'interaction coopérative avec les autres membres du collectif.

En parallèle avec les travaux sur la coopération, notre préoccupation scientifique restait l'étude des systèmes complexes. La rencontre avec des éthologues et l'étude de systèmes naturels a conforté notre idée qu'un système est plongé dans un environnement et que son activité est influencée par ce dernier mais aussi l'activité du système influence l'évolution de l'environnement. On parle alors de couplage structurel entre un système et son environnement au sens de Maturana et Varela [Maturana, 1994]. C'est pourquoi en parallèle avec nos travaux et nos expérimentations nous avons cherché à élaborer une théorie des systèmes complexes dans laquelle l'environnement a un rôle fondamental.

3. La coopération pour l'adaptation

Avec l'avènement du Web, la puissance de calcul des machines toujours grandissante, les applications à développer se complexifient et les concepteurs ont à répondre à de nouveaux besoins ou de nouvelles contraintes tels que :

- L'environnement du système est dynamique, rendant inopérant l'énumération exhaustive des situations que le système rencontrera.
- Le système est ouvert et donc dynamique car constitué d'un nombre variable de composants.
- La tâche à réaliser par le système est tellement complexe qu'il est impossible de garantir une conception parfaite.
- La manière de réaliser la tâche assignée au système est difficile, voire impossible, à appréhender dans sa globalité par le concepteur.

Que des systèmes artificiels puissent faire face à des situations réellement imprévues implique un axe de recherche aboutissant à une méthode de conception de systèmes différente de l'approche globale descendante traditionnelle. Notre méthode de conception est ascendante et émergente. C'est pourquoi nos travaux se sont orientés vers l'étude des théories de l'émergence.

Pour faire face à ces difficultés, une des possibilités consiste à laisser plus d'autonomie aux logiciels afin qu'ils s'adaptent à de nouveaux environnement, à de nouvelles stimulations.

3.1. Le théorème de l'adéquation fonctionnelle

L'adéquation fonctionnelle d'un système est un jugement effectué par un observateur sur la pertinence de son activité dans l'environnement. Ceci nous a amené à étudier les interactions entre un système et son environnement Trois types d'interaction ont été recensés comme dans les travaux de Galliers [Galliers, 1988] et Ferber [Ferber, 1995] soient :

- les interactions coopératives,
- les interactions antinomiques,
- les interactions indifférentes ou neutres.

Une interaction est aninomique si l'action d'un des agents empêche l'autre d'atteindre son but. Une interaction est indifférente si l'action d'un des agents n'a aucune influence sur le fait que l'autre agent atteigne son but. Une interaction est coopérative si l'action de l'un des agents est bénéfique à l'autre pour atteindre son but.

A partir de ces interactions nous avons défini trois familles de systèmes :

- les systèmes fonctionnellement adéquats,
- les systèmes coopératifs,
- les systèmes à milieu intérieur coopératif.

Un système est fonctionnellement adéquat si un observateur-oracle juge qu'il réalise une activité pertinente dans l'environnement dans lequel il est situé.

Un système coopératif est un système dont toutes les interactions avec son environnement sont uniquement coopératives. Elles ne sont donc ni indifférentes, ni antinomiques.

Un système à milieu intérieur coopératif est un système dont toutes les entités dont il est composé, sont uniquement en interactions coopératives.

Ensuite, nous avons proposé le théorème de l'adéquation fonctionnelle qui est à la base de la théorie des AMAS.

Théorème. Pour tout système fonctionnellement adéquat, il existe au moins un système à milieu intérieur coopératif qui réalise une fonction équivalente dans le même environnement.

La démonstration de ce théorème [Camps 1998], se déduit de l'application de l'axiome et des quatre lemmes présentés dans le tableau 1 par des opérations de surjection et d'inclusion d'ensembles.

Cette théorie ainsi que tous les concepts utilisés et les démonstrations sont explicités et détaillés dans le mémoire d'habilitation à diriger des recherches de Pierre Glize [Glize, 2000], je n'en donne dans le tableau suivant que les principaux axiomes et lemmes annotés.

Axiome et Lemmes	Explications
Axiome : Si un système n'a aucune activité antinomique sur son environnement alors il est fonctionnellement adéquat	La véracité de cette assertion ne peut pas être prouvée, car il faudrait un observateur extérieur à l'activité de tous les systèmes évoluant dans un certain univers physique, tout en n'interagissant aucunement avec celui-ci afin de ne pas le perturber.
Tout système coopératif est fonctionnellement adéquat.	La démonstration s'appuie sur l'axiome précédent, car par définition un système coopératif n'a pas d'activité antinomique.
Pour tout système S fonctionnellement adéquat, il existe au moins un système coopératif S* qui soit fonctionnellement adéquat dans le même environnement.	La démonstration consiste en une expérience de pensée de déconstruction du système S pour en construire un nouveau S*. Elle est réalisée en quatre étapes : définir un algorithme de construction d'un système coopératif, montrer que cet algorithme se termine, montrer que le système coopératif obtenu est équivalent au système initial pour l'environnement, montrer que le nouveau système est fonctionnellement adéquat.
Tout système à milieu intérieur coopératif est un système coopératif.	Le milieu intérieur correspond aux parties du système ainsi qu'aux supports physiques nécessaires à leurs échanges. Un système à milieu intérieur coopératif possède des échanges coopératifs avec son environnement, car ceux-ci sont un sous-ensemble des échanges que réalisent ses parties.
Pour tout système coopératif, il existe au moins un système à milieu intérieur coopératif avec une fonction équivalente dans le même environnement.	La méthode est identique à celle du lemme 2. La particularité réside dans l'objet de la construction qui est maintenant chaque partie du système.

Tableau 1 - Récapitulatif du théorème de l'adéquation fonctionnelle

Ce résultat permet de ne s'intéresser qu'à des systèmes très particuliers (à milieu intérieur coopératif) pour obtenir des systèmes fonctionnellement adéquats dans un environnement donné. Une conséquence sur la conception de ces systèmes est que le concepteur ne doit plus avoir une démarche descendante mais ascendante. En effet, il doit développer les parties du système et faire en sorte que toute partie soit coopérative. Le théorème de l'adéquation fonctionnelle à la base de la théorie, permet donc de mettre en œuvre une technologie associée pour la conception des systèmes à fonctionnalité émergente.

3.2. L'aspect technologique

Les systèmes étudiés sont des systèmes multi-agents selon la définition donnée dans l'introduction. Au sein de ces systèmes les agents et leurs interactions définissent l'organisation du système. On peut définir l'organisation d'un système comme l'ensemble des liens entre les agents qui composent le système. Une organisation dépend fortement du problème en cours de résolution : elle est contextuelle et à une organisation correspond une fonction réalisée par le système.

Dans la technologie des AMAS, pour changer la fonction il suffit de changer l'organisation des composants du système. Ces mécanismes sont spécifiés par des règles locales régissant l'auto-organisation entre les composants et ne dépendant pas de la connaissance de la fonction collective. En effet, l'auto-organisation est définie comme la capacité d'un système à changer son organisation en cours de fonctionnement sans intervention externe du concepteur dans le cas d'un système artificiel.

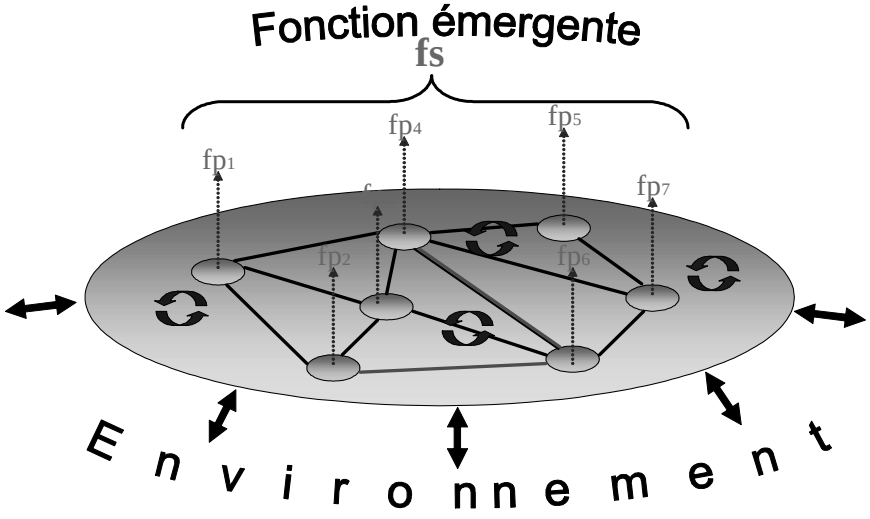


Figure 20 - Les parties d'un système en interaction dynamique

Spécifier un modèle a priori pour un système qui aura à faire face à des imprévus, c'est contraindre (peut être inopportunément) l'espace des possibles. De nombreux auteurs ont étudié des "systèmes de divers ordres qui ne peuvent s'appréhender que par l'étude de leurs parties prises isolément" [Bertanlanffy, 1993].

Un moyen d'apprendre pour un système S consiste à transformer sa fonction actuelle fs de manière autonome (donc par auto-organisation) afin de s'adapter à l'environnement, considéré comme une contrainte qui lui est donnée. Chaque partie Pi d'un système S réalise une fonction partielle fpi de la fonction globale fs. fs est le résultat de la combinaison des fonctions partielles fpi, notée par l'opérateur £ . La combinaison étant déterminée par l'organisation courante des parties, il s'ensuit que fs = fp1 £ fp2 £ … £ fpn. Comme généralement fp1 £ fp2 ≠ fp2 £ fp1, transformer l'organisation conduit à changer la combinaison des fonctions partielles et donc à modifier la fonction globale fs, devenant par là même un moyen d'adapter le système à l'environnement (cf. figure 20).

Pour un système multi-agent, la mise en œuvre de cette adaptation implique que le concepteur ne s'intéresse qu'à l'agent et lui donne les moyens de décider de manière autonome de changer ses liens avec les autres agents pour tendre vers une organisation coopérative. Ainsi, en fonction des interactions que le système multi-agent a avec son environnement, l'organisation entre ses agents émerge et constitue une réponse pour faire face aux imprévus.

La méthode sera la suivante : dès que, pour un état du monde particulier, un agent détecte une situation non coopérative, il engage un processus de transformation des liens qu'il a avec les autres agents, pour revenir à une situation coopérative. Parce que le corpus de croyances d'un agent n'est pas parfait pour agir de manière coopérative, il doit le créer dynamiquement. Il procède pour cela par rétroaction sur ses actions (a priori coopératives) afin d'ajuster ses croyances ; le résultat devrait être un système organisé selon un modèle d'organisation plus optimal. Un observateur de cette activité collective pourrait ainsi noter un état généralement stable, suivi par des réorganisations lorsque des perturbations sont détectées, pour retomber ensuite dans un nouvel état d'équilibre. Comme des agents modifient localement leurs relations dans ce processus, c'est une activité auto-organisatrice au sein du multi-agent. Il s'agit d'un apprentissage distribué étudié comme un processus d'auto-organisation [Camps, 1996].

Nous supposons que l'attitude coopérative est un principe général pour améliorer le résultat global de la société et pour définir un critère d'apprentissage efficace.

3.3. Les différents types de systèmes adaptatifs

Un **système multi-agent adaptatif** [Bernon, 2002c] est un système multi-agent qui est capable de changer son comportement en cours de fonctionnement pour l'ajuster dans un environnement dynamique, soit pour réaliser la tâche pour laquelle il a été conçu, soit pour améliorer sa fonction ou ses performances. Il est caractérisé par le fait :

- d'être plongé dans un environnement dynamique,
- de réaliser une tâche (fonction),
- d'être composé d'agents en interaction.

La particularité de la théorie des AMAS réside dans le fait que l'on ne code pas le moyen d'atteindre la fonction globale au sein d'un agent. Grâce à la capacité des agents à

s'auto-organiser, le système est capable de s'adapter par lui-même et réalise une fonction qui n'est pas codée dans l'agent. Cette fonction émerge et ceci est dû, en partie, aux interactions entre composants. Si l'organisation entre les agents change, la fonction qui est réalisée par la collectivité change aussi.

La manière de concevoir un système multi-agent adaptatif est récursive. En effet pour mettre en œuvre les croyances ou les compétences de l'agent, le concepteur peut utiliser un système multi-agent adaptatif s'il a besoin que ces dernières s'adaptent. Ainsi, on distingue trois types de systèmes :

- Les systèmes auto-organisés, pour lesquels les croyances et les compétences n'évoluent pas. Les systèmes du Tileworld et ANTS sont deux exemples de ce type, ainsi qu'une équipe de football logicielle développée pour la RoboCup [Carré, 1998].
- Les systèmes auto-organisateurs avec apprentissage des croyances, pour lesquels les croyances évoluent en cours de fonctionnement. C'est une nécessité dans le cas où des composants apparaissent et disparaissent dans le système.
- Les systèmes auto-organisateurs avec apprentissage des compétences, pour lesquels les compétences évoluent en cours de fonctionnement. Cette évolution implique de manière implicite une évolution des croyances.

3.4. ARCADIA (1995 – 1998) et ABROSE (1998 - 1999)

ARCADIA : ARchitecture pour la Coopération d'Agents D'Information Autonomes (1995 – 1998) est un projet CTI CNET avec comme partenaires : France Telecom CNET, l'IRIT Université Paul Sabatier Toulouse et l'ONERA-CERT Toulouse. ARCADIA représente la partie expérimentale de la thèse de Valérie Camps et le travail de Laurent Heurtevin [Heurtevin, 1996]. Ce projet a été aussi porté par les permanents : André Machonin, Pierre Glize et moi-même au sein de l'IRIT.

ABROSE : Agent based BROkerage SErvices in electronic commerce (1998 – 1999) est un projet européen ACTS qui comporte neuf partenaires : Deutsche Telekom Berkom, InfoMures SA Roumanie, France Telecom CNET, Universidad Polytechnica de Madrid, Onyx Ltd Angleterre, Dégriftour SA Paris, National Technical University of Athens, Technical University of Berlin et l'IRIT Université Paul Sabatier of Toulouse. Les permanents de l'IRIT impliqués dans ce projet étaient André Machonin, Pierre Glize et moi-même. J'étais responsable pour l'IRIT et responsable du lot concernant la technologie.

Ces deux projets ont été conçus à partir des mêmes concepts et de la même architecture, instanciée par des agents particuliers correspondant au domaine d'application considéré. C'est pourquoi ils sont présentés dans la même section. Ces deux projets illustrent les systèmes auto-organisateurs dans lesquels les croyances évoluent. Le système qui gère les croyances est présenté dans la partie 3.5 du chapitre II.

3.4.1. Objectifs

L'objectif d'ARCADIA est de gérer et d'obtenir l'information de sources de données dispersées géographiquement. C'est notamment le cas du World Wide Web où les sources d'information sont très dynamiques et autonomes, rendant ainsi tout contrôle global inadéquat.

ABROSE a pour objectif de modéliser une place de marché électronique constituée d'une collection importante de fournisseurs et de clients qui ont des offres et des besoins très dynamiques. ABROSE est un outil de commerce électronique de médiation de services fondé sur des agents. L'idée principale est qu'une mémoire collective se forme au sein d'une population de fournisseurs et de clients, représentés par des agents. Cette mémoire collective est constituée des expériences individuelles ou des recommandations formulées par les usagers. ABROSE cherche à modéliser et à traiter cette mémoire collective (et individuelle) dans une place de marché électronique, pour faciliter la qualité des échanges. Les principales fonctionnalités offertes sont :

- pour les clients (consommateurs) des interactions simplifiées, un assistant personnalisé, une notification spontanée des nouvelles offres, une aide à la navigation et à la formulation de requêtes, ainsi que la proposition d'une liste de fournisseurs pertinents répondant à la requête donnée par le client.
- pour les fournisseurs de service une diffusion ciblée des offres vers les seuls clients potentiellement intéressés, une collecte des informations sur les intérêts réels des consommateurs et sur les offres du marché.

3.4.2. Description de l'architecture

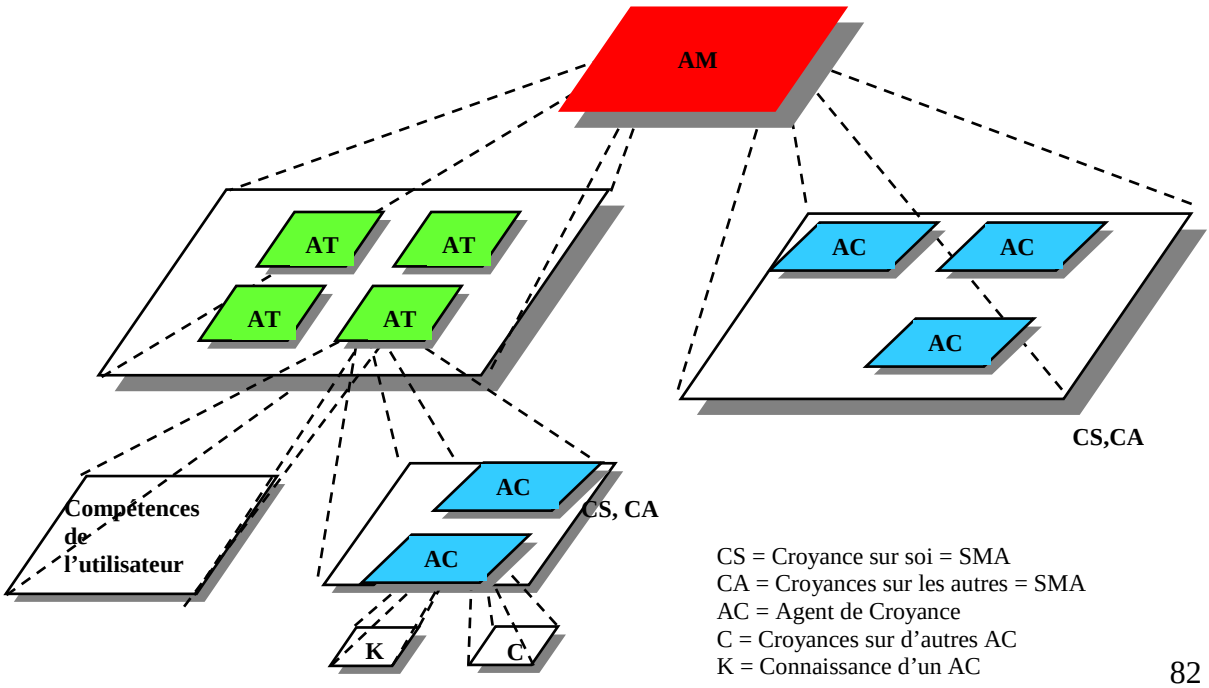


Figure 21 - Architecture

Pour atteindre ces objectifs, une architecture à plusieurs niveaux de systèmes multi-agents (figure 21) a été spécifiée et réalisée :

1- Le niveau supérieur est un système multi-agent de sites (agents de médiation : AM) communiquant par envoi de messages.

2- Chaque site est un système multi-agent de niveau intermédiaire (agents de transaction) contenant des agents interagissant en son sein. Chacun de ces agents représente soit un client, soit un fournisseur de service.

3- Pour que les interactions dans chacun des deux niveaux s'effectue de manière pertinente, les agents raisonnent en fonction de connaissances privées et dynamiques qu'ils possèdent les uns sur les autres : des croyances. Elles sont gérées dans un système multi-agent appelé réseau de croyances qui est privé à chaque agent (agent de transaction ou agent de médiation). Tous ces réseaux de croyances constituent le niveau inférieur. L'approche par auto-organisation coopérative de ces systèmes multi-agents garantit une satisfaction individuelle et collective optimale alors qu'il n'existe pas de connaissance globale de l'état du système.

L'agent de transaction

Chaque agent de transaction représente un utilisateur et agit pour son compte. Ils sont créés quand un nouveau fournisseur de service est créé ou quand un nouveau client est enregistré dans le système. Les agents de transaction coopèrent pour répondre à une requête ou pour propager une offre.

La structure d'un agent de transaction

Un agent de transaction est constitué de croyances, de compétences, d'un langage d'interaction et d'une attitude sociale. Les compétences correspondent aux compétences de l'utilisateur qu'il représente. Quand un agent reçoit un message, il l'interprète et agit en conséquence. Les croyances décrivent les croyances que cet agent a sur lui-même et sur d'autres agents de transaction. Leur implémentation est décrite au chapitre suivant. L'attitude sociale est la coopération, elle guide leur comportement. Les agents de transaction coopèrent en échangeant des messages. Ils utilisent pour cela des actes du langage et des protocoles qui sont des sous-ensembles de ceux définis par les actes de communication ACL de FIPA [Léger, 1998], [Glize, 1999].

Le comportement d'un agent de transaction

Les agents de transaction constituent les parties d'un système multi-agent adaptatif. Ils doivent donc essayer de maintenir des interactions coopératives entre eux, c'est pourquoi, chaque agent est doté d'une attitude sociale coopérative qui lui confère trois propriétés :

- sincérité : un agent dit la vérité sur ce qu'il sait,
- bienveillance : un agent essaie de satisfaire les requêtes qu'il reçoit si elles sont en adéquation avec ses propres compétences et ses objectifs,

- réciprocité : un agent sait que les autres ont la même attitude coopérative que lui. Ses actes sont donc dépendants des croyances qu'il a sur autrui.

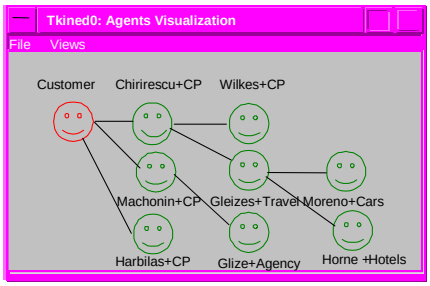


Figure 22 - Coopération entre agents de transaction

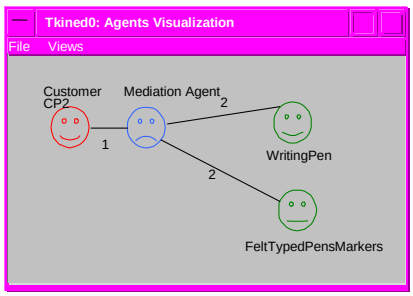


Figure 23 - Coopération entre un agent de transaction et un agent de médiation

Par conséquent, un agent coopératif envoie spontanément une information à un autre agent s'il pense qu'elle est utile pour cet agent. S'il ne peut pas satisfaire une requête donnée, il va automatiquement recruter les agents qu'il pense pertinents en interrogeant ses croyances. Un agent de transaction qui ne sait pas répondre à une requête ou à une offre qu'il reçoit, peut faire appel soit à d'autres agents de transaction qu'il connaît et qu'il trouve pertinents (figure 22), soit à l'agent de médiation qui a un point de vue général sur la place de marché (figure 23).

L'évolution des croyances

Un agent de médiation possède des croyances sur lui-même tandis qu'un agent de transaction possède des croyances sur lui-même et sur d'autres agents de transaction. Une croyance correspond à une description par un ensemble de termes d'un agent donné. Plusieurs croyances peuvent être utilisées pour décrire un agent. Chaque terme est encapsulé par un agent de croyance et l'ensemble de ces agents constitue un réseau de croyances qui correspond à un système multi-agent adaptatif. A la création d'un agent de transaction ou de médiation, son réseau de croyances est vide. Au fur et à mesure qu'il va avoir des interactions avec d'autres agents son réseau va se modifier : il va ajouter, renforcer ou oublier des croyances. Chaque agent possède son propre réseau de croyances qui évolue de manière indépendante, autonome et locale à l'agent auquel il appartient. Les deux parties suivantes décrivent les moments où le réseau de croyances d'un agent de transaction et d'un agent de médiation va apprendre à partir d'une interaction avec un agent.

L'évolution des croyances d'un agent de transaction

Les croyances d'un agent de transaction évoluent :

- Si l'agent représente un client (respectivement un fournisseur de services), quand une nouvelle requête (respectivement offre) lui est fournie, il apprend que le client est intéressé par la requête (respectivement que le fournisseur de services peut répondre à l'offre).
- Si l'agent représente un client, quand il évalue l'offre reçue ou la transaction qu'il a effectuée, il apprend sur lui et sur le fournisseur de services qui a fait l'offre ou qui a répondu à la requête.

- Si l'agent représente un client (respectivement un fournisseur de services) et s'il a demandé de l'aide à l'agent de médiation, il apprend sur les fournisseurs de services (respectivement sur les clients potentiels) que lui a communiqués l'agent de médiation.
- Si l'agent représente un client (respectivement un fournisseur de services) et s'il reçoit une réponse d'un fournisseur de services (respectivement d'un client) par un autre agent de transaction, il apprend sur le fournisseur de services (respectivement sur le client).

Ainsi au cours du fonctionnement du système, les interactions entre les agents de transactions évoluent : le processus d'auto-organisation correspond à l'évolution de leurs croyances réciproques.

L'évolution des croyances de l'agent de médiation

Les compétences de l'agent de médiation sont une image des compétences de tous les agents de transaction qui sont situés sur le site. Il dispose d'un réseau de croyances sur ses compétences c'est-à-dire sur lui-même. Il a pour fonction de créer des agents de transaction et de maintenir la mémoire des expériences de tous les utilisateurs pour donner une vue globale de la place de marché dans ses croyances. Ainsi, ses croyances sur les agents de transaction évoluent de manière autonome, elles ne correspondent pas à l'union des croyances des agents de transaction.

Les croyances qu'il a sur lui-même sont enrichies quand :

- Une nouvelle offre est donnée. Il apprend qu'une nouvelle offre est proposée par le fournisseur de services si l'agent de ce dernier fait appel à lui pour connaître les clients de la place de marché susceptibles d'être intéressés.
- Un agent de transaction lui transmet une requête, parce qu'il est bloqué. Il apprend que ce client est à la recherche de produits ou de services décrits dans la requête.
- Il apprend sur le client et sur le fournisseur de services quand le client évalue les offres.
- Il apprend sur le client et sur le fournisseur de services quand le client évalue la requête.

3.4.3. Résultats et analyse

L'originalité de ces systèmes et leurs principaux avantages sont les suivants:

- Le système n'utilise pas d'ontologie prédéfinie, l'ontologie se construit par apprentissage du système lors de ses échanges.
- Le système n'est pas dédié à la vente d'un certain type de produit. Il peut être utilisé pour n'importe quel domaine.
- Le système n'est pas dédié à une langue, les fournisseurs de service peuvent décrire leurs services dans la langue de leur choix. Bien entendu les requêtes devront être réalisées dans la même langue.

- Le système est ouvert, la création ou la suppression d'agents de transaction se fait en dynamique. La découverte d'un nouvel agent ou la disparition d'un agent est apprise par les autres agents au fur et à mesure.
- Le système s'adapte en permanence à l'évolution de la place de marché et des nouvelles distributions de l'information.
- Chaque agent a une connaissance partielle des autres agents.

3.5. Le réseau de croyances

Les personnes ayant participé à sa conception sont Laurent Heurtevin (stage IUP) [Heurtevin, 1996], Jérôme Carré (stage de DEA) [Carré, 1998], Valérie Camps (thèse de l'UPS) [Camps, 1998a]. Les permanents ayant en charge ce projet sont André Machonin Pierre Glize, et moi-même.

Le réseau de croyances est un système multi-agent adaptatif permettant de mémoriser et de mettre à jour de manière automatique (par adaptation) les croyances qu'un agent a sur d'autres agents ou sur son environnement. Ce système est ensuite implanté dans un agent de plus grosse granularité et correspond dans cet agent au module de croyances. Il est donc au coeur des projets ARCADIA, ABROSE et FORSIC[9] qui illustrent la notion de systèmes auto-organisateurs dans lesquels les croyances évoluent.

3.5.1. Objectifs

L'utilisation toujours croissante de l'Internet et le volume considérable des informations disponibles en ligne exigent que des outils adaptés aident un utilisateur à accéder aux informations pertinentes pour lui. En effet, la sélection d'informations pertinentes dans un ensemble d'informations est une opération très coûteuse en temps. Les moteurs de recherche peuvent aider à trouver des documents mais il est nécessaire d'avoir aussi une description des centres d'intérêts et des préférences de l'utilisateur. C'est ainsi qu'une aide logicielle appelée "profil utilisateur" permet de maintenir des informations sur ses préférences, sur les services utilisés… et peut être construit explicitement ou déduit par le système en observant son comportement. Pour l'apprendre automatiquement, l'agent représentant l'utilisateur doit "regarder par dessus son épaule" comme le dit Maes [Maes, 1994]. A cette fin, nous proposons une architecture générique de système multi-agent adaptatif : le système SCIO.

Le domaine de l'apprentissage d'un profil utilisateur dynamique impose plusieurs contraintes fortes, qui ont fait porter notre choix vers un système multi-agent adaptatif :

1- L'apprentissage doit se réaliser à partir de phrases linguistiquement bien formées ou non. En effet, ces phrases sont saisies par un utilisateur dont on ne peut préjuger de la qualité linguistique, notamment par l'usage courant du style « télégraphique ».

[9] FORSIC (FOrmation et Recherche/Sciences de l'Information et de la Communication) est un projet national conçu sur les mêmes bases qu'ABROSE.

2- L'ontologie du domaine dans lequel les échanges vont s'effectuer ne peut se définir au moment de la conception, car l'objectif est d'employer le même type de système dans des milliers d'agents représentant des usagers ayant des besoins très diversifiés.

3- Les centres d'intérêt d'un utilisateur évoluant en permanence, l'apprentissage ne peut se réaliser que durant une phase initiale.

La gestion d'un profil utilisateur demande au système chargé de cette maintenance une interaction permanente avec l'utilisateur. Un système multi-agent adaptatif étant un système réactif, il répond à cette contrainte de fonctionnement.

La gestion d'un profil utilisateur est un domaine d'application typique pour les systèmes adaptatifs. En effet, cette application est dynamique car le profil doit évoluer en cours de fonctionnement ; l'environnement étant évolutif on ne connaît pas a priori les centres d'intérêt des utilisateurs. Le système doit donc être ouvert, il faut pouvoir rajouter des termes ou en enlever… Le besoin d'adaptation aux évolutions du profil utilisateur est un deuxième besoin auquel répondent les systèmes multi-agents adaptatifs.

Savoir si un document est pertinent pour un utilisateur impose l'activité corrélée de plusieurs de ses composants : les termes. Le fait qu'un profil utilisateur soit composé d'un nombre important de termes qui sont interdépendants et dont les liens peuvent évoluer au cours du temps correspond bien à l'architecture d'un système multi-agents.

Tous ces éléments justifient le choix d'un système multi-agent adaptatif pour la gestion de profils utilisateurs.

3.5.2. Description

Les interactions avec l'environnement

Pour apprendre et s'adapter, le système SCIO est plongé dans un environnement qui lui permet de percevoir l'utilisateur qu'il représente et les informations qui lui sont présentées. Pour être en interactions coopératives avec cet environnement, le système doit être apte à interpréter les descriptions reçues de manière à les filtrer pour l'utilisateur. Mais si ses réponses ne semblent pas correctes il recevra ultérieurement en retour des messages dont le contenu perturbera son organisation interne et provoquera une réorganisation.

Il perçoit des signaux dont il connaît a priori la structure mais pas le contenu (Description). Le tableau ci-dessous résume ces échanges. La première colonne recense le type de signaux envoyés par l'environnement, la seconde l'activité interne des agents du système SCIO et la troisième la réponse du système SCIO à son environnement.

Les messages reçus par le système sont de deux types :

- les messages qui permettent au système d'apprendre sur l'utilisateur,
- les messages qui permettent d'interroger le profil utilisateur avec la description d'un document pour faire du filtrage.

< Utilisateur ! Description > et < Utilisateur ¡ Description > sont des informations sur l'intérêt qu'a un utilisateur par rapport à une description (! intérêt positif, ¡ intérêt négatif). Ces descriptions peuvent être données par l'utilisateur ou déduites du feed-back que donne l'utilisateur sur une description reçue.

<? Description > permet d'interroger le système avec une description si la réponse <Utilisateur ! Description> est donnée par le système c'est que l'utilisateur est intéressé par cette description sinon aucune réponse n'est retournée.

Message reçu et interprétation	Raisonnement dans SCIO	Message émis et interprétation
< Utilisateur ! Description > L'utilisateur donne une description de ce qui l'intéresse dans Description	Activation et apprentissage par réorganisation	Néant
< Utilisateur ¡ Description > L'utilisateur n'est pas intéressé par la description donnée dans Description	Activation et apprentissage par réorganisation	Néant
< ? Description > Cherche si l'utilisateur est intéressé par la description donnée dans Description	Activation	< Utilisateur ! Description > L'utilisateur est intéressé par la description donnée dans Description

Tableau 2 - Les échanges SCIO / Environnement

Les agents coopératifs du système SCIO

Les agents composant SCIO sont appelés agents de croyances. Dans cette architecture, nous avons donc dû créer les agents de manière à ce que l'ensemble de leurs fonctions individuelles soit suffisant pour obtenir le comportement collectif souhaité, par auto-organisation. Les spécifications nous ont permis d'aboutir à un système constitué de trois types d'agent de croyances : les agents Terme, les agents User, et l'agent Interface, identifiés de manière univoque par un nom.

L'agent Interface est unique et son rôle consiste à initier l'activation du système et à collecter les réponses. Il est l'intermédiaire entre SCIO et l'environnement pour percevoir son état et y agir. C'est le seul agent existant à l'initialisation du système. Pour plus de clarté, il est représenté deux fois sur la figure 2 en entrée et en sortie du système.

L'agent de type User représente l'utilisateur. Il a pour rôle d'agréger les descriptions des centres d'intérêt et des préférences de cet utilisateur.

Les agents de type Terme encapsulent tous les termes des descriptions apprises par SCIO. Ainsi, chaque mot n'a qu'un agent terme qui le représente, même s'il est présent dans plusieurs descriptions au cours du temps. Ils sont au cœur du processus d'auto-organisation qui détermine la dynamique du système.

L'architecture de SCIO

Un agent de croyances possède une représentation des autres agents pour interagir et analyser les situations non coopératives potentielles. L'interaction entre agents de croyances s'effectue par messages dont le contenu porte sur le résultat de l'activité ou l'information d'une situation non coopérative vers autrui. Les représentations d'autrui mémorisées chez un agent de croyances s'expriment sous la forme d'un nombre entier et sont représentées par les flèches dans la figure 24. Une valeur nulle n'est pas stockée car elle signifie qu'un agent n'a pas de représentation de l'autre. Ces représentations sont regroupées au sein des agents dans deux listes : la liste des agents pères et la liste des agents fils

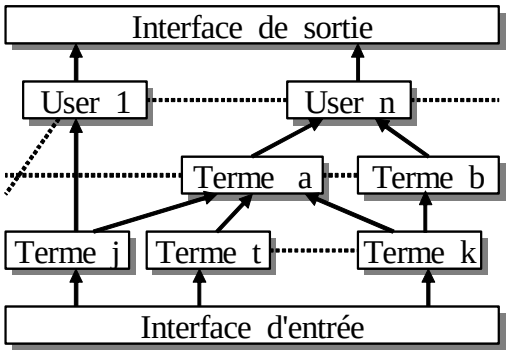


Figure 24 - Organisation des agents du système SCIO

La relation père / fils a été définie comme suit :

- si un agent A est le fils d'un agent B alors l'agent B est le père de l'agent A,
- un agent peut avoir plusieurs pères et plusieurs fils,
- un agent ne peut pas être plusieurs fois le père (respectivement le fils) d'un même agent,
- si un agent A est le père d'un agent B alors l'agent B peut aussi être le père de l'agent A.

De plus, chaque agent de type User ou Terme, possède une fonction d'activation à deux seuils : un seuil d'activité et un seuil de réveil. Ainsi l'agent de croyances peut prendre trois états : passif, réveillé et actif.

- L'état passif de l'agent de croyances survient si aucun de ses fils n'est actif.
- L'état actif signifie que la somme des poids des croyances associées à ses fils actifs est supérieure au seuil d'activité.
- L'état réveillé correspond à une prémisse d'activité : il a des fils actifs mais la somme des poids de leurs croyances est insuffisante. C'est le seul cas de non coopération, car un agent seulement réveillé a perçu des signaux mais ne peut pas arriver à réaliser sa fonction.

L'activité d'un agent de croyances

L'agent Interface est à l'origine de l'activation directe d'un agent Terme lorsque le terme que représente cet agent est présent dans une description textuelle envoyée par l'agent Interface (voir tableau 2). Pour que l'activation d'un agent se propage, il faut que les agents qui reçoivent son message puissent eux-mêmes devenir actifs.

Après l'étape d'activation, seuls les agents réveillés correspondent à un cas de non coopération, car un agent réveillé a perçu des signaux mais ne peut pas arriver à réaliser sa fonction. Ces agents deviennent générateurs de messages de situations non coopératives vers leurs fils pour les prévenir qu'ils ont peut-être fait une erreur sur leur état. Si un agent a reçu une majorité de messages de non coopération, il va les propager vers ses fils.

Après la détection de situations non coopératives, chaque agent tente en autonome de la résorber. Ceci est réalisé par l'ajustement des poids des pères ou fils pour que l'agent devienne soit totalement passif, soit actif et qu'il ne soit plus en contradiction avec ses accointances. Cet ajustement peut conduire à des réorganisations :

- Montée lorsque le poids sur le lien avec le père est supérieur à la valeur absolue du seuil d'activation, l'agent remonte dans l'organisation hiérarchique locale.
- Descente lorsque le poids sur le lien avec le père est nul, l'agent descend dans l'organisation s'il n'est pas au dernier niveau
- Disparition lorsque le poids sur le lien avec le père est nul et que l'agent est au dernier niveau.

Cet ajustement permet aux agents de modifier leurs représentations des autres et donc de modifier l'organisation entre les agents de croyances. Une nouvelle organisation émerge, elle peut être observée (voir chapitres suivants) et permet au système de modifier ses réponses à de nouvelles requêtes, donc une nouvelle fonction émerge. L'architecture et le fonctionnement que nous venons de spécifier ne suffisent pas pour déterminer les propriétés de convergence, de stabilité, de robustesse du système. Cela est d'autant plus vrai pour un système multi-agent adaptatif dont les hypothèses initiales sont minimes laissant alors une large plage de comportements possibles.

3.5.3. Résultats et analyse

Quand il y a apprentissage à partir d'une nouvelle description, la propagation d'information entre les agents de croyances entraîne une restructuration du système SCIO et pas seulement l'ajout des termes non encore présents dans le profil. Cette modification peut donner lieu automatiquement à une suppression d'agents de croyances ou à une réorganisation de ces agents.

Pour valider l'utilisation d'un système, il faut maintenant en tester les performances et interpréter les résultats. Nous allons donc ici observer quelques-uns de ses paramètres à partir d'expérimentations :

- La suppression de croyances d'agents qui ne sont pas pertinents dans un domaine.
- L'accroissement de croyances d'agents qui sont pertinents dans un domaine.
- La stabilisation du système lorsque chaque agent a trouvé sa bonne place dans l'organisation.

C'est l'ensemble de ces processus d'ajustement qui permettent d'obtenir un système coopératif lors des échanges avec l'agent de transaction. Cette coopération permet d'affirmer que le système a convergé vers l'adéquation fonctionnelle.

Suppression d'agents non signifiants d'un domaine

Nous considérons un domaine de descriptions préalablement apprises par le système SCIO et portant sur des voitures d'occasion. Elles contiennent des termes qui ne sont pas directement signifiants tels des articles. Nous observons ici l'article anglais "a" présent dans trois descriptions qui possèdent respectivement 5, 8 et 20 termes. Ce domaine est maintenant interrogé à plusieurs reprises avec la description "*I search for a computer with its monitor*" contenant l'article mais n'ayant aucun rapport avec le domaine considéré. La figure 25 indique en abscisse le nombre d'interrogations et en ordonnée la valeur courante des poids relatifs à chaque description. Nous observons une décroissance régulière (et souhaitée) du poids de l'article dans chacune des descriptions jusqu'à sa disparition totale. Cela provient de situations non coopératives détectées entre l'agent représentant "*a*" et ses parents conduisant à un relâchement des liens.

Emergence d'agents pertinents d'un domaine

Si la disparition de termes non pertinents est possible, il faut aussi observer l'émergence d'une organisation entre les termes usités d'un domaine. Pour cela, nous réalisons des transactions dans le domaine des voitures avec des descriptions similaires à celle-ci : "*I want to buy a Peugeot car with air conditioning maximum 5000 kilometers the color is not important*". Les termes ont tous un poids initial identique (y compris les articles), comme l'indique l'ordonnée ; tandis que l'abscisse représente les échanges avec l'environnement. Sur la figure 26 nous observons progressivement la prééminence du terme générique "car", tandis que l'ajustement de "*Peugeot*" et "*Ford*" oscille autour des mêmes valeurs. Le système arrive ainsi à créer des relations de type générique/spécifique entre termes, bien que le processus d'adaptation n'ait connaissance d'aucune ontologie du domaine ni de capacité d'analyse linguistique de phrases.

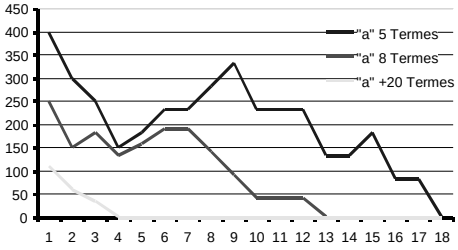


Figure 25 - ***Evolution d'agents non pertinents***

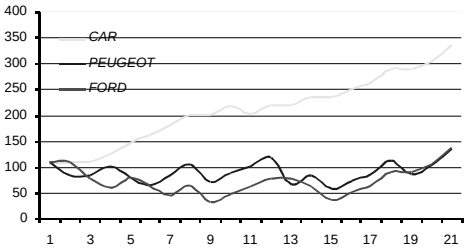


Figure 26 - ***Evolution d'agents pertinents***

Stabilisation du système

Si les précédentes expériences montrent un processus effectif d'ajustement interne par réorganisation, il ne faut pas qu'il soit permanent, ce qui indiquerait une incapacité à apprendre des comportements adéquats dans son environnement. C'est pour cela que l'expérience de la figure 26 reprend l'exemple du domaine des voitures d'occasion, dans laquelle nous supposons des descriptions assez proches des apprentissages antérieurs du système. L'abscisse indique les transactions effectuées avec l'environnement, tandis que

l'ordonnée représente le nombre d'agents impliqués dans le processus de réorganisation à chaque instant. Le nombre moyen des ajustements décroît clairement et cela pour deux raisons :

- les termes non signifiants pour le domaine sont progressivement supprimés du système, comme nous l'avons indiqué dans les expérimentations antérieures,
- les relations entre les termes du domaine tendent vers des valeurs qui s'équilibrent ce qui diminue progressivement le nombre d'ajustements.

La conséquence essentielle de cette stabilité est que, en accord avec la théorie, le système converge vers la fonction globale souhaitée (adéquation fonctionnelle). Il en résulte aussi une grande efficacité car le système étant bien organisé tous les échanges entre les agents sont productifs pour la réponse globale.

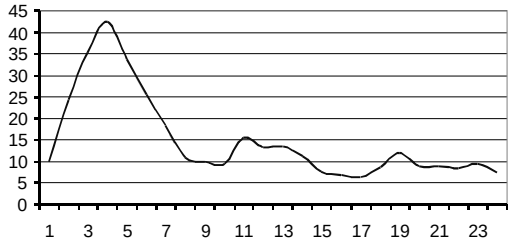


Figure 27 - Nombre d'agents en situation non coopératives au cours des échanges

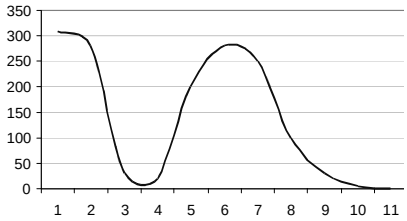


Figure 28 - Apparition de situations non coopératives dans le système

Si la figure 27 porte sur un système avec un seul niveau d'organisation interne, l'expérience de la figure 28 est réalisée avec deux niveaux. Les conditions en sont les suivantes: un système initial est construit avec une quinzaine de descriptions relatives à des Users différents constituant un système à deux niveaux d'organisation entre les agents Terme. Le système ainsi constitué est interrogé une vingtaine de fois avec la même description (représentée par l'abscisse) contenant deux à trois termes présents dans les descriptions initiales. Nous observons un processus de réorganisation répétitif, dont les nombres sont indiqués en ordonnée. A chaque maximum de phase, beaucoup d'agents User sont réveillés, mais non activés, par ces termes ce qui produit beaucoup de situations non coopératives qui ont pour conséquence de faire disparaître progressivement des liens entre les agents, d'où le creux des phases. A la fin le système ne se réorganise plus car toutes les situations non coopératives ont disparu.

D'un point de vue général avec un système contenant beaucoup plus d'agents, nous observons des situations similaires lorsque des termes non signifiants dans des descriptions sont activés au cours de l'interrogation. Ils ont progressivement tendance à disparaître, pour ne laisser émerger dans l'organisation que les termes pertinents d'un domaine. Comme pour l'exemple précédent, l'adaptation se réalise en quelques étapes (interrogations du système) car le processus coopératif entre agents amplifie les phénomènes de réorganisation.

L'utilisation du système SCIO dans le prototype d'ABROSE a permis de montrer que la technologie des systèmes multi-agents adaptatifs permet de résoudre des problèmes dans un environnement dynamique tel qu'une place de marché électronique et que la qualité du service de courtage était améliorée grâce à l'utilisation des profils des usagers d'ABROSE.

Les propriétés essentielles du système peuvent se résumer en quatre points :

- La généricité. Le profil peut être utilisé pour la vente ou la promotion de n'importe quel produit ou service. Cela est rendu possible par l'absence d'ontologie prédéfinie, elle se construit dynamiquement par apprentissage.
- Le multilinguisme. Les fournisseurs de service peuvent décrire leurs produits ou leurs services dans n'importe quelle langue (voire plusieurs), bien entendu les interrogations de ces profils devront être réalisées dans la même langue.
- L'ouverture. C'est un système ouvert car la création ou la suppression d'agents de croyances se fait en dynamique sans intervention humaine directe.
- L'adaptation. Le profil tient compte en temps réel et de manière automatique de l'évolution de son environnement (les préférences des utilisateurs et les services des fournisseurs).

3.6. Contributions

Dans le cadre des travaux sur la théorie des AMAS, une manière ascendante, complètement différente des méthodes classiques de conception qui sont généralement descendante a été confortée. Elle se rapproche de la manière de travailler des concepteurs de systèmes multi-agents réactifs. La différence réside dans le fait que cette démarche est utilisée indépendamment de la granularité des agents.

Les systèmes multi-agents adaptatifs conçus comme ARCADIA, ABROSE, FORSIC sont des systèmes dans lesquels les représentations qu'un agent a des autres sont modifiées en cours d'utilisation. La théorie des AMAS, qui permet de concevoir des systèmes qui s'adaptent, a été utilisée au sein d'un agent pour gérer les représentations qu'il a des autres agents. Cet apprentissage de connaissances n'est réalisé à partir d'aucune connaissance. L'architecture des systèmes d'information est conforme à l'architecture standard des systèmes de recherche d'informations de la FIPA réalisés à la même période (1997) (www.fipa.org). L'originalité de notre approche pour des systèmes de recherche d'informations est que la gestion du service de pages jaunes est effectuée automatiquement.

L'apprentissage réalisé par système multi-agent adaptatif est une technique complètement nouvelle par rapport aux techniques d'apprentissage existantes comme l'apprentissage par renforcement pour ne citer que la plus importante utilisée dans le domaine des systèmes multi-agents.

4. Enseignements de cette période

Les travaux de cette période consistent en une étude des interactions entre les agents d'un système et des interactions entre le système et son environnement. Le principal résultat est d'avoir élaboré une théorie des systèmes multi-agents adaptatifs, d'avoir

commencé expérimentalement sa validation sur les projets présentés dans ce mémoire mais aussi sur les projets tels que STAFF [Sontheimer, 1999] industrialisé de nos jours, la répartition de charges sur un réseau téléphonique [Dotto, 1996], les robots footballeurs (SMACKers) [Pesquet, 1999].

4.1. La coopération

Une partie de nos travaux ont eu comme objectif d'étudier la coopération ou plus exactement ce qui est défini comme un comportement coopératif. La coopération a été définie comme une heuristique pour améliorer le résultat global d'un collectif et aussi comme le moteur de l'auto-organisation. L'étude des interactions entre agents nous a permis de montrer par la réalisation d'applications que la coopération est une stratégie de résolution efficace et que la résolution de situations non coopératives permet de s'adapter à un environnement évolutif comme dans le Tileworld ou dans ANTS. Nos principaux résultats à ce niveau ont été la réalisation d'un modèle d'agent coopératif notamment avec la définition plus complète de ce que doit être l'attitude coopérative d'un agent et la définition de la coopération. En effet, une définition idéale de la coopération doit à notre sens inclure les aspects suivants :

- Du point de vue de la perception, tout signal devrait être interprété sans ambiguïté,
- Du point de vue de la déduction, toute information (signal interprété) devrait entraîner des conséquences logiques. Notamment, une information doit amener de la nouveauté : une différence avec les informations antérieures.
- Du point de vue de l'action, les conclusions (résultat de la fonction) doivent être utiles (à autrui ou l'environnement).

Les deux premières conditions peuvent être jugées localement par l'agent en utilisant les connaissances qu'il possède sur ses propres compétences. La dernière condition peut être détectée par l'agent après qu'il ait étudié son environnement et notamment l'image qu'il possède sur les capacités de ses congénères.

Le qualificatif "idéale" provient du fait que dans un système complexe, la coopération telle que nous la définissons n'est jamais réellement atteinte, si ce n'est ponctuellement. D'autre part, notre définition ne fait intervenir que le point de vue individuel de chaque agent et ne repose en aucun cas sur des critères globaux ; elle ne dépend pas de la fonction globale qu'est censé réaliser le système.

Ce travail sur la coopération va donc au-delà de la mise en œuvre de mécanismes de coopération comme le partage de tâche, l'élaboration de contrats… utilisés dans de nombreux systèmes multi-agents. Nous avons voulu étudier ce que signifiait être coopératif et ainsi nous avons donné une définition générique de la coopération.

4.2. La théorie des AMAS

La théorie des AMAS est le résultat d'années de recherche et d'un travail collectif dans l'équipe SMAC. Pour établir une théorie scientifique la démarche a été la suivante,

nous avons énoncé des théorèmes qui ont été démontrés et des axiomes. Dans un premier temps, la théorie a été validée par expérimentation.

Au cours de la période 94-99, la méthode de travail pour poursuivre nos recherches s'est écartée de l'approche formelle de conception des systèmes d'IA classique. L'élaboration d'une théorie scientifique a été expérimentée, les résultats observés et interprétés. Dans ce travail les critiques le plus souvent faites concernent la décidabilité, la complétude, la convergence mais comme le fait remarquer Drogoul dans son habilitation la nouveauté peut émerger de travaux éloignés de la démarche formelle [Drogoul, 2000]. Je rappelle que l'approche a été d'abord expérimentale. En ce sens c'est la construction de systèmes réels qui a permis dans un premier temps d'étayer cette théorie et dans un deuxième temps de la valider expérimentalement.

La théorie des AMAS s'écarte des théories logiques classiques utilisées dans les systèmes multi-agents, c'est une nouvelle manière d'appréhender et de concevoir ces systèmes dans lesquels la fonction globale émerge des interactions entre les agents et donc de faire de la résolution distribuée de problèmes.

Inspirés par les travaux de Maturana et Varela, le couplage structurel entre un système et son environnement mis en évidence par ces chercheurs au niveau des systèmes naturels, nous a semblé aussi important pour des systèmes artificiels devant s'adapter. Nous nous sommes donc intéressés à la notion de système et c'est pourquoi le théorème de la théorie des AMAS a été énoncé et démontré. Ce théorème est essentiel car il guide la manière de développer les systèmes multi-agents, en imposant une manière ascendante. L'application ou le problème à résoudre selon la théorie des AMAS peut se résumer au suivi des deux étapes suivantes : d'abord déterminer les agents, puis leur donner un comportement coopératif. En effet, le théorème dit qu'un système à milieu intérieur coopératif est fonctionnellement adéquat.

Décomposer un système en agents et leur attribuer des compétences (leur donner la capacité de résoudre des problèmes ou d'exécuter des tâches) est une activité qui est réalisée par tous les concepteurs de système multi-agent de cette période, l'originalité de notre approche réside dans le fait que nos travaux précédents sur la coopération nous guident aussi pour donner un comportement aux agents. En fait l'attitude coopérative telle que nous la définissons contrôle au niveau local le comportement de l'agent.

Durant cette période, dans l'équipe de nombreuses applications ont été développées : Tileworld, ANTS, SMACkers, Répartition adaptative de charge dans un réseau téléphonique (en coopération avec le LAAS [Dotto, 1999], Résolution d'équations, ARCADIA, ABROSE, STAFF (prévision de crues) en collaboration avec ARTAL Technologies [Sontheimer, 1999]. Ces applications nous ont permis de valider expérimentalement la théorie élaborée. De plus, le développement d'applications en partenariat avec des industriels a permis le transfert de technologie du système de prévision de crues STAFF.

Au terme de cette période, nous avons caractérisé exactement les applications auxquelles la théorie des AMAS apporte une aide pour la conception. Ces caractéristiques sont les suivantes :

- l'environnement du système est dynamique, rendant impossible une énumération exhaustive de toutes les situations que le système peut rencontrer,

- le système est ouvert et donc dynamique car constitué d'un nombre variable de composants,
- la tâche que le système doit remplir est trop complexe pour être spécifiée de manière sûre,
- le moyen pour le système d'atteindre son but est trop complexe pour être globalement spécifié par le concepteur.

4.3. De nouvelles problématiques de recherche

Les systèmes développés selon la théorie des AMAS dans cette période sont :

- des systèmes dits auto-organisés comme ANTS ou le Tileworld pour lesquels il n'y a pas de modification au sein de l'agent. Le système s'adapte aux changements qui surviennent dans l'environnement.
- des systèmes dits auto-organisateurs avec apprentissage de croyances comme ARCADIA, ABROSE.

En terme d'applications et de recherche nous allons nous orienter vers la troisième catégorie de systèmes (définie dans la partie 3.3 du chapitre II), des systèmes pour lesquels il y a apprentissage des compétences. L'intérêt va se porter sur l'agent et ses constituants.

La fonction globale réalisée par le système émerge du comportement et des interactions entre les agents. L'émergence est au coeur de notre théorie c'est une notion très ancienne et étudiée dans d'autres domaines tels la philosophie, la biologie, la physique, la chimie… En 1999, dans le domaine des systèmes multi-agents, des travaux sur ce concept voient le jour tels les travaux de Servat [Servat, 1999], de Steels [Steels, 1999]. On parle de plus en plus d'émergence dans le domaine, au sein du groupe Colline (groupe de travail de l'AFIA collège SMA), nous avons commencé à définir cette notion [M. R. Jean, 1997] mais des questions restent en suspens telles que :

- Comment définir l'émergence dans les systèmes artificiels ?
- Comment caractériser un phénomène émergent ?
- Comment valider ou montrer la convergence d'un phénomène émergent ?
- Comment contrôler l'émergence ?

En général dans le domaine, les systèmes multi-agents sont développés par un seul concepteur, les agents sont homogènes, la plupart du temps ces systèmes sont fermés dans le sens où ils n'admettent pas la création en cours d'exécution d'autres agents ou bien la prise en compte d'un environnement dynamique. Les protocoles de communications sont prédéfinis par les concepteurs et les approches de conception sont ad hoc. Il est donc nécessaire d'orienter les recherches vers des méthodes de conception de tels systèmes. L'enjeu est essentiel, la création de méthodes permettra notamment d'effectuer un transfert de technologie vers les industries. Ce dernier point a été renforcé par notre collaboration avec les industriels d'ARTAL Technologies sur le projet STAFF. Le besoin d'expliquer la manière de concevoir les systèmes selon la théorie des AMAS et de fournir une méthode de développement adaptée est devenu nécessaire. Les besoins d'outils de développement tels que des plates-formes ou des infrastructures sont tout aussi prépondérants.

5. Bibliographie relative au chapitre II

[Agre, 1995] Agre P. E., Rosenschein J. S. (Guest Ed.) *« Special Volume Computational research on Interaction and Agency »,* Artificial Intelligence Vol 72-73 N°1-2 January-February 1995 Elsevier Science

[Bernon, 2002] Bernon C., Gleizes M-P., Picard G., Glize P., *« The Adelfe Methodology For an Intranet System Design »,* Fourth International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS-2002), 27-28 May 2002, Toronto (Ontario, Canada) at CAiSE'02.

[Bertanlanffy, 1993] von Bertanlanffy, *«Théorie générale des systèmes* », Editions Duniod 1993.

[Camazine, 2001] Camazine S., Deneubourg J-L., Franks N., Sneyd J. Theraulaz G., Bonabeau E ., *« Self-organization in Biological Systems »,* Princeton N.J. Princeton University Press

[Camps, 1996] Camps V., Gleizes M-P, *« Attitudes coopératives individuelles pour l'adaptation collective »,* Actes des quatrièmes journées francophones IAD&SMA, Port Camargue 1-3 avril 1996 Editions Hermès

[Camps, 1997a] Camps V., Gleizes M.P, *« Une technique multi-agent pour rechercher des informations réparties »,* Actes des cinquièmes journées francophones IAD&SMA, La Colle sur Loup, Editions Hermès, pages 29-46

[Camps, 1997b]Camps V., Gleizes M.P., Trouilhet S., *«Properties analysis of a learning algorithm for adaptative systems»,* First International Conference on Computing Anticipatory Systems, August 11-15 1997, Liège

[Camps, 1998a] Camps V., *« Vers une théorie de l'auto-organisation dans les systèmes multi-agents basée sur la coopération : application à la recherche d'information dans un système d'information répartie »,* Thèse de Doctorat de l'Université Paul Sabatier, IRIT, N° d'ordre 2890, Toulouse, 05 janvier 1998

[Camps, 1998b] Camps V., Glize P., Gleizes M.P., Léger A., Athanassiou E., Lakoumentas N., *« A Framework for Agent Based Information Brokerage Services in Electronic Commerce »,* EMSEC Conference 1998

[Carré, 1998] Carré J., *« Robots footballeurs auto-organisés : spécification et implémentation »* Rapport d'IUP SI Université Paul Sabatier Toulouse 1998

[Chavez, 1996] Chavez A., Maes P., *« Kasbah: An Agent Marketplace for Buying and Selling Goods »,* The First International Conference on the Practical Application of Intelligent Agents and Muli-Agent Technology, PAAM'96, London, UK, April 96

[Costa, 1996] Costa A.C.R., Demazeau Y., *« Toward a formal model of multi-agent systems with dynamic organizations »,* Proceedings of the Second International Conference on Multi-Agent Systems, AAAI Press/ The MIT Press, Kyoto, 1996.

[Demazeau 1993] Demazeau, Y., *« La plate-forme PACO et ses applications »,* in Actes des 2emes journees du PRC IA / SMA, Montpellier : CNRS-LIFIA, 1993.

[Demazeau, 1998] ICMAS 98, Paris, Y. Demazeau (ed), IEEE Press, 1998

[Dotto, 1996] Dotto F., *« Etude d'un algorithme génétique fondé sur le principe de la coopération »,* DEA RCFR – Université Paul Sabatier, Toulouse, 1996

[Drogoul, 1993] Drogoul A., *« De la simulation multi-agent à la résolution collective de problèmes »,* Thèse de doctorat de l'Université Paris VI – 1993

[Drogoul, 2000] Drogoul A., *« Systèmes Multi-Agents Situés »,* Habilitation à diriger des recherches Université de Paris VI, 2000

[Ferber, 1995] Ferber J., *« Les systèmes multi-agents »,* InterEditions 1995

FIPA www.fipa.org

[Finin, 1994] Finin T., Fritzon R., McKay D., McEntire R., *« KQML as an Agent Communication Language »,* Proceedings of the 3rdInternational Conference on Information and Knowledge Management (CIKM'94), 1994

[Foisel, 1996] Foisel R., Chevrier V., Haton J-P., *« De l'organisation d'une société à sa réorganisation »,* Actes Journée Systèmes Multi-Agents du PRC-GDR IA Toulouse 2 février 1996

[Foisel, 1997] Foisel R., Chevrier V., Haton J-P., *« Un modèle pour la reorganisation de système multi-agent »,* Actes des 5ièmes journées IAD-SMA pages 261-277 La Colle sur Loup Avril 1997

[Galliers, 1988] Galliers J. R., *« A Strategic Framework for Multi-Agent Cooperative Dialogue »,* Proceedings of European Conference on Artificial Intelligence, ECAI 88, Munich, August 1-5, pages 415-420

[Glize, 1999] Glize P., Gleizes M.P., Léger A., Einsiedler H.J., *« Abrose : Multi-Agent Based Brokerage Services in Electronic Commerce »,* Agent Link, Barcelona, Spain

[Glize, 2000] Glize P., *« L'adaptation des Systèmes à Fonctionnalité Emergente par Auto-Organisation Coopérative »,* Habilitations à diriger des recherches Université Paul Sabatier Toulouse Juin 2000

[Goldman, 1994] Goldman C.V., Rosenschein J.S., *« Emergent Coordination through the Use of Cooperative State-Changing Rule »,* AAAI 1994

[Guessoum, 1999] Guessoum Z., *« La plate-forme DIMA »,* AFIA n°39 Octobre 1999

[Gutknecht, 1998] Gutknecht, O., Ferber J., *« A meta-model for the analysis and design of organizations in Multi-Agent Systems »,* Proceedings of ICMAS'98, Demazeau Y. Editor, Paris, France. IEEE Computer Society, 1998

[Harrouet, 1997] Harrouet F., Cozien R., Reignier P., Tisseau J., *« ORIS: une plate-forme de simulation multi-agents réactifs »,* pages 81- 82 Actes des 5ièmes journées IAD-SMA pp 261-277 La Colle sur Loup Avril 1997

[Heurtevin, 1996] Heurtevin L., *« Un système multi-agent auto-organisateur non-cognitiviste pour la représentation de croyances »,* Rapport d'IUP SI Université Paul Sabatier Toulouse, Septembre 1996

[Hewitt, 1991] Hewitt C., *« Open information systems semantics for Distributed Artificial Intelligence »,* Special volume foundations of Distributed Artificial Intelligence, Elsevier Science Publishers B.V. Vol.47 1991

[Kuhn, 1998] Kuhn B., *« Plate-forme de simulation de fourmis fourrageuses : les outils graphiques et mathématiques »,* Rapport de DUT, Université Paul Sabatier, 1998

[Léger, 1998] Léger A., Glize P., Gleizes M.P., Camps V., *« ACTS-Abrose and FIPA - Preliminary studies FIPA»,* Osaka, Japan 1998

[Maes, 1994] Maes P., *« Agents that Reduce Work and Information Overload »,* Communications of the ACM Vol 37, 1994

[MARCIA, 1997] MARCIA, *« Auto-organisations := émergence de structures »,* Journée du PRC GDR IA Grenoble Mars 1997, Editons Hermès

[Maturana, 1980] Maturana H.R., Varela F.J., *« Autopoiesis and cognition - The realization of the living »,* D. Reidel Publishing Company - North Holland 1980

[Maturana, 1994] Maturana H. R., Varela F.J., *« L'arbre de la connaissance »,* Addison Wesley 1994

[Morisset, 1997] Morisset B., *« Etudes d'algorithmes auto-organisateurs pour le fourragement de fourmis »,* DEA Paris Dauphine Juin 1997

[Moukas, 1996] Moukas A., *« Amalthaea : Information Discovery and Filtering using a Multiagent Evolving Ecosystem »,* Conference on Practical Applications of Agents and Multiagent Technology, London, April 1996

[M R Jean, 1997] M. R. Jean : Nom collectif pour Batard, Brassac, Delépine, Gleizes, Glize, Labbani, Lenay, Marcenac, Magnin, Müller, Pesty, Quinqueton, Vidal, *« Emergence et SMA »,* Actes des cinquièmes journées francophones IAD&SMA, La Colle sur Editions Hermès, pages 323-341

[Nodine, 1998] Nodine M., Perry B. and Unruh A., *« Experience with the InfoSleuth Agent Architecture »,* Proceedings of AAAI-98 Workshop on Software Tools for Developing Agents

[O'Hare, 1996] O'Hare G.M.P, Jennings N.R., *« Foundations of distributed Artificial Intelligence »,* Sixth Generation Computer technology Series Wiley InterScience 1996

[Parunak, 1997] Parunak V. D. H., *« Go to the Ant: Engineering Principles from Natural Multi-Agent Systems »,* Annals of Operations Research 75, pages 69-101, Special Issue on Artificial Intelligence and Management Science

[Parunak, 2001] Parunak V. D. H., Brueckner S., *« Entropy and Self-Organization in Multi-Agent Systems »,* Proceedings of the International Conference on Autonomous Agents Agents 2001

[Pelletier-Prouls, 1999] Pelletier-Proulx D., *« Réalisation d'un logiciel d'analyse de résultats »,* DUT Université Paul Sabatier, Juin 1999

[Pesquet, 1999] Pesquet B., Gleizes M-P., Glize P., *« Une équipe de robots footballeurs auto-organisées : les SMACkers »,* Intelligence Artificielle Située, Cerveau, corps et environnement, Drogoul A. & Meyer J-A. Coordonnateurs. Editions Hermès, pages 243-256

[Pitt, 1999] Pitt J., Bellifeme F., *« A Protocol-Based Semantics for FIPA '97 ACL and its implementation in JADE »,* CSELT internal technical report. Part of this report has been also published in Proceedings of AI*IA, 1999.

[Pollack, 1990] Pollack M.E., Ringuette M., *« Introducing the Tileworld: Experimentally Evaluating Agent Architectures »,* Proceedings of the national conference on Artificial Intelligence, 1990

[Reicken, 1994] Reicken D. (Guest Ed.) *« Special Issue in Intelligent Agents »*, Communications of the ACM Vol 37, N°7 July 1994 ACM Press

[Russel, 1995] Russel S., Norvig P., *« Artificial Intelligence. A modern Approach »*, Prentice Hall Series in Artificial Intelligence

[Salle, 1997] Salle N., *« Plate-forme graphique de simulation d'insectes sociaux : les fourmis fourrageuses »*, DUT de l'Université Paul Sabatier Juin 97

[Sekaran, 1995] Sekaran M., Sen S., *« To Help Or Not To Help »*, Seventeenth Annual Cognitive Sciences Conference, Pitsburg, Pennsylvannia, July 22-25 1995

[Sen, 1995] Sen S., Sekaran M., *« Using reciprocity to adapt to others »*, International Joint Conference on Artificial Intelligence 1995

[Servat, 1999] Servat, D., Leonard, J., Perrier, E., Treuil, J.P., *« The Rivage Project : a new approach for simulating runoff dynamics »*, Modelling of transport processes in soils, 24-26 Novembre 1999, Leuven, J.Feyen & K.Wiyo.Eds, Wageningen Pers, The Netherlands, pages 592-601

[Sontheimer, 1999] Sontheimer T., *« Modèle adaptatif de prévision de crues par systèmes multi-agents auto-organisateurs »*, Rapport de stage Institut Universitaire Professionnalisé - Direction Régionale de l'Environnement Midi-Pyrénées Université paul Sabatier Toulouse 1999

[Steels, 1999] Steels, L., *« The Spontaneous Self-organization of an Adaptive Language »*, Koichi Furukawa and Donald Michie and Stephen Muggleton, editors, Machine Intelligence 15, pages 205 - 224, St. Catherine's College, Oxford: Oxford University Press.

[Tokoro, 1996] Tokoro M. Editor, ICMAS 96, Kyoto, AAAI & MIT Press, 1996

[Topin, 1998] Topin X., *« Réalisation d'une plate-forme de simulation pour l'étude du comportement des fourmis »*, Rapport d'IUP SI Université Paul Sabatier, Toulouse, Septembre 1998

[Topin, 1999a] Topin X., *« Etude de l'auto-organisation par coopération appliquée à l'apprentissage comportemental de robots fourmis »*, DEA RCFR – Université Paul Sabatier, Toulouse Juin 1999

[Topin, 1999b] Topin X., Fourcassié V., Gleizes M.P., Théraulaz G., Régis C., Glize P., *« Theories and experiments on emergent behaviour : From natural to artificial systems and back »*, Proceedings on European Conference on Cognitive Science, Siena 1999

[Weiß, 1996] Weiß G., *« Adaptation and Learning in Multi-Agent Systems »*, Lecture Notes in Artificial Intelligence 1042. Springer Verlag.

[Weiß, 1999] Weiß G., *« Multiagent Systems, A modern Approach to Distributed Artificial Systems »*, MIT Press 1999

[White, 1994] White J., *« Telescript Technology : The Foundation for the Electronic Marketplace »*, General Magic White Paper, 1994

[Wooldridge, 1995] Wooldridge M., Jennings N.R., *« A theory of cooperative problem solving »*, Intelligent Agents Workshop, pp. 15-26, Oxford, 23 Novembre 1995

[Wooldridge, 1996] Wooldridge M., Jennings N.R., (Guest Ed.) *« Special Issue on Intelligent Agents and Mulit-Agent Systems »*, Applied Artificial Intelligence Journal Vol 9 N°4 July/August 1995 and Vol 10 n°1 January/February 1996 Francis & Taylor

Workshops on Cooperative Information Agents: The proceedings of the CIA workshop series are published as volumes in the Springer series of Lecture Notes on Artificial Intelligence (CIA-97: LNAI 1202, CIA-98: LNAI 1435, CIA-99: LNAI 1652, CIA-2000: LNAI 1860, CIA-2001: LNAI 2182, CIA-2002: LNAI 2446, CIA-2003: LNAI 2782).

Workshops on Agent Mediated Electronic Commerce: The proceedings of the AMEC workshop series are published as volumes in the Springer series of Lecture Notes in Computer Science (AMEC-98: LNCS 1571 AMEC-99: LNCS 1788, AMEC-2000: LNCS 2003, AMEC-2001: LNCS 2531, AMEC-2002: LNCS 2531).

Chapitre III. La résolution de problèmes par émergence

1. Objectifs - Contexte

Cette partie liste les principales interrogations scientifiques qui guident mes travaux ainsi que ce qui les motive, et permet de positionner mes travaux par rapport aux problématiques prises en compte par la discipline.

1.1. Objectifs

L'évolution des domaines d'application de l'informatique demande la conception de logiciels de plus en plus complexes. Les systèmes multi-agents « classiques » ont apporté une nouvelle manière de concevoir de tels logiciels. En effet, le comportement du système est le résultat des interactions et du comportement des agents qui le constituent. A ce stade de la recherche dans ce domaine, pour permettre une diffusion plus large des systèmes multi-agents, il est nécessaire de fournir des modèles et des méthodes de conception de ces logiciels. Pour le concepteur, la tâche est simplifiée par le fait qu'il n'a plus à développer et à appréhender le système dans sa totalité. Son activité consiste à développer les agents qui sont des entités logicielles plus simples que le système global ainsi que les mécanismes d'interaction entre ces derniers.

Dans le cadre de mes travaux, une nouvelle manière ascendante de conception de ces systèmes a été développée basée sur l'auto-organisation par coopération. La notion d'auto-organisation est très liée à la notion d'émergence comme le souligne la définition donnée par Camazine et al. [Camazine, 2001], «l'auto-organisation est un processus dans lequel des structures au niveau global émergent seulement des nombreuses interactions entre des composants du système d'un niveau plus bas». L'auto-organisation est un mécanisme permettant d'obtenir une fonction émergente. Je suis persuadée, ainsi que ceux avec qui je travaille, que le domaine des systèmes multi-agents a besoin de théories de l'émergence

pour tenir ses promesses concernant la conception de systèmes distribués, composés d'un grand nombre d'éléments autonomes. L'autonomie au niveau d'un système signifie que le système dans son ensemble est capable de réagir à des sollicitations de son environnement sans intervention du concepteur [Boissier, 2003]. La théorie des AMAS présentée précédemment est une théorie de l'émergence, bien d'autres restent à découvrir au sein d'un nouvel axe de recherche dans le domaine des systèmes multi-agents. A partir de la définition de cette théorie, de nombreuses problématiques restent posées telles que : une spécification précise de la coopération, la validation expérimentale, la définition de l'émergence dans un système artificiel ou l'étude des propriétés de tels systèmes.

La notion d'émergence est étudiée depuis fort longtemps (les premiers travaux seraient ceux du grec Anaximandre) dans de nombreuses disciplines telles que la philosophie, les mathématiques, la physique, la chimie, la biologie… Actuellement ce concept est de plus en plus répandu dans des domaines qui s'intéressent à la conception de systèmes artificiels comme : la vie artificielle, les réseaux de neurones, les systèmes multi-agents et ce de manière floue. Dans la suite de mes travaux, une partie a été consacrée à la clarification de ce concept dans le domaine des systèmes multi-agents en essayant de répondre aux questions suivantes : quelle définition de l'émergence en informatique ? Quelles sont les caractéristiques d'un phénomène émergent ? Comment maîtriser l'émergence ? Comment passer du local au global ? Comment influencer au niveau local le comportement d'un agent sans programmer la manière d'obtenir le comportement global ? Nous avons donc travaillé sur ce concept dans le cadre de la thèse de Jean-Pierre Georgé dans le but de proposer une définition permettant de dissocier ce qui est émergent de ce qui ne l'est pas.

Mes travaux, dans cette période, portent donc essentiellement sur l'étude de l'auto-organisation et de l'émergence c'est pourquoi je suis co-animatrice[10] du groupe de travail TFG[11] Self-Organisation (www.irit.fr/TFGSO) d'AgentLink III (www.agentlink.org) dont la première rencontre a eu lieu en juin 2004.

En parallèle, de nouveaux projets tels que la conception de systèmes mécaniques, la détermination des relations fonctionnelles entre les gènes d'une cellule [Macchion, 2004], la détermination de la conformation spatiale d'une molécule… ont été réalisés dans le but de continuer à valider la théorie des AMAS sur de nouveaux problèmes et de montrer la faisabilité de l'approche pour la résolution de problèmes. Ces expérimentations permettent d'améliorer la manière de mener un développement en clarifiant et précisant, par exemple, la notion de coopération. Comment être sûr que le comportement est coopératif en se limitant aux traitements des situations non coopératives ?

Comme il a été exposé dans le chapitre II, nous avons développé de nombreux systèmes multi-agents adaptatifs en utilisant la théorie des AMAS : une équipe de robots footballeurs (SMACkers) [Pesquet,1999], des fourmis fourrageuses (ANTS) [Topin,1999b], un système contrôlant l'acheminement du trafic dans un réseau téléphonique [Dotto, 1999], un système de prévisions des crues (STAFF) [Sontheimer, 2002], [Régis, 2002] et un système de mise en correspondance de clients et

[10] Les autres responsables sont Anthony Karageorgos (Université de Thessaly – Grèce) et Giovanna Di Marzo Serugendo (Université de Genève, Suisse)
[11] TFG Technical Forum Group

de fournisseurs de services pour le commerce électronique (ABROSE) [Gleizes, 2000b]. Pour permettre à d'autres informaticiens, non compétents dans le domaine des systèmes multi-agents adaptatifs, de concevoir de tels systèmes, nous travaillons à l'élaboration d'un atelier de développement ADELFE[12] dédié aux systèmes adaptatifs. Son originalité réside dans le fait qu'il prend en compte les aspects dynamiques liés à l'ouverture du système ou à l'environnement. Les travaux antérieurs sur la théorie des AMAS ont guidé la réalisation d'une méthode de conception de ces logiciels. En effet, l'application du théorème suivant : « pour tout système fonctionnellement adéquat il existe au moins un système à milieu intérieur coopératif qui réalise une fonction équivalente » permet au concepteur de ne s'intéresser qu'à la conception des agents. La conception d'un système fonctionnellement adéquat revient à la conception d'un système à milieu intérieur coopératif, soit à la conception d'agents coopératifs. Cette manière de concevoir est ascendante.

Ce travail de conception de l'atelier a été réalisé dans le cadre du DEA de Jean-Christophe Casteran en 2000 et de la thèse de Gauthier Picard (2001 – soutenance prévue fin 2004) et a donné lieu à de nombreuses publications.

D'une manière plus générale, mes travaux s'intéressent à l'ingénierie des systèmes multi-agents et tendent à répondre aux questions suivantes :

- Qu'est ce qui différencie un objet d'un agent ?
- Doit-on utiliser des standards ? Doit-on créer une méthode de conception totalement nouvelle ? Que manque t-il aux méthodes orientées objet pour concevoir des systèmes multi-agents ? Doit-on couvrir tout le cycle de vie d'un logiciel ?
- Quel type d'outils doit-on offrir aux concepteurs ?
- Quel type de benchmark est le plus représentatif des systèmes multi-agents ?
- Comment comparer les différentes méthodes existantes ? Comment les associer ?

Ces questions sont prises en compte dans les groupes de travail auxquels je participe tant au niveau national, avec le groupe ASA de l'AFIA (www.afia.polytechnique.fr/) qu'au niveau international avec notre implication au sein du Special Interest Group MSEAS d'AgentLink II (polaris.ing.unimo.it/MSEAS/) et ma position de promoteur du Technical Forum Group AOSE dans le cadre d'AgentLink III (www.agentlink.org/). Je participe de plus, au comité technique de la FIPA (www.fipa.org) dédié aux méthodologies (www.pa.icar.cnr.it/~cossentino/al3tf1/).

1.2. Contexte

Parce que les travaux sur les systèmes multi-agents sont de plus en plus nombreux, il est difficile de donner une vue complète des recherches dans ce domaine, durant la période 1999-2004. C'est pourquoi, l'état de l'art qui suit ne sera d'une part relatif qu'aux travaux sur l'auto-organisation et l'émergence dans le domaine des systèmes multi-agents et, d'autre part qu'aux méthodes de développement orientées agents dans le cadre de l'ingénierie logicielle des systèmes basés sur des agents.

[12] Projet RNTL dont les partenaires industriels sont ARTAL Technologies et TNI-Valiosys, et les partenaires universitaires, IRIT et L3I.

1.2.1. Auto-organisation et émergence dans les systèmes artificiels

Les notions d'émergence et d'auto-organisation sont de plus en plus répandues dans le domaine informatique et notamment dans les Systèmes Multi-Agents. En effet, l'objectif du domaine est l'étude et la conception de systèmes complexes ayant un comportement global produit par l'interaction de nombreux agents ayant un comportement autonome. Cet objectif rejoint les problématiques étudiées déjà par d'autres disciplines telles que la philosophie, la physique, la thermodynamique, la cybernétique…

L'émergence dans les systèmes artificiels est très proche conceptuellement de la computation émergente définie par Stephanie Forrest en 1990 comme suit :

- une collection d'agents en interaction : le processus,
- un épiphénomène produit par ce processus au macro niveau et
- une interprétation naturelle de cet épiphénomène en tant que calcul ou résultat d'un calcul.

L'auto-organisation est, en général, associée à la notion d'émergence, c'est un mécanisme qui permet d'obtenir de l'émergence. Elle est donc définie comme un processus d'émergence spontanée d'ordre dans un système, dû à des relations internes au système et/ou à des relations avec son environnement et à la manifestation de ces relations dans l'écoulement du temps. Une définition intuitive de l'auto-organisation considère qu'il y a auto-organisation quand le système change son organisation sans commande externe explicite. L'auto-organisation est alors l'ensemble des processus au sein d'un système, issus de mécanismes basés sur des règles locales, qui conduisent ce système à produire des structures, des comportements ou des propriétés spécifiques non dictés (non explicites) par l'extérieur du système [Georgé 2004].

En 1993, la communauté française aborde cette problématique notamment avec les travaux au sein du groupe de travail « Collectif » de l'AFCET dont je faisais partie. Notre objectif était de tracer la voie d'une approche émergentiste des systèmes multi-agents, les travaux de ce groupe ont fait l'objet d'un article en 1997 [M. R. Jean, 1997]. En parallèle, le groupe de travail du PRC GDR SMA MARCIA, que j'animais avec Vincent Chevrier, travaillait sur l'auto-organisation car nous étions convaincus que c'était un moyen essentiel pour que les systèmes multi-agents puissent évoluer et s'adapter dans un environnement dynamique [MARCIA, 1997].

L'émergence peut alors être constatée dans des travaux de simulations, le phénomène émergent correspond à l'observation du comportement du collectif comme dans les travaux sur la plate-forme de simulation de fourmis MANTA [Drogoul, 1993], l'émergence d'œuvres d'arts [Hutzler, 1997], la formation de mares [Servat, 1999] ou l'émergence de formes ou de structures. Ce dernier type de phénomènes émergents est au centre des travaux de Grégory Beurier, Olivier Simonin et Jacques Ferber où des structures multi-niveaux émergent du comportement réactif des agents [Beurier, 2003].

L'émergence est aussi étudiée dans le but de faire faire à des systèmes artificiels des tâches que les humains réalisent et dont les mécanismes mis en œuvre ne sont pas encore complétement identifiés et compris comme l'acquisition d'un langage. Ce thème est abordé dans les travaux de Kaplan et Steels [Steels, 1999], [Kaplan, 2000].

Des travaux sont plus centrés sur la recherche de mécanismes permettant à un système de s'auto-organiser et d'exhiber des propriétés d'adaptation. De nombreux mécanismes sont inspirés des systèmes naturels et des travaux en biologie. Salima Hassas utilise la notion de stigmergie [Hassas, 2003]. Avec Foukia, ils ont réalisé un système de détection d'intrus basé sur un éco-système composé d'agents [Foukia, 2003]. Le comportement des agents pour répondre à une intrusion est inspiré des insectes sociaux, ils agissent par dépôt de phéromone pour signaler l'intrusion. Le mécanisme de détection de l'intrus s'inspire du fonctionnement du système immunitaire.

Vincent Chevrier s'inspire du comportement de systèmes naturels (les rats) dans le cadre de la résolution de problèmes [Chevrier, 2002]. Avec Christine Bourjot, ils ont aussi expérimenté le principe de stigmergie pour la reconnaissance d'images dans laquelle les agents se comportent comme les araignées sociales [Bourjot, 2003].

Zahia Guessoum et Alain Cardon se sont orientés vers un couplage entre une structure neuro-mimétique et une organisation d'agents pour concevoir des systèmes adaptatifs [Cardon, 2000] [Guessoum, 2003]

Au niveau de la communauté internationale des systèmes multi-agents, les travaux les plus anciens sur les notions d'émergence, d'auto-organisation, d'adaptation sont ceux de Van Parunak. Dans son article « Go to the Ant » en 1997 [Parunak, 1997a], il dresse les avantages à s'inspirer des systèmes naturels, et notamment des sociétés d'insectes sociaux, et il donne un ensemble de principes pour concevoir des systèmes artificiels complexes. En 2001, avec Sven Brueckner, il montre la relation qui existe entre la thermodynamique et les systèmes multi-agents et cherche à dégager des principes de conception pour les systèmes artificiels. Ils expliquent le couplage qu'il y a entre une réduction de l'entropie au niveau micro et une augmentation de celle-ci au niveau macro. Pour diminuer le désordre dans un système multi-agent, il le couple à un autre dans lequel le désordre augmente [Parunak, 2001]. En 2003, Brueckner et Parunak suivent les principes généraux de la conception d'applications en essaim. Ils appliquent ce style de programmation, couplé à l'utilisation du principe de stigmergie, pour gérer un réseau de nœuds, de clients et de serveurs mobiles [Brueckner, 2003].

Radhika Nagpal utilise l'auto-organisation dans le cadre d'un projet sur l'informatique diffuse. Le mécanisme d'auto-organisation est basé sur les principes d'auto-organisation dans les organismes multi-cellulaires [Nagpal, 2003].

Vasirani, Mamei et Zambonelli [Vasirani, 2003] étudient l'ingénierie de l'auto-organisation et proposent trois types d'approches. La première consiste à faire de la conception directe dans laquelle les concepteurs codent des algorithmes distribués en se basant sur des mécanismes comme la localisation et la diffusion. La deuxième approche s'apparente à de la rétro-conception, les concepteurs donnent le comportement des agents et les interactions. Ensuite, par simulation, ils observent le fonctionnement du système et, éventuellement, ils modifient les agents. La dernière approche consiste à utiliser les deux précédentes pour contrôler le comportement émergent.

Zambonelli et Mamei ont aussi proposé des outils pour faciliter l'auto-organisation des agents. L'auto-organisation est considérée comme une coordination entre les agents. Pour se coordonner les agents doivent se connaître et se trouver. Pour cela, une infrastructure TOTA est réalisée, elle comporte des nœuds sur lesquels les agents vont émettre des tuples et TOTA gère la maintenance des nœuds dans cet environnement [Mamei, 2003].

D'autres travaux sont relatifs à la conception c'est-à-dire comment donner des règles locales aux agents qui permettent à un comportement global souhaité d'émerger. Dans ce cadre Jean-Pierre Müller et ses collègues proposent un guide dans lequel le concepteur doit spécifier les schémas qui doivent se produire sans donner le comment et l'applique à des robots fourrageurs [Labbani, 1997]. Puis, il propose une méthode associée pour aider le concepteur qui comprend quatre étapes :

- décomposer la structure de l'espace de recherche en composants,
- déterminer les interactions qui produisent ces composants,
- déduire les agents et leur dynamique qui engendre ces interactions,
- compléter la spécification de l'environnement qui contient l'état courant de la recherche, les contraintes exogènes et sa propre dynamique [Müller, 1998], [Müller, 2001], [Müller, 2002]. Depuis 1995, mon équipe travaille aussi sur cette problématique.

Les travaux menés au sein de mon équipe se distinguent des travaux cités précédemment. Bien que les travaux de Maturana et Varela aient été une source d'inspiration pour nos travaux, nous ne cherchons pas à appliquer les modèles des systèmes naturels pour concevoir les systèmes. La voie que nous avons choisie est de trouver de nouveaux mécanismes permettant aux parties d'un système artificiel de s'auto-organiser. Ces nouveaux modèles peuvent s'éloigner de la manière dont fonctionnent les systèmes naturels. Nous pouvons aisément faire cela car nous ne sommes pas assignés comme les éthologues à avoir des modèles dont le fonctionnement correspond à des systèmes naturels. Nos travaux s'éloignent aussi des travaux de simulation. Notre objectif principal est que le système réalise une certaine tâche, résolve un problème alors que l'objectif premier d'une simulation est d'observer un système en fonctionnement en faisant varier éventuellement des données pour comprendre ou prévoir des phénomènes observés dans la nature.

Actuellement, les techniques de conception de systèmes dans lesquelles tout le comportement du système pouvait être prévu et décrit ne suffisent plus pour concevoir des applications qui sont devenues de plus en plus complexes. C'est pourquoi il est nécessaire de trouver de nouveaux modèles, de nouvelles théories, de nouveaux outils pour concevoir ces systèmes. La communauté scientifique des systèmes multi-agents marque, depuis 2003, sa volonté d'orienter les recherches en ce sens. Des workshops dédiés à l'auto-organisation comme le workshop ESOA[13] [Di Marzo, 2003] qui a eu lieu la première fois en juillet 2003, aux systèmes ouverts TAPOCS[14] (TAPOCS 2002 [Frederiksson, 2003] et TAPOCS 2003 [Frederiksson, 2004]), des sessions comme dans le workshop EUMAS[15] 2003 ont été créés. Dans les appels à communication des conférences, les thèmes d'auto-organisation, d'adaptation, d'émergence sont de plus en plus fréquents (cf JFSMA[16] en 2004).

[13] ESOA : Engineering Self-Organising Applications
[14] TAPOCS : Theory And Practice Of Open Computational Systems
[15] EUMAS : European Workshop on MultiAgent System
[16] JFSMA : Journées Francophones sur les Systèmes Multi-Agents

1.2.2. L'ingénierie logicielle

Après la révolution de la conception et de la programmation orientée-objet, nous sommes à l'aube d'une nouvelle révolution qui serait celle de la conception et de la programmation orientée-agent/interaction/organisation. Le constat concernant la conception de systèmes multi-agents, dans les années 1998-1999, est que le développement de systèmes multi-agents est coûteux en temps à cause de leur complexité, mais aussi par le fait que pour chaque application, il faut créer tout le système multi-agent adéquat. En effet, la majorité des applications existantes en SMA sont développées de manière ad hoc [Treur, 1998]. Ce foisonnement a conduit en parallèle à de multiples propositions de modèles d'agents, notamment par le rapprochement effectué avec l'approche objet. De là, plusieurs formalismes sont apparus, chacun mettant en avant une représentation de l'agent et de son système. En 1999, se fait donc sentir le besoin de fournir des modèles, des méthodologies, des plates-formes pour faciliter la prise en compte de la complexité des systèmes à concevoir [Jennings, 1999] et pour aider les concepteurs qui ne sont pas nécessairement spécialistes des systèmes multi-agents. Les industriels sont aussi fortement intéressés par une approche d'ingénierie basée sur leurs exigences qui prennent en compte tout le cycle de vie du logiciel [Bussman, 1998]. Les premiers travaux sur les méthodologies sont ceux sur AAII [Kinny, 1996], sur Cassiopée [Collinot, 1996], sur DESIRE [Brazier, 1999]. En 2000, cet axe des systèmes multi-agents se confirme et les aspects ingénierie des systèmes multi-agents représentent alors le thème de groupes de travail au niveau national : PRC-GDR, I3-AFIA[17], ou européen : SIG MSEAS d'AgentLink (2000-2003), ainsi que le thème de conférences ou de workshops (AOIS[18], AOSE[19] 2000 à 2004), ESOA[20] (2003 à 2004), JFIADSMA'99 [Gleizes, 1999], JFIADSMA'00 [Pesty, 2000], MAAMAW'99. De nombreuse méthodologies sont conçues [Wooldridge, 2001], [Tveit, 2001]. Les principales sont Gaia [Zambonelli, 2000], [Cernuzzi, 2004], MaSE [Wood, 2000], [Deloach, 2001], PASSI [Cossentino, 2002], INGENIAS/MESSAGE [Caire, 2001], [Gomez Sanz, 2002], Tropos [Giunchiglia, 2002], Prometheus [Padgham, 2002], Voyelles [Demazeau, 2001]… Les deux grandes familles de travaux concernant les méthodes orientées agent étendent les concepts soit vers des méthodes orientées objet, soit vers des méthodes issues de l'ingénierie des connaissances [Iglesias, 1999],

Actuellement, il est de plus en plus difficile - sinon impossible - de contrôler correctement l'activité de ces logiciels situés dans des environnements de plus en plus dynamiques. Pour faire face à ces difficultés, une solution consiste à laisser plus d'autonomie aux logiciels afin qu'ils s'adaptent au mieux aux imprévus et donc concevoir des systèmes multi-agents adaptatifs. En effet, les systèmes multi-agents classiques prennent bien en compte la complexité des interactions, mais la dynamique n'est que très peu présente [Casteran, 2000b], les systèmes multi-agents adaptatifs répondent, quant à eux, à ce besoin. C'est l'une des raisons essentielles pour lesquelles les méthodes de conception classiques de systèmes multi-agents ne sont pas totalement utilisables pour concevoir des systèmes adaptatifs [Gleizes, 1999], [Cardon, 2000], [Parunak, 2001] et c'est pourquoi les travaux sur une méthode adaptée, ADELFE, ont commencé en 2000. ADELFE a des points communs avec les méthodes de la même génération telles que :

[17] AFIA : Agence Française de l'Intelligence Articicielle
[18] AOIS : Agent Oriented Information Systems
[19] AOSE : Agent Oriented Software Engineering
[20] ESOA : Engineering Self-Organising Systems

MASSIVE, PASSI ou Tropos, notamment de couvrir le cycle de vie complet du logiciel. ADELFE fournit aussi un outil graphique de modélisation comme les méthodes DESIRE, INGENIAS/MESSAGE[21], MaSE, PASSI ou Prometheus. Par contre, ADELFE comme Tropos prend en compte la dynamique du système ou de l'environnement. Sa spécificité réside dans le fait qu'elle est spécialisée pour concevoir des systèmes multi-agents adaptatifs en se basant sur la théorie des AMAS, alors que la plupart des méthodes sont générales. De plus, ADELFE est l'une des rares méthodologies proposant des outils interactifs afin d'aider l'utilisateur à la mettre en oeuvre. En effet, un premier outil interactif permet au concepteur de suivre l'application du processus et un second lui permet, en cours d'analyse, de décider s'il est utile d'appliquer la théorie des AMAS pour concevoir son application. Actuellement, de nombreuses méthodologies ont été développées [Bergenti, 2004] et les travaux s'orientent vers la conception de méta-modèles et vers la prise en compte des phases de développement de test et de déploiement [Garcia, 2003].

2. La notion d'émergence

L'originalité de la résolution de problèmes avec les AMAS est qu'elle s'articule autour de trois notions essentielles : apprentissage, auto-organisation et émergence. La résolution de problèmes par une approche émergente et la notion d'émergence sont de plus en plus étudiées dans la communauté des systèmes multi-agents. Il nous est donc apparu nécessaire d'essayer de définir de manière plus précise les caractéristiques d'un phénomène émergent et la notion même d'émergence en informatique, pour ensuite pouvoir juger si le résultat d'un logiciel peut ou non être qualifié d'émergent. Ce travail avait été initié dans le groupe de travail Colline et avait donné lieu à un article aux JFIADSMA de 1997 [M. R. Jean, 1997], il a été continué dans l'équipe notamment dans le cadre de la thèse de Jean-Pierre Georgé [Georgé, 2003a], [Georgé 2004].

2.1. Caractérisation du phénomène émergent

Les phénomènes émergents sont observés et étudiés depuis la Grèce antique, ils ont d'abord été caractérisés puis des systèmes ont été conçus pour construire de tels phénomènes. Le phénomène peut être une propriété, une structure, un comportement. L'origine de l'émergence pourrait bien être le postulat : "le tout est plus que la somme de ses parties" [Ali, 1997], [Holland, 1997]. Ses débuts sont surtout caractérisés par la distinction que fait G. H. Lewes entre résultant et émergent. Ce philosophe anglais explique en 1875 que, pour le résultant, la séquence d'étapes qui produisent un phénomène est traçable, alors que pour l'émergent, nous ne pouvons pas tracer les étapes du processus. Ainsi, nous ne pouvons pas voir dans le produit le mode d'opération de chaque facteur. L'émergence ne peut pas être réduite ni à la somme, ni à la différence des forces coopérantes.

[21] http://grasia.fdi.ucm.es/ingenias

Deux points de vue se complètent pour caractériser un phénomène émergent : le point de vue de l'observateur du phénomène qui considère l'aspect statique de l'émergence, le résultat et le point de vue du concepteur du système qui va produire le phénomène qui considère l'aspect dynamique de l'émergence, le processus.

Selon le premier point de vue, l'émergence est définie afin de reconnaître un phénomène émergent et de le différencier de phénomènes explicables grâce à d'autres théories ou modèles. Les propriétés inter-reliées, communes qui permettent d'identifier le phénomène comme émergent sont :

1. L'observation d'un phénomène ostensible au niveau global. Observer un phénomène ostensible au niveau macro ou global signifie que ce phénomène s'impose à l'observateur et que rien au niveau des composants du système ne permettait de le prédire [Atlan, 2000]. Churchland définit l'émergence en termes d'irréductibilité des propriétés associées à une théorie de haut niveau, à des propriétés associées à des composants dans une théorie de plus bas niveau [Ali, 1997].

2. L'observation de nouveauté. La nouveauté radicale du phénomène (il n'est pas observé au niveau micro et n'est pas prévisible). Pour Van de Vijver il y a apparition de nouveauté que ce soit au niveau des propriétés, des structures, des formes ou des fonctions [Van De Vijver, 1997]. Pour Lewes (1874), la combinaison d'unités d'un certain ordre réalise une entité d'ordre supérieur dont les propriétés sont entièrement nouvelles [Grumbach, 1997].

3. L'observation de cohérence et de corrélation du phénomène. La cohérence et la corrélation du phénomène correspondent au fait qu'il a une identité propre mais liée fortement aux parties qui le produisent. Selon Goldstein, l'émergence fait référence à l'apparition durant le processus d'auto-organisation dans un système complexe de structures ou de schémas ("patterns") ou de propriétés nouvelles et cohérentes [Goldstein, 1999].

4. L'observation d'une dynamique. Observer une dynamique particulière revient à dire que le phénomène n'est pas connu au départ et qu'il y a "auto-maintien" du phénomène. Langton (1989) définit l'émergence en termes de relation de feedback entre les niveaux dans un système dynamique. Les micro-dynamiques locales causent les macro-dynamiques et les macro-dynamiques globales contraignent les locales [Odell, 2001].

En adoptant le point de vue concernant la construction d'un phénomène émergent, le système doit présenter les caractéristiques suivantes :

1. La non linéarité. La linéarité permet une traçabilité analytique des interactions locales jusqu'au phénomène global. Il n'y a donc pas apparition de nouveauté radicale et donc pas d'émergence possible. Il faut donc que les parties du système interagissent de façon non linéaire pour pouvoir parler d'émergence. Ainsi, une partie du système doit pouvoir être influencée par une autre partie même si elle ne lui est pas directement rattachée. On peut, par exemple, utiliser des boucles de feedback négatif ou positif qui sont non linéaires par nature.
2. Un état proche de l'équilibre. Au début, les théories cherchaient à comprendre comment des systèmes tendent vers un état final d'équilibre ou homéostasie. La théorie de la complexité s'intéresse plus aux conditions autour de l'équilibre qui entretiennent

l'émergence. L'amplification des événements aléatoires est une des raisons clés pour lesquelles l'émergence fait apparaître des caractéristiques imprédictibles.

3. L'auto-organisation. Dans la théorie des premiers systèmes, l'auto-organisation faisait référence au processus d'autorégulation. En théorie de la complexité, on fait référence au comportement créatif et auto-généré qui produit les changements, ainsi qu'à la recherche d'adaptabilité d'un système complexe. Le système doit être capable de se modifier d'une façon ou d'une autre afin de changer de comportement en fonction de son environnement.
4. Des attracteurs. Toujours dans la théorie des premiers systèmes, le seul attracteur valide était celui qui amenait à un état final d'équilibre. Le fait d'utiliser un seul type d'attracteur simple avec comme finalité un état d'équilibre, conduit à la prévisibilité du système. En théorie de la complexité, il y a de nouveaux types d'attracteurs (le point fixe, le cycle limite et l'attracteur étrange). Ces attracteurs ne sont pas prédonnés dans le sens de la "Gestalt". Ils ne dictent donc pas au système l'état à atteindre mais lui donnent seulement les moyens de changer [Goldstein, 1999].
5. De la dynamique. La non-linéarité et les nouveaux attracteurs permettent cette métamorphose quantitative et qualitative du système si particulière. Tout doit donc être fait pour permettre au système d'exhiber un dynamisme suffisant pour en profiter et ainsi, tout en se maintenant dans des états proche/loin de l'équilibre, permettre l'auto-organisation qui conduit au phénomène émergent.

2.2. Une définition du phénomène émergent en informatique

Malgré l'effort de recherche mené, l'émergence reste un phénomène peu connu et on pourrait même dire un peu mystérieux. Nous proposons dans le cadre de la conception de systèmes informatiques, une définition de la construction d'un phénomène émergent par un logiciel informatique. Cette définition "utilitaire" [Georgé, 2004] est décomposée en trois parties : ce que l'on veut faire émerger, à quelle condition il y a émergence et comment nous nous en servons.

Objet	Un système informatique a pour finalité de réaliser une fonction adéquate à ce que l'on attend du système. C'est cette fonction, pouvant évoluer au cours du temps, que nous voulons faire émerger.
Condition	Cette fonction est émergente si le codage du système ne dépend aucunement de la connaissance de cette fonction. Ce codage doit contenir des mécanismes permettant l'adaptation du système au cours de ses échanges avec l'environnement afin de tendre à tout instant vers la fonction adéquate.
Méthode	Dans la théorie des AMAS, pour changer la fonction il suffit de changer l'organisation des composants du système. Ces mécanismes sont spécifiés par des règles régissant l'auto-organisation entre les composants et ne dépendant pas de la connaissance de la fonction collective.

Il est en général entendu que le phénomène qui émerge correspond à la fonction exécutée par le système informatique [Müller, 2002]. La condition pour qualifier ce phénomène d'émergent est que ne soit pas codé dans les agents la manière d'obtenir la fonction globale. Un agent ne doit pas connaître la manière d'obtenir le but global. Il est à noter que les agents peuvent accéder ou connaître ce résultat global, c'est ce que jean,-Pierre Müller appelle l'émergence forte [Müller, 2001]. Cette notion d'émergence forte est différente de la notion d'émergence forte de Searle [Searle, 1995], pour lequel l'émergence forte correspond à la présence au niveau global d'une propriété irréductible aux propriétés du niveau local. Dans la définition proposée, nous différencions donc le résultat du système et la manière d'obtenir ce résultat.

Les systèmes multi-agents adaptatifs conçus selon la théorie des AMAS possèdent plusieurs propriétés permettant de situer cette théorie dans les théories de l'émergence :

- Que l'application matérielle de cette théorie s'incarne ou non par des agents, elle se situe dans le cadre de la théorie des systèmes.
- Un système coopératif dans l'environnement est fonctionnellement adéquat, ce qui lui évite de connaître la fonction globale qu'il doit réaliser pour s'adapter.
- Même si un système n'a pas de but, il peut agir pertinemment dans son milieu. En fonction de ses perceptions de l'environnement, des représentations qu'il en possède et de ses compétences, il agira au mieux pour que son comportement soit coopératif.
- La notion de rétroaction n'est pas contraignante dans cette théorie car le système doit seulement juger si les changements s'opérant dans le milieu sont coopératifs de son point de vue sans savoir si ces changements sont dépendants de ses propres actions passées.

Ce travail sur la notion d'émergence a renforcé la manière dont on doit prendre en compte les problèmes à résoudre c'est-à-dire en se focalisant sur le comportement des parties du système et non sur le système. Les parties du système ne doivent donc pas contenir la manière dont le problème doit être résolu au niveau global.

2.3. La résolution de problèmes par émergence

De manière classique les méthodes de résolution de problèmes reposent sur la formalisation a priori du domaine du problème. Dans les domaines dynamiques et incertains cette formalisation devient difficile et requiert des capacités adaptatives [M. R. Jean, 1997]. Dès 1995, nous étions convaincus que l'émergence devait fournir une approche alternative pour la résolution de problèmes. La méthode classique de résolution de problèmes classique consiste à décomposer le problème en sous-problèmes et à donner les étapes de la résolution, le concepteur code le chemin que va parcourir le processus de résolution lors de son exécution. Dans l'approche émergente, le processus de résolution doit construire le chemin qui l'amènera à la solution. Le programmeur code les agents, l'environnement et les interactions.

Dans le cadre de nos travaux le moyen mis en œuvre pour opérationnaliser l'émergence pour en faire une méthode de conception au niveau des agents, repose sur la

coopération (décrite dans le chapitre II de ce mémoire). Dans le cas où la résolution de problèmes est vue comme un ensemble d'états initiaux et finaux et un ensemble d'opérateurs pour atteindre un état final à partir d'un état initial, la coopération est considérée comme une heuristique pour parcourir l'espace d'états.

La notion de coopération dans le cadre de la théorie des AMAS a donc été définie comme la capacité d'un agent à travailler avec un autre agent, mais aussi de détecter et de traiter les situations non coopératives, ou les erreurs de coopération, en cours de fonctionnement. L'attitude coopérative de l'agent est donc « curative » c'est-à-dire qu'il agit pour traiter des problèmes de coopération. Nos travaux actuels sur la théorie des AMAS ont eu pour conséquence de compléter cette attitude par une attitude prescriptive qui prévient les erreurs de coopération en renforçant des actions coopératives comme, par exemple, agir en essayant de gêner le moins possible les autres.

Le fait de donner de l'autonomie aux agents, dans le sens que chaque agent détermine les actions qu'il va exécuter à partir de son état et de l'état de l'environnement qu'il perçoit sans commande extérieure explicite [Parunak, 2001]), peut entraîner un comportement global non souhaité, il est donc nécessaire, dans le cadre de la conception de systèmes artificiels, de donner des moyens de maîtriser cette émergence. Dans le cadre de nos travaux, ce contrôle passe par l'environnement. En effet, le système multi-agent réagit aux interactions qu'il possède avec son environnement et cherche à être en interaction coopérative avec lui.

Dans les premières applications que nous avons développées, comme ANTS ou le Tileworld, les agents du système réagissent à l'environnement dynamique et l'environnement contraint le résultat global. En effet dans l'application ANTS, si aucune nourriture n'est déposée, le taux de nourriture ramenée au nid sera nul et le rendement ne sera pas bon dans cet environnement, le système n'y peut rien. Dans ce cas, l'environnement ne contrôle pas la résolution du système, l'environnement est subi par le système.

Pour la résolution de problèmes par émergence, le rôle que joue l'environnement est donc fondamental et les interactions, sous la forme de feed-backs en réaction au comportement du système, doivent permettre de contrôler le comportement global du système. C'est pourquoi, les actions de l'environnement sur le système ont été particulièrement étudiées au sein de la programmation émergente, par exemple. Le travail du concepteur consiste donc, comme le souligne Van Parunak [Parunak, 1997a], à coder les agents, les interactions mais aussi l'environnement.

2.4. L'environnement de programmation émergente : EPE

L'objet du projet de recherche EPE (Emergent Programming Environment) est de faciliter la conception de programmes complexes. En effet, lorsque la complexité du programme à écrire croît, le programmeur a de plus en plus de difficulté à le produire. Cet axe de recherche a été essentiellement mené par Jean-Pierre Georgé dans le cadre de ses

travaux de DEA en 1999 et de thèse, et une description plus détaillée peut être consultée dans [Georgé, 2004].

Dans ce mémoire, EPE permet d'illustrer d'une part, le rôle fondamental de l'environnement dans son interaction avec le système pour la résolution de problèmes, et d'autre part, la nécessité de compléter le comportement d'un agent coopératif. C'est dans le cadre de ce projet que des mesures ont été effectuées pour vérifier s'il y avait une correspondance entre la proximité organisationnelle et la proximité fonctionnelle, ce qui a permis de donner d'autres critères pour concevoir un agent coopératif.

2.4.1. Objectifs du projet EPE

La programmation émergente peut être définie comme une technique qui permet au programme de se construire sans que le programmeur ait à définir la suite d'instructions qui le constituent. Partant de cette image, on peut définir ce type de programmation de manière plus précise. La programmation émergente a pour objet de fournir un moyen de faire une recherche automatisée efficace pour explorer l'espace des organisations possibles qui représentent l'ensemble des programmes possibles [Georgé, 2004]. La programmation peut donc être vue comme une résolution de problèmes au sens classique du terme. Notre motivation dans ce projet était de montrer la faisabilité de l'approche des systèmes multi-agents adaptatifs basée sur l'émergence pour la résolution de ce type de problèmes.

Les objectifs plus concrets du projet EPE consistent d'une part, à mettre en place un environnement de programmation pour aider le programmeur à participer au codage de son programme et d'autre part, à concevoir le système qui va effectivement réaliser le bon agencement des instructions. L'environnement de programmation n'est pas décrit dans ce mémoire, seuls les agents et l'environnement du système sont détaillés. De manière naturelle, les agents manipulés sont les instructions d'un langage de programmation. Dans ce cadre, l'émergence est utilisée pour la conception de programmes. Le principe de l'émergence correspond à l'apparition d'un tout cohérent à partir d'interactions locales non explicitement informées de la manière d'obtenir ce tout cohérent. En effet, au niveau local, celui de l'instruction, il est impossible de connaître et de coder la fonction globale du système c'est-à-dire le programme. Le programme doit être le résultat d'un travail collectif entre les instructions.

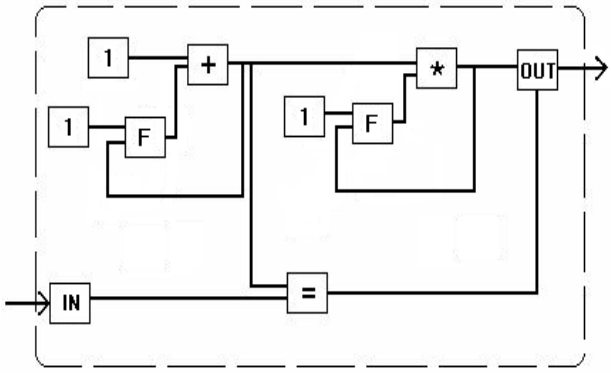


Figure 29 - Mécanisme itératif de la factorielle. Cette organisation des agents calcule la factorielle pour n > 2.

L'environnement du système de programmation émergente est constitué par le programmeur (appelé néo-programmeur). Dans le cadre de ce travail, le rôle et le comportement de l'environnement qui permet de guider le système vers sa fonction

adéquate ont été particulièrement étudiés. Le rôle du néo-programmeur est de juger du résultat du système et ensuite d'interagir de manière à provoquer un changement de comportement du système.

Parce que coder un programme de plusieurs milliers de lignes est un problème difficile, dans un premier temps, le problème a été abordé dans un cadre plus restreint pour faciliter sa compréhension et son observation. Dans le projet EPE, les instructions ont été limitées aux instructions du calcul mathématique et le programme à réaliser correspond à une fonction mathématique simple. Dans un premier temps, la fonction factorielle a été mise en œuvre puis nous nous sommes intéressés à la fonction (A+B) * C.

2.4.2. Description

Un agent « instruction » possède l'architecture d'un agent coopératif définie précédemment dans la théorie des AMAS :

- Des compétences : c'est la capacité, propre à chaque instruction, d'agir suivant sa spécification sémantique.
- Des représentations (accointances) : ce sont les connaissances que possède un agent à propos d'autres agents. La connaissance qu'un agent a d'autres agents est locale, c'est-à-dire qu'un agent ne connaît pas tous les agents du système.
- La capacité à communiquer et à interagir avec les autres agents. Un agent est schématisé par une boîte noire recevant des données sur ses entrées et produisant des données en sortie. Pour des agents instructions, le moyen de communiquer est l'envoi de messages : les données produites seront envoyées à l'agent dont l'entrée correspond à la sortie de celui qui a produit la donnée. Ce lien de travail est exprimé par les accointances.
- Une attitude sociale qui est la coopération.

Chaque fois qu'un agent est mis en contact avec un agent qu'il ne connaissait pas il le rajoute à ses accointances. Cette action correspond à la mise en place d'un lien entre les deux agents. Ainsi, l'organisation du système à un instant donné correspond à sa topologie. L'organisation est modifiée localement par les agents lors de la détection de situations non coopératives et de leur traitement.

Le néo-programmeur, assimilé à l'environnement, interagit avec le système de la manière suivante : chaque fois que le système lui fournit un résultat non souhaité, le néo-programmeur ne pouvant rien faire de ce résultat, enverra une situation non coopérative au système. Le système et le néo-programmeur seront en interaction coopérative quand le système fournira le résultat souhaité au néo-programmeur. L'environnement joue donc un rôle fondamental, c'est lui qui contrôle en quelque sorte le comportement du système. Toute la difficulté réside dans la définition du feed-back que l'environnement doit renvoyer. Dans le cadre d'un calcul mathématique, le feed-back peut prendre trois formes :

- le résultat est juste ou faux,
- le résultat est trop grand ou trop petit,
- donner la distance qui sépare le résultat donné du résultat solution.

2.4.3. Le système factorielle

Les agents utilisés pour réaliser la fonction factorielle sont : un agent chiffre 1, les opérateurs numériques + et *, l'opérateur d'égalité, l'opérateur de choix FirstTime et deux agents d'interface In et Out. In est l'agent qui recevra du néo-programmeur, la valeur dont le système doit calculer factorielle. Out est l'agent qui recevra le feed-back du néo-programmeur. Chaque agent est doté d'entrées et de sorties. Le comportement des agents, de manière générale, est le suivant :

- Dès qu'un agent est disponible pour effectuer sa tâche, il le fait savoir à ses entrées.
- Lorsqu'il a reçu les informations dont il a besoin, il effectue son calcul et envoie le résultat à ses accointances en sorties lorsqu'elles sont prêtes à recevoir.
- Il est ensuite disponible pour le calcul suivant.
- Les agents traitent ensuite les situations non coopératives (SNC) qui proviennent soit de l'environnement, soit d'un autre agent du système. Pour changer l'organisation au sein du système, les agents modifient les liens qui les unissent à d'autres agents, appelés partenaires. Pour cela, ils gèrent des taux de confiance en leurs partenaires.

La partie la plus importante et la plus difficile de la conception revient à donner aux agents un comportement coopératif en détectant et traitant les situations non coopératives (SNC). Deux groupes de SNC ont été définis : d'une part, les SNC endogènes détectées par l'agent qui sont liées à un problème de partenariat et d'autre part, les SNC liées à l'environnement qui concernent les interactions du système avec son environnement. Elles sont liées à un problème de résultat.

Les traitements associés à ces SNC consistent en des envois de messages permettant soit de propager les SNC, soit de modifier les taux de confiance en des partenaires. Cette dernière action peut entraîner la rupture d'un lien entre deux agents.

Pour choisir un nouveau partenaire, un agent ayant reçu des propositions de partenariat de la part d'autres agents peut choisir parmi cette liste de propositions celle qui l'intéresse le plus, afin de créer un partenariat entre lui et son choix. Il y a un mécanisme de négociation par messages pour que les agents arrivent à se synchroniser afin d'établir un partenariat.

Le choix du partenaire potentiel à contacter se fait en comparant les taux de confiance mais aussi en évitant de rester trop longtemps dans les mêmes configurations ou de revenir dans celles déjà parcourues. Dans la mesure du possible, l'agent va privilégier des agents avec lesquels il n'a encore jamais eu de partenariat.

2.4.4. Résultats et analyse de factorielle

Les expérimentations pour factorielle ont été réalisées dans les conditions suivantes. Le type et le nombre d'agents nécessaires (10) pour réaliser factorielle sont donnés au système. Le travail du système consiste donc à trouver la bonne organisation entre les agents sachant que le mécanisme ne prend pas en compte les cas 0 et 1 et que le feed-back utilisé est de type booléen c'est-à-dire que le résultat donné par le système est juste ou faux.

Malgré l'efficacité des agents d'EPE à traiter les SNC endogènes, la stratégie du traitement du feedback de l'environnement ainsi qu'un certain nombre de variantes, se sont révélées décevantes. Tous les mécanismes techniques pour explorer l'espace des organisations possibles sont bien en place, mais cette exploration ne tend pas vers une organisation fonctionnellement adéquate. En effet, la fonction est discontinue et le type de feed-back utilisé ne permet pas de guider le système. Les agents ne peuvent donc pas parcourir l'espace de recherche par "petits pas" en jugeant de la pertinence du collectif pour choisir la direction à prendre.

Il a fallu trouver d'autres critères sur lesquels les agents puissent s'appuyer pour orienter leur comportement. Une première orientation du travail a consisté à étudier les organisations obtenues, dans le cadre de factorielle, pour vérifier qu'il y avait une correspondance entre l'organisation et la fonction réalisée.

Une deuxième modification a été de donner aux agents un feed-back plus informé sur le résultat attendu (cf partie 2.4.5). Bien entendu, pour rester dans le cadre de la définition de l'émergence, il n'est pas question de leur donner d'informations sur le moyen d'obtenir ce résultat.

Un troisième axe de travail a été d'approfondir le travail sur le comportement coopératif d'un agent (cf partie 2.4.5). Actuellement, un agent pour être coopératif dans la théorie des AMAS, traite les situations non coopératives qu'il détecte, fait de la communication spontanée et de la relaxation de situations non coopératives. Jusqu'à présent, ce comportement associé à l'attitude coopérative était suffisant pour les problèmes traités.

2.4.5. Calcul élémentaire

Suite aux résultats mitigés obtenus avec factorielle, pour apporter les modifications au niveau du feed-back et du comportement coopératif des agents, nous avons choisi d'étudier une fonction mathématique plus simple que factorielle. Plus simple dans le sens où le nombre d'agents est moins élevé, ce qui facilite l'observation du comportement du système global et permet de mieux le comprendre. Le fonction mathématique élémentaire est la suivante (A+B)*C. Les agents sont les agents constantes 2, 10 et 100 et les agents + et *, et le système doit produire la valeur la plus grande possible.

L'espace de recherche est ici de 6^5 soit 7 776 organisations possibles. 120 organisations sont complètes, 24 sont fonctionnelles et 6 sont différentes en termes de résultats (en enlevant les organisations où les permutations des entrées ne changent pas les résultats). Pour la conception de ce système, l'objectif est double, d'une part, améliorer le feed-back et d'autre part, améliorer le comportement coopératif des agents

Feed-back informé

Le feed-back devient ici un feed-back plus informé car l'environnement transmet l'information « plus grand » ou « plus petit ». Il est à noter que cela ne fournit pas aux agents la manière d'obtenir le résultat. Ceci a entraîné une modification des SNC qui indiquent s'il faut fournir une valeur plus petite ou plus grande.

Compléments apportés au comportement coopératif

La communication spontanée est ajoutée en plus du traitement des SNC. Après une réorganisation, la valeur que l'agent est capable de fournir est mise à jour et cette valeur est spontanément communiquée aux partenaires de l'agent. De même, un agent qui perd un partenaire et ne peut plus produire de valeur en avertit ses partenaires. Un agent qui a atteint son but, c'est-à-dire qu'il produit une meilleure valeur, avertit aussi ses partenaires.

Le traitement que les agents ont à effectuer pour traiter une SNC doit aussi être le plus coopératif possible. Une réflexion à ce niveau a permis de rajouter, à la définition de l'attitude coopérative d'un agent, que ce dernier doit en agissant faire en sorte de gêner le moins possible les autres agents. Dans le cadre de la fonction mathématique élémentaire, cela revient à gérer des préjudices et à choisir un partenaire de sortie en faisant en sorte de gêner le moins possible les autres, (voir l'algorithme de traitement coopératif du préjudice dans la thèse de JP Georgé). Il y a deux sortes de préjudices : le préjudice de valeur et le préjudice indirect de situation. Le préjudice de valeur correspond à la situation où les entrées fournissent des informations moins « bonnes » pour l'agent que celles que l'agent possédait. Le préjudice indirect de situation permet à un agent d'indiquer à un partenaire, pour une entrée donnée, si le lien est unique ou multiple.

2.4.6. Résultats et analyse

Les expérimentations ont montré que lorsque le système est dans un état initial quelconque (sans aucun lien par exemple, ou toute autre configuration), le système parcourt moins d'une centaine d'organisations sur les 7 776 possibles pour atteindre une des 4 organisations produisant la valeur la plus élevée. On peut considérer que cette stratégie permet un parcours pertinent de l'espace de recherche. Mais, le plus intéressant est que quelle que soit la configuration recevant un feedback demandant de fournir une meilleure valeur, la prochaine valeur fournie à l'environnement sera effectivement meilleure car le système aura atteint une configuration produisant une meilleure valeur (si cette configuration existe).

Les modifications apportées au feed-back et au comportement coopératif ont permis d'obtenir un système qui converge vers la fonction mathématique désirée.

2.5. Système d'aide à la conception mécanique : SYNAMEC

L'approche que nous préconisons pour la résolution de problèmes par émergence a été utilisée pour l'aide à la conception de systèmes physiques. Pour concevoir un mécanisme qui réalise une certaine fonction, les concepteurs partent habituellement d'un mécanisme existant dont ils connaissent l'équation et apportent les modifications nécessaires en ajustant les paramètres de l'équation. Ce type de conception est très long et requiert des journées de simulations. Dans le cadre du projet SYNAMEC[22], l'objectif est de proposer une aide à la conception de mécanismes pour optimiser le temps de cette conception. La conception peut être vue comme le parcours d'un ensemble de mécanismes (espace d'états) jusqu'à l'obtention d'un mécanisme adéquat et donc comme une résolution

[22] SYNAMEC : SYNthesis of Aeronautical MEChanism

de problèmes. Cette tâche est complexe car le concepteur ne connaît pas a priori le nombre de composants mécaniques nécessaires et une modification de la topologie du mécanisme change complètement la fonction qu'il réalise : l'espace de recherche est discontinu et il n'existe pas d'heuristique pour le parcourir.

Ce projet a été financé par l'Europe dans le cadre du programme "Competitive and Sustainable Growth Program", et a été réalisé partenariat avec SAMTECH (B), INTEC (Argentine), CRANFIED (UK), ALENIA (I), SABCA (B), SNECMA (F). Il fait l'objet du travail de thèse de Davy Capera et a été suivi par Pierre Glize et moi-même. Ce système est présenté dans ce mémoire d'une part, pour montrer que la préoccupation de conception est totalement centrée sur les parties du système et non sur le système lui-même et d'autre part, pour caractériser le résultat obtenu par le système. Une description détaillée de ces travaux peut être trouvée dans [Capera, 2004a], [Capera, 2004b]. Tout comme le projet EPE, SYNAMEC fait partie des systèmes de résolution de problèmes par approche émergente.

2.5.1. Description générale

Le mécanisme à concevoir est implémenté par un système multi-agent et évolue dans un environnement spécifique généralement défini par une trajectoire, une enveloppe et un mouvement de contrôle. Pour guider la résolution, l'environnement doit être capable de calculer les mouvements des pièces mécaniques (interaction entre les composants, collisions avec l'enveloppe, etc.) et de retourner les informations relatives à chaque agent du système. Dans le contexte de cette application, le feed-back reçu par chaque agent est informé puisqu'un agent reçoit des informations concernant sa position, les forces qui s'exercent sur lui et pas seulement sur l'exécution correcte (ou pas) du mouvement. C'est le moteur de simulation cinématique MECANO qui joue le rôle de cet environnement en retournant les forces exercées par le mécanisme sur l'enveloppe et en donnant la distance à la trajectoire.

Le logiciel appelé MSS (Mechanical Synthesis Solver) qui permet au système multi-agent de concevoir le mécanisme, a un fonctionnement cyclique composé de trois phases :

- Le moteur de simulation calcule le mouvement du mécanisme courant.
- Les données relatives au nouvel état du mécanisme sont récupérées et communiquées aux agents mécaniques pour mettre à jour leur état.
- Le système multi-agent adaptatif met en œuvre « l'optimisation » : les agents détectent localement et résolvent au mieux les situations non coopératives. Cette étape conduit à un nouveau mécanisme qui est mû dans la phase 1.

Chaque composant mécanique élémentaire de la bibliothèque de composants devient un agent coopératif qui gère ses propriétés mécaniques (par exemple, une barre gère sa longueur). Dans le système, les agents sont les suivants : charnières, barres, points d'attachement, point trajectoire. L'agent « point trajectoire » est une exception : il ne représente aucun composant mécanique. Son but est d'être en permanence sur la trajectoire. Il peut percevoir la distance qui le sépare éventuellement de la trajectoire souhaitée et interprète toute distance non nulle comme une situation non coopérative.

Ce système est basé sur la théorie des AMAS, ainsi pour chaque agent et de manière exhaustive, les situations non coopératives qu'il peut rencontrer ont été déterminées. A ces situations ont été associées des actions correctrices telles que des changements de paramètres internes aux agents (ajustements de longueur, par exemple) et/ou des modifications de la place des agents dans le mécanisme (changements topologiques). Les conseils d'experts dans le domaine de la mécanique ont été nécessaires pour la définition de ces situations non coopératives qui guident le processus d'auto-organisation. Ce travail a été réalisé en collaboration avec l'université de CRANFIELD.

Les situations non coopératives peuvent être classées en trois types : incompréhension, incompétence et inutilité et peuvent être consultées dans [Capera 2004a], [Capera, 2004b]. Je ne les présente pas dans ce mémoire. Seul un exemple est donné ci-dessous concernant l'agent « Barre ».

Nom	Inutilité
Description	L'agent est totalement inutile pour le mécanisme
Condition(s)	L'agent n'est connecté à aucun autre agent
Action(s)	Trouver un agent de type joint pour se connecter Ou créer un nouvel agent de type joint Ou se suicider

Tableau 3 - Situation d'inutilité pour un agent « Barre »

De manière évidente, un composant mécanique non connecté au reste du mécanisme est totalement inutile. Dans ce cas, l'agent « Barre » essaie de s'intégrer dans le fonctionnement en recherchant des agents de type joint (c'est-à-dire une charnière, un point d'attchement) avec lesquels se connecter ou bien (en cas d'échec) il disparaît du système. Cette situation d'inutilité est en fait définie de la même manière pour tous les agents et le nombre d'actions à envisager est fonction des types d'agent avec lesquels chacun peut se connecter (par exemple, les actions de liaison ou de création d'un agent charnière concernent des corps rigides seulement).

Le processus d'auto-organisation pour la définition d'un mécanisme peut démarrer à partir d'un mécanisme vide en spécifiant au système uniquement les points trajectoires. Ainsi, quatre phases sont **observables** durant le processus d'auto-organisation :

- la création d'un composant à partir de l'ensemble des composants génériques disponibles,
- la réorganisation des composants au cours de laquelle les composants peuvent modifier leur position dans le mécanisme,
- l'ajustement des composants au cours duquel les composants peuvent modifier leurs caractéristiques comme leur longueur pour un agent barre,
- la disparition d'un composant.

2.5.2. Résultats et analyse

Les deux principaux mécanismes sur lesquels a été validé MSS sont les systèmes X-bars [Capera, 2004b] et le mécanisme de la tuyère (« Nozzle »)[Capera, 2004a].

Le Système 5-bars

Un système 5-bars doit permettre à un point de suivre une trajectoire (le point rouge sur la figure 30). Le mécanisme est composé de 5 barres, 4 charnières et 2 points d'attachement. Le problème consiste à bien ajuster la longueur des barres. A l'état initial le bon nombre de barres, les charnières et les points d'attachement sont donnés. La résolution correspond à déterminer la bonne dimension des composants rigides.

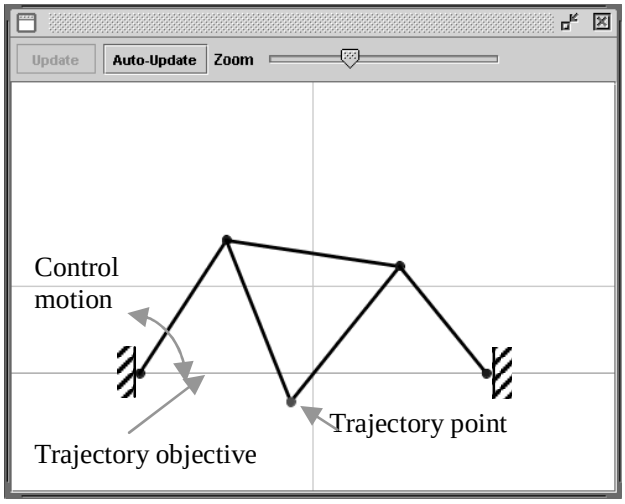


Figure 30 - Etat initial d'un mécanisme 5-barres mal dimensionné

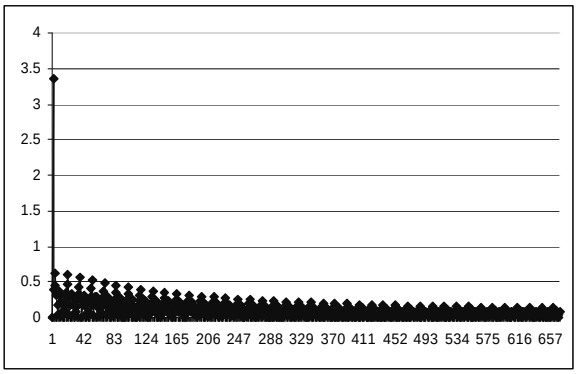


Figure 31 - Résultat du processus d'auto-organisation d'un mécanisme 5-barres

La figure 31 montrent les résultats obtenus par le système au cours du temps. Les ordonnées représentent la distance entre le point trajectoire et la trajectoire souhaitée et les abscisses le nombre de pas de simulations. Le système converge vers une solution car la distance à la trajectoire atteint des valeurs proches de zéro.

La tuyère

La tuyère se compose de deux composants majeurs : les volets primaires (volets chauds) et les volets secondaires (volets froids) reliés les uns aux autres au moyen de mécanismes articulés, le tout étant actionné par des vérins. Parmi les volets primaires, on distingue les volets commandés, des volets suiveurs situés entre deux volets commandés. Les volets secondaires sont asservis au mouvement des volets primaires. La conception d'une tuyère consiste à définir les mécanismes permettant de contrôler par le mouvement des volets primaires et secondaires, les sections d'éjection des gaz d'un turboréacteur au bord de fuite de ces volets, afin de lui assurer dans le domaine de vol la meilleure efficacité possible.

Pour résoudre ce problème, deux phases vont être mises en place par le système MSS : la phase de définition de la topologie et la phase d'ajustement des composants. Ces phases n'ont pas été programmées par le concepteur, c'est le système qui en fonction de l'état du mécanisme les a exécutées. Les composants mécaniques nécessaires au mécanisme final ne sont pas donnés au départ. L'état initial du mécanisme est représenté figure 32.

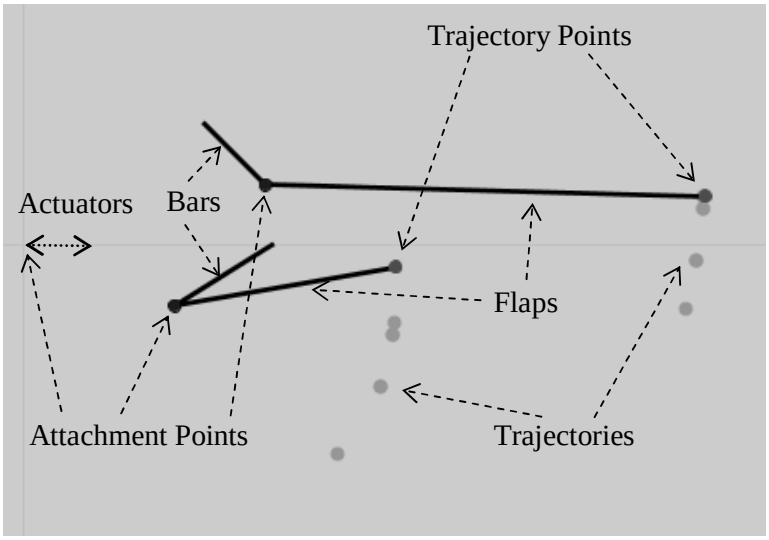


Figure 32 – Etat initial du problème de la tuyère

Le système commence par définir la topologie du mécanisme c'est-à-dire il détermine combien le système comprend de corps rigides et de quelle manière ils sont liés. Après la première étape de fonctionnement du cycle, représentée figure 33, le système a la bonne topologie. En tant qu'observateur on peut voir que le système a évolué vers un nouveau mécanisme.

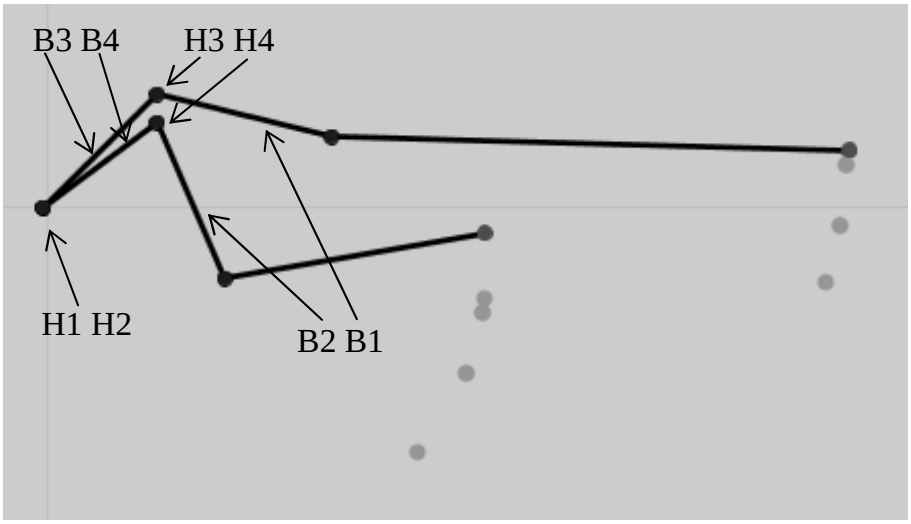


Figure 33 – Topologie du mécanisme après la première phase de fonctionnement

Après un certain nombre d'étapes le système remplit bien sa fonction qui était de passer par les points trajectoire. Un observateur peut interpréter ces étapes comme de l'ajustement du système (figure 34).

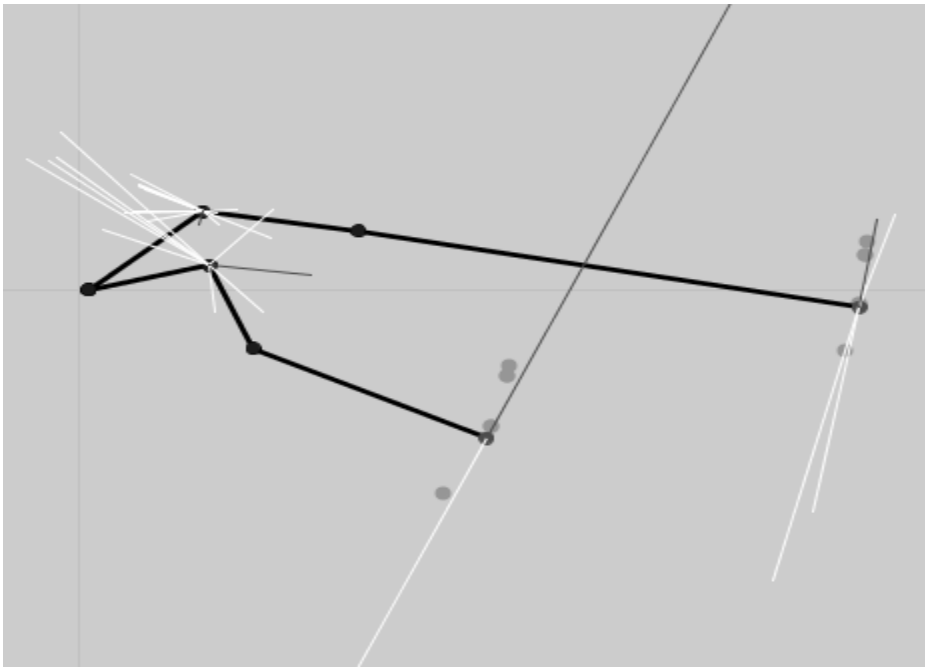


Figure 34 – Mécanisme après les phases d'ajustement

De manière étonnante, dans les applications basées sur la théorie des AMAS, nous n'avons jamais rencontré de problèmes liés aux attracteurs locaux (extrema locaux) dans lesquels tomberait le processus d'adaptation. Un explication (mais non une preuve) pourrait être l'inexistence d'une fonction globale de coût pour les théories (comme celle des AMAS) permettant l'émergence de la fonction globale: puisque la fonction n'est pas a priori connue, il ne peut exister de fonction de coût.

2.5.3. Caractéristiques de SYNAMEC

Les résultats obtenus par le MSS peuvent être qualifiés d'émergents et ceci est justifié par le fait que le mécanisme obtenu et sa propriété de suivi de trajectoire, vérifient les caractéristiques d'un phénomène émergent détaillés en 4.2.1 soient :

- l'observation d'un phénomène ostensible au niveau global,
- l'observation de nouveauté,
- l'observation de cohérence et de corrélation du phénomène,
- l'observation d'une dynamique.

Pour un observateur qui regarde le système évoluer, le mécanisme notamment dans le cas du moteur, est ostensible. En effet, l'observation des composants de manière individuelle et de leur propriété physique ne permet de connaître ou de prédire le mécanisme final. Sachant qu'il n'y a, dans aucun des agents du MSS (niveau local), un comportement visant directement à faire décroître la distance à la trajectoire (niveau global). La nouveauté est observée au niveau de la modification physique du mécanisme que ce soit par la modification de sa topologie ou de ses dimensions. Il y a bien une auto-organisation des composants mécaniques avec sa dynamique propre et la formation du mécanisme qui permet de voir l'identité du mécanisme.

Tous ces constats permettent de qualifier le mécanisme produit par MSS de phénomène émergent.

2.6. Contributions

Ce projet a permis de mettre en évidence que la notion de feed-back était plus ou moins simple à mettre en œuvre suivant les caractéristiques des applications.

La programmation émergente met en évidence le fait que le feed-back de l'environnement est fondamental pour guider le comportement du système. Dans le cas de factorielle, ce retour très peu informé n'est pas suffisant pour que le système produise le bon résultat. Dans le cas de l'exemple de la fonction mathématique simple, le supplément d'information « plus grand plus petit » permet au système de converger vers la solution.

Dans le cas de la fonction simple, le système étant dans un état initial quelconque nécessite le parcours de moins d'une centaine d'organisations pour trouver la solution. Rappelons qu'il y a 7776 organisations possibles. Au vu du problème général de la programmation émergente, ces travaux représentent une première avancée, un premier résultat pour montrer la faisabilité de la conception d'un système de programmation émergente réalisée à l'aide de systèmes multi-agents adaptatifs.

3. Méthode de conception de systèmes à fonctionnalité émergente

Depuis fin 2000, nous développons la méthode ADELFE : Atelier de DEveloppement de Logiciels à Fonctionnalité Emergente. ADELFE a fait l'objet d'un projet RNTL[23] (2001-2003) dont j'étais responsable pour l'IRIT. Les partenaires sont le laboratoire L3I de l'université de La Rochelle, et les entreprises ARTAL Technologies de Toulouse et TNI Valiosys de Brest. Nous avons collaboré avec l'équipe de génie logiciel de l'IRIT et notamment avec les collègues Thierry Millan, Jean-Paul Bodeveix et Christian Percebois. Ce travail a fait l'objet du DEA de Jean-Christophe Casteran [Casteran, 2000a], d'une partie du DRT de Sylvain Peyruqueou [Peyruqueou, 2002], il fait actuellement l'objet de la thèse de Gauthier Picard pour laquelle j'ai demandé une autorisation d'encadrement à 100%. Les permanentes impliquées dans ce projet sont Carole Bernon et moi-même. Notre objectif est de pouvoir transférer notre savoir-faire dans la conception de systèmes adaptatifs à des concepteurs informatiques non spécialistes de ce type de logiciels.

Une méthodologie doit permettre de faciliter le processus d'ingénierie des systèmes. Booch [Booch, 1992] donne la définition suivante d'une méthodologie : "une méthodologie est un ensemble de méthodes appliquées tout au long du cycle de développement d'un logiciel, ces méthodes étant unifiées par une certaine approche philosophique générale". Pour Shehory et Sturm [Shehory, 2001], une méthodologie est constituée de guides qui couvrent tout le cycle de vie du développement d'un logiciel. Certains sont des guides techniques et d'autres permettent de gérer le projet.

Une méthode est définie comme "un processus rigoureux permettant de générer un ensemble de modèles qui décrivent divers aspects d'un logiciel en cours de développement en utilisant une certaine notation bien définie". Une méthode de développement de systèmes multi-agents est constituée d'un processus, d'une notation et d'outils pour supporter ce processus et ces notations et/ou pour aider le développeur (CAME : Computer Aided Method Engineering tool) [Arlabosse, 2004]. La méthode ADELFE fournit un processus, une notation et des outils associés. Dans un souci d'intégration dans le monde industriel, elle repose sur des outils et notations couramment utilisés, voire normalisés. Dans ADELFE, nous disposons donc de :

- un processus de développement basé sur le RUP et, plus particulièrement, le processus développé dans le cadre du projet Européen Neptune [Neptune, 2001]. Une notation basée sur UML[24] et AUML[25] ,
- l'outil OpenTool[26], enrichi pour prendre en compte les besoins des systèmes adaptatifs, qui est un outil de modélisation graphique mettant en œuvre UML similaire à Rational Rose,
- une bibliothèque de composants permettant des simulations et un prototypage rapide,
- un outil d'aide à la décision pour que le concepteur puisse décider si un système adaptatif est nécessaire pour son application,

[23] RNTL : Réseau National des Technologies Logicielles
[24] UML : Unified Modelling Language
[25] AUML : Agent UML
[26] Développé par la société TNI-Valiosys

- un outil interactif qui met en œuvre le processus et guide ainsi le concepteur.

3.1. Le processus

Le processus d'ADELFE est calqué sur le processus utilisé dans le projet Européen Neptune [Neptune, 2001] qui lui-même suit le RUP (Rational Unified Process [Jacobson, 1999]. Un concepteur peut naviguer d'une étape à une autre afin de la compléter ou de l'enrichir. Néanmoins, ADELFE s'intéresse à la conception de logiciels bien spécifiques, présentés ci-dessus, et ne constitue pas une méthode générale comme, par exemple, Gaia [Zambonelli, 2000] ou Tropos [Castro, 2001]. C'est pourquoi certaines étapes du RUP ont été modifiées et d'autres ajoutées afin de prendre en compte les spécificités des systèmes multi-agents adaptatifs. Ce paragraphe décrit succinctement le processus d'ADELFE. Ce processus a été exprimé en SPEM (Software Process Engineering Metamodel) [Gleizes, 2003] et le vocabulaire utilisé dans la description du processus en dérive : on utilisera ainsi les termes : « définitions de travail » (WorkDefinitions ou WDi), Activités (Aj) et d'Étapes (Steps ou Sk). Sur la figure 35, les activités ou étapes spécifiques à la conception de systèmes multi-agents adaptatifs sont notées en gras. Dans ce paragraphe, seules ces activités ou étapes sont détaillées par rapport aux définitions de travail les englobant, les autres restent inchangées par rapport au RUP et à une conception orientée objet classique.

Besoins préliminaires. Les besoins préliminaires forment un ensemble d'activités dédiées à l'expression des besoins par l'utilisateur. Il en résulte un cahier des charges consensuel entre développeur et client. Ces activités ne diffèrent en rien du RUP classique.

Besoins finals. Les besoins finals correspondent à l'étude de l'environnement du système à réaliser ainsi qu'à ses interactions avec les utilisateurs. Étant donné que le couplage entre environnement et système, dans la théorie des AMAS, est une condition nécessaire à l'auto-organisation des agents, l'étape de **caractérisation de l'environnement** est primordiale pour la bonne suite de la conception. Cet environnement peut se caractériser par le fait qu'il est accessible ou non, déterministe ou non, dynamique ou statique ou discret ou continu. Cette caractérisation, issue de [Russel, 1995] permet de pointer du doigt les différents problèmes auxquels le système devra faire face dès les phases préliminaires de conception.

Une autre activité, l'**identification des échecs de coopération** entre le système et son environnement, a pour but d'aider le concepteur à détecter les problèmes liés au principe de coopération au sens de la théorie des AMAS. Cette identification s'affinera tout au long du processus et permettra, lors de l'analyse, d'identifier les agents.

Analyse. À partir des spécifications obtenues lors des besoins finals, l'analyste peut effectuer une analyse du domaine, identifier les différentes entités en jeu et ainsi construire un diagramme de classe préliminaire.

Il devient alors raisonnable de se poser la question de la pertinence de l'utilisation de la technologie AMAS pour la conception du système à étudier. En effet, toutes les applications ne nécessitent pas son utilisation ; c'est le cas, par exemple, si un algorithme efficace est déjà connu pour résoudre le problème. L'activité de **vérification de l'adéquation des AMAS** propose au concepteur un certain nombre de critères afin de l'aider à déterminer cette adéquation au niveau global (système), ce qui revient à déterminer si la technologie AMAS est nécessaire pour implanter le système, et niveau local (composantes du système) afin de déterminer si certaines composantes du système ont besoin d'être modélisées comme des AMAS, i.e. si une décomposition récursive est nécessaire.

Certaines méthodes orientées agent, comme Gaia ou TROPOS, présupposent que toute entité intervenant dans le système est un agent ; d'autres, comme AAII [Kinny, 1996] MESSAGE [Caire, 2001] ou PASSI [Cossentino, 2002], essaient d'identifier ces agents. ADELFE ne fait aucune hypothèse sur la nature des acteurs identifiés lors des besoins finals et guide le concepteur pour qu'il décide si ces entités seront ultérieurement vues comme des agents ou comme de simples objets. L'**identification des agents** est donc une activité caractéristique qui repose sur une analyse des dépendances fonctionnelles et des possibles échecs de coopération à partir des cas d'utilisation jusqu'à leurs diagrammes de séquence. En traçant les problèmes de coopération du niveau global au niveau des composantes ainsi qu'en étudiant leurs propriétés (autonomie, interaction, …), il est possible de déterminer quelles sont les entités susceptibles d'être des agents au sens de la théorie des AMAS.

Enfin, l'**étude des relations inter-agents** permet de modéliser les relations entre agents à partir de diagrammes de séquence pour les aspects événementiels ou bien à partir de diagrammes de classe pour les aspects taxonomiques ou organisationnels statiques.

Conception. Le processus de conception consiste à définir une architecture détaillée pour le système en termes de paquetages, de sous-systèmes, d'objets et d'agents. Ce sont des activités importantes d'un point de vue multi-agent du fait qu'une caractérisation récursive du système multi-agent est obtenue à ce stade. Cela implique différents processus de conception pour les différents niveaux du système identifiés.

Durant cette phase, le RUP a été augmenté de trois activités qui permettent au concepteur d'étudier le langage d'interaction des agents, de mettre en place une architecture agent et de réaliser un prototypage rapide. **Étudier les langages d'interaction** correspond à spécifier, grâce à des diagrammes de protocoles AIP (Agent Interaction Protocol) [Odell, 2000], les protocoles qu'utiliseront les agents afin de communiquer. Ces protocoles sont rattachés à des rôles et sont donc génériques ; ils seront instanciés à des classes d'agents lors de l'activité suivante qui consiste à **concevoir les agents**. ADELFE est dédiée à un type d'agents particulier : les agents coopératifs. Ainsi un modèle d'agent est fourni au concepteur pour lui permettre de doter tout agent des composantes traduisant son comportement : langage d'interaction, représentations du monde (de lui, des autres et de son environnement), compétences, aptitudes et SNC auxquelles l'agent peut avoir à faire face. Ces SNC sont essentielles pour traduire l'attitude sociale coopérative d'un agent. Elles dépendent de l'application et doivent être décrites par le concepteur. Pour le guider dans cette démarche, ADELFE propose des SNC génériques (telles que l'incompréhension, l'ambiguïté, l'inutilité, le conflit ou la concurrence) et un canevas général pour les décrire. Enfin, avant d'achever la conception, le concepteur a la possibilité

de **prototyper rapidement** les agents afin de détecter un manque quelconque de compétence, d'aptitude ou autre composante par une simulation.

Actuellement, ADELFE s'arrête à la fin de la conception, laissant les concepteurs libres d'utiliser des méthodes de leur choix pour l'implantation et le déploiement, en prenant celle du RUP par exemple. Dans le futur, ADELFE proposera la couverture complète du processus de développement de logiciels des besoins à la maintenance.

3.2. Le langage de modélisation

Le langage UML (Unified Modeling Language) s'est, depuis quelques années, imposé comme une norme de facto, pour exprimer le cycle de vie d'un logiciel dans le domaine objet. C'est pourquoi nous avons choisi d'utiliser cette notation et, en tant que support progiciel de cette notation, d'utiliser plus précisément, l'outil OpenTool de TNI Valiosys. Des extensions d'UML pour le domaine agent ont déjà été conçues, nous les utiliserons donc si cela est possible, il s'agit d'AUML [Odell 2001]. Dans ADELFE, UML a été enrichi pour avoir des capacités de description adaptées aux agents appartenant à des systèmes multi-agents adaptatifs. En effet, l'interface d'une classe est statique en programmation par objets et donc en UML, alors qu'elle est dynamique pour un agent. Le spectre d'utilisation de la notation UML peut être étendu d'une part par la notion de stéréotype, et d'autre part par la définition de nouveaux types de schémas pouvant être spécifiés au niveau du méta-modèle décrivant la notation UML elle-même. Ces différentes extensions justifient en partie le fait qu'UML tende à devenir la notation de référence dans de nombreux domaines (commerce électronique, bases de connaissances,...).

Pour prendre en compte les spécificités des systèmes multi-agents adaptatifs, deux solutions ont été considérées : soit utiliser un profil UML, soit créer une extension du métamodèle. Habituellement, cette dernière solution est adoptée lorsque les concepts du domaine sont bien définis. Actuellement, différents points de vue existent à propos des systèmes multi-agents. Par exemple, dans ADELFE, nous utilisons un concept spécifique de systèmes multi-agents adaptatifs où les agents doivent s'auto-organiser. Mais dans Tropos ou Gaia, les rôles des agents sont bien connus et l'organisation peut être *a priori* donnée. Aussi, nous avons retenu la solution des profils UML pour étendre la notation.

Dans ADELFE, neuf stéréotypes [Picard, 2004] peuvent être utilisés pour modifier la sémantique des classes, des attributs, des méthodes ou des associations pour répondre aux spécificités des agents coopératifs. Les stéréotypes dans ADELFE sont les suivants : « cooperativeAgent », « action », « perception », « interaction », « skill », « representation », « aptitude », « characteristic », et « cooperation ». Seul le stéréotype « cooperation » est présenté ci-dessous. Le premier tableau regroupe la description textuelle du stéréotype et ses contraintes d'utilisation. Le deuxième tableau s'inscrit dans la norme UML (pour la définition de nouveaux stéréotypes). Tous les stéréotypes sont décrits de manière complète dans [Bernon, 2002a].

Stéréotype « cooperation »

Stéréotype « cooperation »
Description L'attitude sociale coopérative des agents est mise en œuvre par l'application de règles de résolution de situations non coopératives. Dans le module de Coopération l'agent doit posséder un ensemble de règles (prédicats) permettant la détection de situations non coopératives, ainsi qu'une méthode de résolution de situation non coopérative associant aux situations des actions à mener afin d'en sortir. Une méthode stéréotypée « cooperation » est toujours privée et peut être de deux types : - soit une méthode renvoyant un booléen permettant la détection d'une situation en ayant passé en paramètre des perceptions, des représentation ou des compétences, - soit une méthode de résolution (a priori une seule pour un agent) permettant d'affecter à chaque situation une ou plusieurs actions de résolution. Ces méthodes sont uniquement appelées lors de la phase de décision de l'agent.
Règles d'utilisation - une méthode stéréotypée « cooperation » est privée ;

Stereotype	Base Class	Parent	Tags	Constraints	Description
cooperation	Feature	N/A	N/A	Cf. table ci-dessus	Cf. table ci-dessus

3.3. Les outils

ADELFE est doté de trois outils associés au processus et au langage de modélisation [Bernon 2003] Le premier est basé sur l'outil commercialisé par TNI Valiosys : OpenTool, enrichi pour prendre en compte le développement de systèmes adaptatifs. Le deuxième est un outil d'aide à la décision qui aide le concepteur à décider si l'utilisation de la théorie des AMAS est pertinente pour son application. Le dernier outil est un outil interactif qui guide le concepteur pour suivre le processus et exécuter les tâches associées.

3.3.1. OpenTool

OpenTool est un outil de développement, écrit dans le langage OTScript, qui a été conçu et distribué par TNI-Valiosys, un des partenaires d'ADELFE. D'une part, OpenTool est un outil graphique commercialisé comme Rational Rose qui supporte la notation UML pour modéliser des applications et assurer que ces modèles sont justes. Plus particulièrement, il se focalise sur l'analyse et la conception de logiciels objets écrits en Java. D'autre part, OpenTool permet la méta-modélisation afin de concevoir des configurations spécifiques. Cette dernière caractéristique a été utilisée afin de prendre en compte les spécificités de la conception de systèmes multi-agents adaptatifs. Ainsi, la notation AUML permettant de décrire les protocoles d'interaction entre agents a été intégrée à OpenTool. De plus, les neuf stéréotypes définis pour concevoir les agents ont été ajoutés au logiciel.

3.3.2. Outil d'aide à la décision de l'adéquation des AMAS

Un type de programmation utilisant la théorie des AMAS peut parfois être complètement inutile. Ce sera le cas si l'algorithme requis pour résoudre la tâche est déjà connu, si la tâche n'est pas complexe ou si le système est fermé et que rien d'inattendu ne

peut arriver. C'est pourquoi l'activité A11 a été ajoutée dans la phase d'analyse. ADELFE fournit donc à l'analyste un logiciel lui permettant d'étudier l'adéquation des AMAS à son système pour savoir si un AMAS est utile pour implanter le système étudié. Ce logiciel comporte une série de 11 questions lui permettant d'analyser la pertinence d'utiliser un système multi-agent adaptatif.

3.3.3. Outil interactif de suivi du processus de développement

ADELFE fournit un outil interactif aidant le concepteur à suivre le processus établi dans la méthode

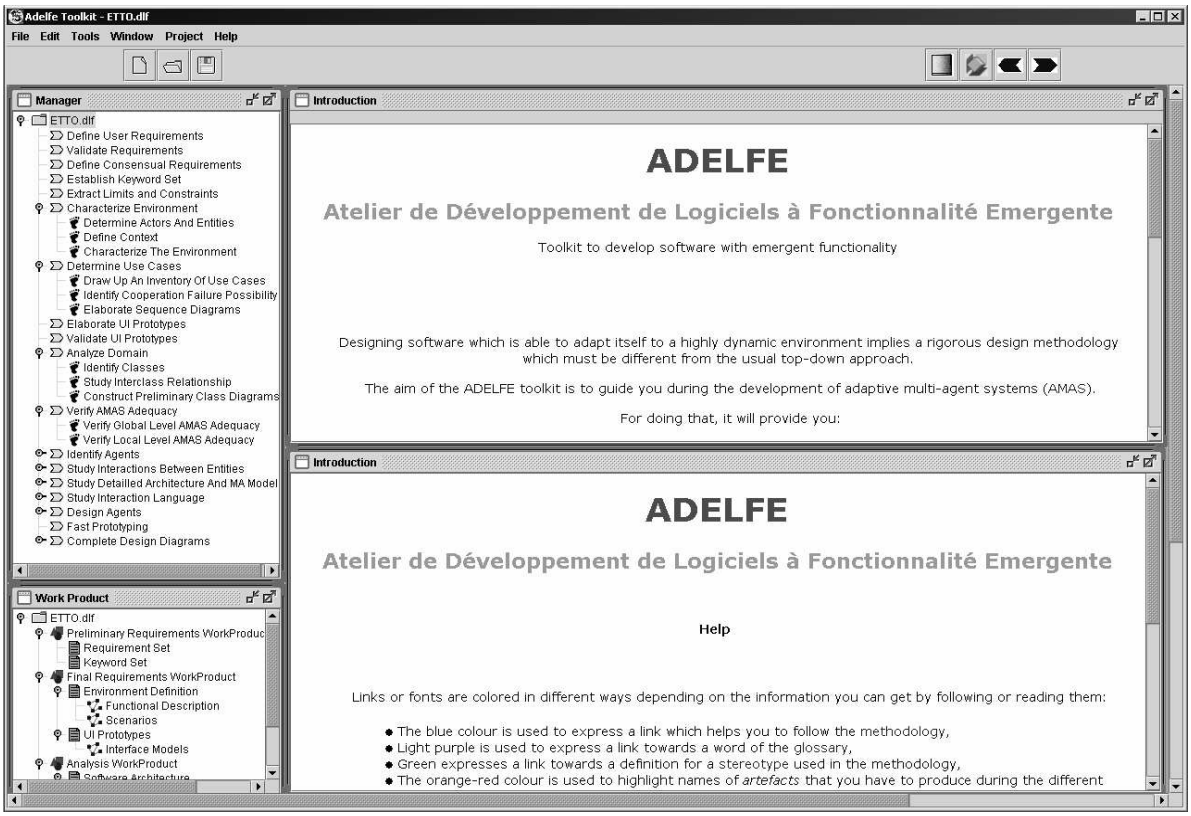


Figure 36 - L'interface de l'outil interactif

Dans les méthodes classiques orientées objet ou agent, ce type d'outil n'existe pas. Même si quelques outils liés à des méthodes orientées agent existent (e.g., AgentTool pour MaSE, PTK pour PASSI ou l'outil INGENIAS), ils ne sont pas vraiment des guides et des outils de vérification pour des concepteurs suivant le processus d'une méthode. Généralement, des guides comme des livres, des pages html sont donnés – e.g., le guide pour suivre le RUP – mais ne sont pas réellement des outils interactifs capable de suivre la progression au travers des différentes activités et étapes. L'outil interactif (figure 36) peut communiquer avec OpenTool dans le but d'accéder aux différents diagrammes alors que le processus progresse. C'est pour cela qu'il peut être considéré comme un guide réel qui supporte la notation adoptée par la méthode et vérifie la consistance du projet. Chaque activité ou étape du processus est décrite dans cet outil et illustrée par l'application de la méthode au problème ETTO. A travers les descriptions et exemples textuels, des termes

spécifiques à la théorie des AMAS sont reliés à un glossaire. De plus, l'outil est composé de plusieurs modules gérant ces différents aspects :

- Le module de gestion indique, pour les différents projets en cours, les différentes activités et étapes que le concepteur doit suivre lorsqu'il applique la méthode.
- Le module des produits affiche la liste des documents pour une activité ou étape donnée. Cette liste est automatiquement remise à jour lors de la production d'un document ou d'un modèle.
- Le module de description explique les différentes activités et étapes du processus que le concepteur doit suivre pour appliquer la méthode.
- Le module d'exemple, permet d'obtenir une aide supplémentaire sur l'application de la méthode en fonction de l'étape ou activité courante.
- Les modules de synthèse et de glossaire sont deux fenêtres optionnelles. La première permet d'afficher les document et modèles généré sous OpenTool par exemple pour une étape données ou pour le processus entier. Le contenu de cette fenêtre représente une aide pour la production de la documentation finale. Le module glossaire affiche et explique une liste de termes relatifs à la méthode ou à la théorie des AMAS. Les stéréotypes ajoutés à UML et OpenTool, ainsi que leurs règles de bon usage, sont aussi expliqués dans cette fenêtre.

3.4. Application à ETTO

ETTO (Emergent TimeTabling Organization) est une application développée par Sylvain Peyruqueou, stagiaire en DRT, Gauthier Picard doctorant, et les permanents Pierre Glize, Carole Bernon et moi-même. Le cahier des charges a été fourni au groupe de travail ASA de l'AFIA comme cas d'étude des différentes méthodes et/ou plates-formes des participants [Bernon, 2002b].Cette application a été utilisée pour valider la méthode ADELFE et est maintenant utilisée comme exemple didactique de l'utilisation d'ADELFE. Après avoir décrit le cahier des charges, la présentation ici est axée sur quelques points spécifiques tels que :

- l'étude des interactions,
- l'utilisation des stéréotypes dans le diagramme de classes.

3.4.1. Cahier des charges

Le problème choisi est un problème classique d'emploi du temps pour des cours. Les acteurs impliqués (enseignants, groupes d'étudiants et salles) ont un certain nombre de contraintes devant être respectées. Les besoins pour l'application ETTO ont été exprimés comme suit. Les participants sont les enseignants, les groupes d'étudiants et les salles de cours. Tout acteur possède des contraintes propres devant être satisfaites de la meilleure façon possible. Un enseignant possède des contraintes sur ses disponibilités (créneaux horaires possibles par exemple), ses capacités (les matières qu'il enseigne) et ses besoins particuliers en matériel pédagogique (rétro-projecteur, vidéo-projecteur, une salle de cours définie pour un cours particulier, ...). Un groupe d'étudiant doit suivre un enseignement particulier consistant en un certain nombre de créneaux horaires et pour un certain nombre

de matières (X créneaux pour la matière 1, Y créneaux pour la matière 2, ...). Une salle de cours est équipée ou non avec du matériel spécifique (rétro-projecteur, ...) et peut être libre ou non pour certains créneaux horaires.

Dans les sections suivantes seuls quelques aspects liés à la méthode ADELFE sont présentés. Une description complète de l'application est accessible dans le rapport [Bernon, 2002b].

3.4.2. L'analyse

Parmi les entités extérieures en interaction avec le système, certaines nécessitent des représentants à l'intérieur même du système. Dans la phase d'analyse, nous identifions, en tant qu'entités :

- les professeurs, les élèves et les salles de manière évidente,
- le gestionnaire des salles pour servir d'intermédiaire entre l'acteur "responsable des salles" et les salles du système,
- le gestionnaire des enseignements en tant qu'interface entre le "responsable des enseignements" et le système,
- le PPN[27] comme référence pour le "gestionnaire des enseignements",
- les contraintes semblent indispensables puisque beaucoup d'entités y sont soumises,
- un gestionnaire des contraintes pour chaque entité possédant des contraintes,
- une grille pour regrouper les salles et représenter le résultat.

L'outil d'adéquation se prononce clairement pour l'utilisation des AMAS au niveau global. En outre, il estime que certaines entités du système méritent, elles aussi, un développement AMAS. Par conséquent, il faudra revenir à cette étape plus tard en ne considérant plus que le niveau local.

Nous identifions ensuite deux catégories d'agents, les professeurs et les groupes d'élèves. Leurs caractéristiques sont tout à fait similaires et leur but individuel parfaitement symétrique : les professeurs cherchent les élèves et inversement. Nous considérons donc qu'il s'agit de deux classes distinctes d'un même type d'agent générique que nous appelons `RepresentativeAgent`.

Pour répondre de nouveau au questionnaire de l'adéquation au niveau local, nous considérons que le système est un `RepresentativeAgent` que nous avons déterminé à l'activité « Identifier les agents » (A12). L'environnement à prendre en compte devient alors celui d'un professeur ou d'un élève. Les professeurs et les groupes d'élèves devant respectivement assurer et recevoir plusieurs cours chacun (atteindre plusieurs objectifs) un agent sera créé par cours. C'est le `BookingAgent` (BA) qui se charge pour le `RepresentativeAgent` de trouver un partenaire et de réserver un lieu. Il y a autant de BA que le RA a de cours à assumer ou à recevoir.

[27] PPN : Plan Pédagogique National

L'étude des interactions entre agents

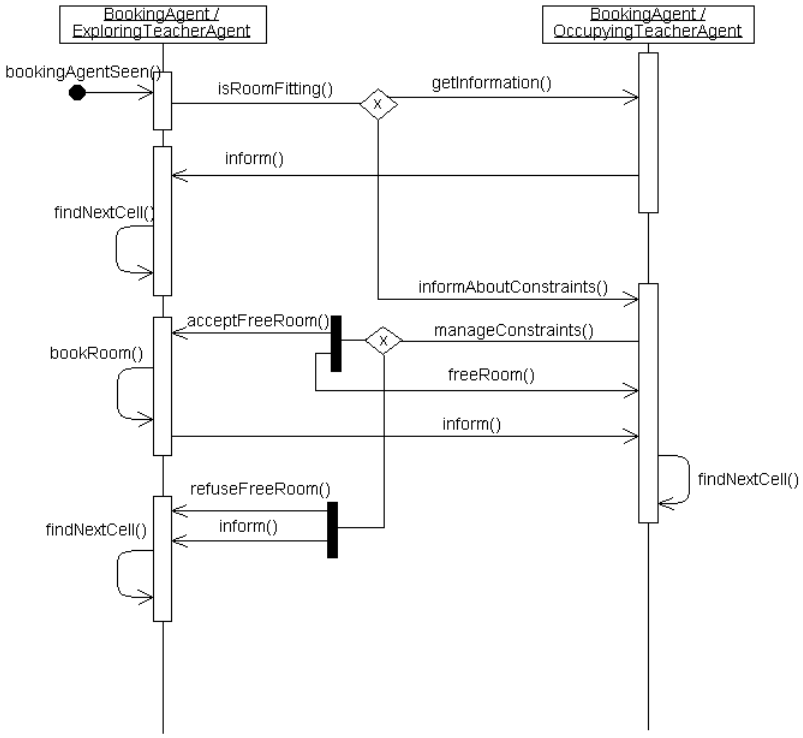


Figure 37 – Un diagramme de protocoles d'ETTO

Un exemple de diagramme de protocole entre deux `BookingAgent` représentant deux enseignants différents est donné figure 37. Le premier agent explore la grille de l'emploi du temps pour trouver un créneau et une salle satisfaisants. Le deuxième occupe déjà une salle et un créneau. Ce diagramme de protocole explique la négociation entre les deux agents lorsque le premier rencontre le second. Cette négociation peut soit aboutir sur un départ du premier agent soit sur une réservation par le premier et le départ de l'autre.

Dans la figure 37, le `BookingAgent` possédant le rôle `ExploringTeacherAgent` reçoit l'événement `bookingAgentSeen` comme condition de déclenchement pour commencer la négociation avec le `OccupyingTeacherAgent`.

De plus la notation AUML reste floue sur les branchements OR et XOR en ce qui concerne la prise de décision des envois de message. Notre contribution a été d'attacher au noeud (OR ou XOR) une méthode stéréotypée `<<aptitude>>` permettant de spécifier ce point. Par exemple, dans la figure 37 la méthode `isRoomFitting` est attachée au premier noeud XOR; i.e., en fonction des résultats de cette méthode, l'agent `ExploringTeacherAgent` peut soit demander plus d'information pour reprendre son exploration de la table d'emploi du temps, soit commencer à négocier en fonction des contraintes des deux agents.

3.4.3. La conception

Les diagrammes de classe

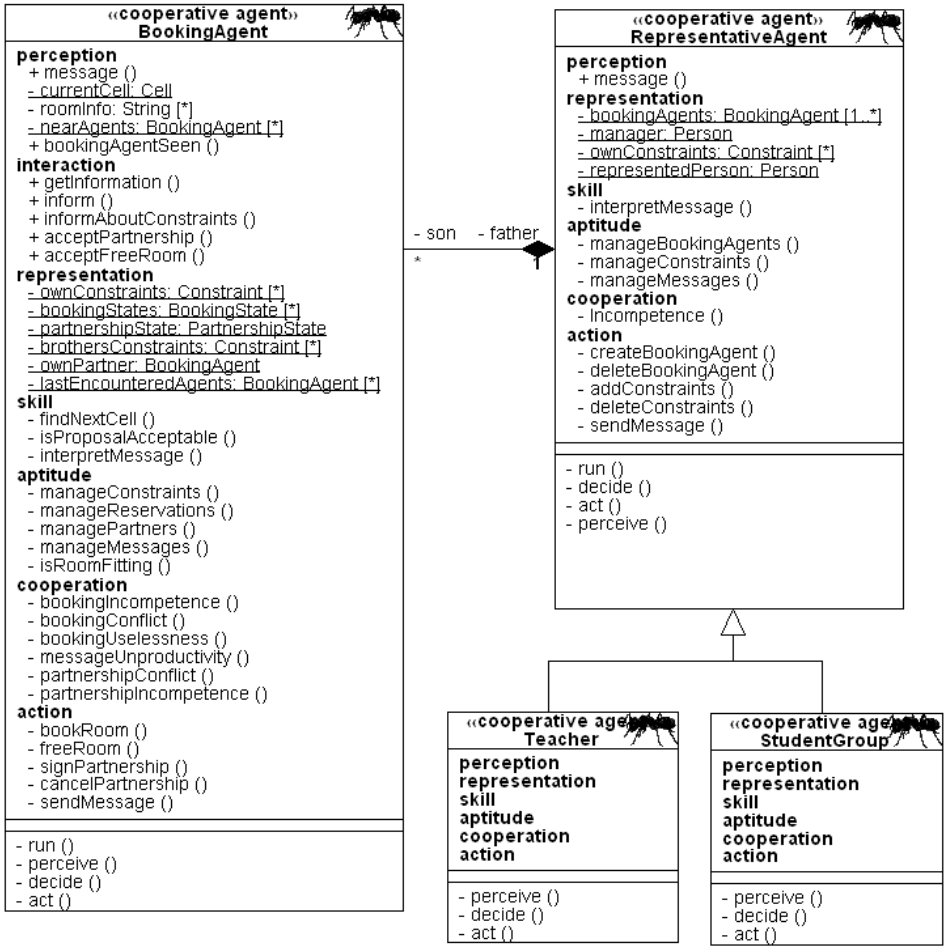


Figure 38 – Une partie du diagramme de classes d'ETTO

Deux classes principales d'agents, stéréotypées `<<cooperative agent>>`, sont définies pour ETTO: `RepresentativeAgent` et `BookingAgent`. La première peut représenter soit un groupe d'étudiants (`StudentGroup`) soit un enseignant (`Teacher`). Un `RepresentativeAgent` est composé de plusieurs `BookingAgent` dont la tâche est de trouver un créneau horaire dans l'emploi du temps pour un cours donné.

Une première modification apportée à OpenTool concerne la vue statique du système : les diagrammes de classe. Différentes boîtes spécifiques à ADELFE ont été définies pour prendre en compte les stéréotypes pré-définis sans perdre l'ergonomie des diagrammes de classe à cause du grand nombre de stéréotypes utilisables. Ainsi, les champs et les méthodes sont rangés dans des boîtes en fonction de leur stéréotype. Chaque classe stéréotypée `<<cooperative agent>>` doit hériter de la classe pré-définie `CooperativeAgent` ou bien d'une autre classe stéréotypée `cooperative agent`. En conséquence, toute classe d'agent possède au moins quatre méthodes – `run`, `perceive`, `decide` et `act` – pour simuler leur cycle de vie. La figure 38 montre un exemple de

diagramme de classe dans lequel apparaissent des classes d'agents. Comme les classes d'agents sont des classes d'objets, toutes les opérations ou associations sur des classes d'objets sont possibles sur des classes d'agents: composition, aggrégation, héritage... Ici, pour l'exemple d'ETTO, la classe `RepresentativeAgent` est composée de plusieurs `BookingAgent`. De plus, les classes `StudentGroup` et `Teacher` héritent de la classe `RepresentativeAgent` et héritent ainsi de ses champs et méthodes.

3.4.4. Résultats et analyse

L' atelier de conception de systèmes à fonctionnalité émergente est opérationnel et accessible en ligne : www.irit.fr/ADELFE comportant un processus de développement, une notation UML/AUML et des outils d'aide à la conception.

Cette méthode a été validée sur l'application ETTO qui sert d'exemple didactique. C'est une application simple et vite comprise par les utilisateurs potentiels de la méthode. Elle est très intéressante car elle aborde le besoin d'un système multi-agent adaptatif au niveau global et au niveau local.

ADELFE a été validé aussi par des étudiants dans le cadre de leur travail comme Davy Capera pour la conception de SYNAMEC [Capera, 2004a], Elsa Macchion pour le développement d'un système de reconnaissance de molécules [Macchion, 2004], Lionel Petit pour la réalisation d'un système coopératif autour de JRULES d'ILOG [Petit, 2004].

Le processus d'ADELFE a été écrit en SPEM (Software Process Engineering Meta-Model), c'est le deuxième processus après le RUP décrit dans ce formalisme. Dans le cadre du groupe TC methodology de la FIPA ce formalisme a été adopté et maintenant Gaia PASSI sont aussi exprimées en SPEM.

3.5. Contributions

La réalisation de la méthode ADELFE a permis de mettre en liaison trois étapes de modélisation. La première étape est l'identification des agents, réalisée en caractérisant les éléments du domaine après avoir décomposé l'application en entités. La deuxième correspond à la conception des agents selon le modèle d'agent coopératif (décrit dans la partie 2.2 du chapitre II). Cette étape correspond à une approche de conception ascendante. La dernière étape intègre la connaissance nécessaire à la mise en œuvre du contrôle. Elle est définie par la détection et le traitement des SNC, permettant à l'agent d'agir de manière autonome.

ADELFE a comme principales caractéristiques de permettre la conception de systèmes plongés dans des environnements dynamiques, de fournir une aide pour l'identification des agents et pour la définition de leur comportement. Elle fournit au concepteur les moyens de vérifier que la technologie des AMAS est bien adaptée à son application. La réutilisation de standards au sein d'ADELFE permet à un concepteur non spécialisé en systèmes multi-agents de l'utiliser. Son utilisation est facilitée par la documentation et les outils fournis.

La méthode ADELFE est une méthode orientée agent dédiée aux systèmes multi-agents adaptatifs, c'est une méthode spécifique. Elle suit les différentes étapes proposées par le RUP et interprétées par Neptune [Neptune, 2001], elle couvre le cycle de vie d'un logiciel classique. Elle est basée sur des standards tels que RUP, UML, AUML ce qui permet de promouvoir les techniques multi-agents dans le monde industriel. De nos jours les concepteurs sont déjà familiers de méthodes de conception basées sur ces standards.

4. Enseignements de cette période

Les travaux de cette période ont permis de suivre les objectifs, le premier consistait à expliciter, définir et opérationnaliser l'émergence pour la résolution de problèmes et le deuxième concernait l'ingénierie de tels systèmes de résolution.

Les applications développées dans la même période telles que la programmation émergente, la conception mécanique la construction d'un emploi du temps ont permis d'affiner et de clarifier le comportement coopératif d'un agent et de valider la méthode élaborée. Dans le cadre de la conception mécanique l'apprentissage se situe au niveau de la compétence de l'agent, en effet un changement de longueur pour un agent barre modifie ce qu'elle est capable de faire.

4.1. L'émergence

Le travail effectué sur la notion d'émergence a permis de clarifier ce qui peut être qualifié de phénomène émergent ou non dans un système artificiel. En vérifiant que le phénomène a bien les caractéristiques associées à un phénomène émergent mais aussi que le concepteur ne donne pas aux agents la manière d'atteindre la fonction que doit réaliser le système. Par exemple dans le projet SYNAMEC, une barre ainsi que les autres types d'agent n'ont pas dans leur code la position finale qu'ils doivent occuper, les types d'agents avec qui ils doivent être liés et les caractéristiques physiques qu'ils doivent avoir comme leur taille. Le même système ne donnerait pas lieu à un phénomène émergent si les composants étaient dirigés par un superviseur qui leur donnerait leur position, longueur… La méthode que nous préconisons pour opérationnaliser l'émergence est l'auto-organisation par coopération, c'est bien entendu une méthode et il existe sans nul doute d'autres méthodes non encore explorées. Son opérationnalisation a impliqué un travail sur l'environnement et sur la notion de coopération.

Dans les premiers systèmes développés selon la théorie des AMAS, l'environnement tout en contraignant le comportement du système n'en attendait pas un comportement très précis. La résolution de problèmes se passait entre les agents du système comme dans l'application du Tileworld, les tuiles et les trous sont passifs par rapport à la résolution. Le système et l'environnement sont toujours en interactions coopératives. Dans les derniers systèmes développés selon la théorie des AMAS, l'environnement joue le rôle de juge de l'adéquation fonctionnelle du système. Il a une participation plus active au cours de la

résolution car les interactions entre le système et l'environnement ne sont plus tout le temps coopératives. Le concepteur joue le rôle de l'environnement pour la programmation émergente et renvoie sous la forme de situation non coopérative le fait que le résultat du système ne le satisfait pas. La notion de feed-back est donc fondamentale pour maîtriser l'émergence comme cela a été vu pour la programmation émergente et pour la conception de mécanismes.

La réflexion au niveau local uniquement a permis de compléter la définition d'attitude coopérative. La détection et le traitement des situations non coopératives pour guider le comportement coopératif d'un agent avaient suffi pour obtenir un fonctionnement adéquat. Les systèmes pour lesquels la fonction est discontinue comme factorielle ou la conception mécanique ont fait apparaître que ce comportement n'était pas suffisant. La notion d'attitude coopérative a donc été complétée par le choix pour un agent des actions qui ne l'amènent pas dans une situation non coopérative. A l'action de « réparation » de la non coopération est ajouté la notion de « prévention ».

4.2. Un outil opérationnel

L'atelier de développement de logiciels à fonctionnalité émergente est opérationnel et en libre accès sur le site Web : www.irit.fr/ADELFE. Au cours du processus, un cadre formalisé d'un agent coopératif a été défini pour réaliser les stéréotypes, cela donne un guide aux concepteurs.

Les principales phases de conception couvrent l'analyse des besoins, l'analyse et la conception. Contrairement à de nombreuses méthodes, les agents ne sont pas supposés connus avant de mettre en œuvre la méthode, mais le concepteur doit les déterminer en cours de conception. Pour cela, ADELFE ne fournit pas d'outil automatique mais seulement un guide textuel qui permet de différencier une entité qui restera sous la forme d'objet et celle qui sera agentifiée.

Les trois outils associés au développement de logiciels facilitent le travail du concepteur. L'outil de base est OpenTool qui supporte le processus. L'intégration d'AUML lui apporte une spécificité et une originalité par rapport aux produits concurrents qui n'implantent pas AUML. Le deuxième outil correspond à un outil d'aide à la décision. Il permet au concepteur de choisir ou non une conception orientée système multi-agent adaptatif ou non. Cet outil est important car supportant une méthode très spécifique, elle ne doit pas être utilisée sur tout type de problème. Le concept de système multi-agent adaptatif doit être réservé pour des applications en ayant besoin. Le dernier outil est l'atelier en lui-même qui met du lien entre tous les outils et qui permet de guider de manière didactique le concepteur. Classiquement, les concepteurs peuvent trouver de l'aide dans les manuels, éventuellement un manuel en ligne comme dans le cas de Rational Rose, mais peu d'outils arrivent à ce niveau d'intégration.

4.3. De nouvelles problématiques de recherche

La théorie des AMAS a montré sa faisabilité pour des problèmes comportant des caractéristiques différentes. L'objectif suivant serait de montrer la convergence de ces systèmes. Démontrer la possibilité d'émergence semble quelque peu contradictoire mais une formalisation permettrait sans nul doute de montrer que le système grâce à l'auto-organisation par coopération peut atteindre l'état solution s'il existe.

Concernant la méthode fournie aux concepteurs, les phases de développement et de déploiement qui représentent les dernières phases du cycle de développement sont à concevoir. Pour cela deux pistes peuvent être explorées. La première consiste à utiliser les progrès réalisés en génie logiciel pour utiliser la technique MDA[28] [Marschall, 2003] pour, à partir d'un méta modèle multi-agent adaptatif, engendrer du code sur une plate-forme cible existante comme JADE [Pitt, 1999] par exemple. L'autre piste consiste à aller vers de la « conception vivante » ou « living design » [Georgé, 2003b]. Dans un premier temps, ADELFE fournirait au concepteur les moyens d'observer le système en cours de fonctionnement et lui donnerait les outils pour modifier le comportement des agents en cours de fonctionnement. Pour aller plus loin dans un deuxième temps, le système pourrait s'auto-construire comme dans la programmation émergente, le concepteur guiderait le codage en jouant le rôle de l'environnement.

La résolution de problèmes par émergence basée sur la théorie des AMAS a permis d'aborder des systèmes dont la fonction est discontinue et où l'environnement contraint le système pour l'orienter vers la solution. Le concepteur construit les agents coopératifs et l'environnement du système en ayant une connaissance du problème à résoudre.

5. Bibliographie relative au chapitre III

[Ali, 1997] Ali, S. M., Zimmer, R.M., *« The question concerning emergence »,* CASYS'97, Abstract Book, First International Conference on Computing Anticipatory Systems, CHAOS 1997.

[Arlabosse, 2004] Arlabosse F., Gleizes M-P., Occello M., *« Méthodes de conception »,* pages 137-171 Observatoire Français des Techniques Avancées : Systèmes Multi-Agents, Chapitre IV, Série ARAGO 29, 2004

[Atlan, 2000] Atlan H. , *« La finalité »,* Hors série Science&Avenir, 2000.Brazier F.M., Jonker C. M., and Treur J., Compositional design and reuse of a generic agent model. In Proceeding of Knowledge Acquisition Workshop - KAW'99, 1999

[Bergenti, 2004] Bergenti F., Gleizes M-P., Zambonelli F., *« Methodologies and Software Engineering for Agent. The Agent-Oriented Software Engineering handbook »,* Kluwer Publishing, 1-4020-8057-3, July 2004.

[Bernon, 2002a] Bernon C., Camps V., Gleizes M-P., Picard G., *« Projet ADELFE - délivrables D3.1 : Méthodologie et notations »,* Réseau National des Techniques Logicielles (RNTL), Avril 2002.

[28] MDA Model Driven Architecture

[Bernon, 2002b]Bernon, 2002b Bernon C., Camps V., Gleizes M-P., Glize P., Peyreuqueou S., Picard G., *« Ingénierie des Amas pour l'emploi du temps. ETTO – Emergent TimeTable Organization »,* Rapport Groupe de travail ASA AFIA 2002

[Beurier, 2003] Beurier, G., Simonin, O., Ferber, J., *« Un modèle de système multi-agent pour l'émergence multi-niveau »,* Actes des JFSMA03, pages 235-247, nov. 2003

[Boissier, 2004] Boissier O., Gitton S., Glize P., *« Caractéristiques des systèmes et des applications »,* Observatoire Français des Techniques Avancées : Systèmes Multi-Agents Série ARAGO 29

[Booch, 1992] Booch G., *« Conception orientée objets et applications »,* Addison-Wesley, 1992

[Bourjot, 2003] Bourjot, C., Chevrier, V., Thomas, V., *« A new swarm mechanism based on social spiders colonies: from web weaving to region detection »,* Web Intelligence and Agent Systems: An International Journal, 1, 1, pages 47-64.

[Brueckner, 2003] Brueckner S.A., Parunak H. V.D., *« Self-Organizing MANET Management »,* Di Marzo Serugendo G., Karageorgos A., Rana O.F., Zambonelli F. Editors of The First International Workshop on Engineering Self-Organising Applications (ESOA) Melbourne, Australia, July 2003 LNAI 2977 State of the Art Survey Springer Verlag 2004

[Bussman, 1998] Bussman S., *« Position Paper on Agent-Based Software Engineering »,* Contribution to Methodologies/software engineering SIG - First SIG Meeting - 1998 http://www.cs.vu.nl/~treur/SIG.meth0.3.html

[Caire, 2001] Caire G., Leal F., Chainho P., Evans R., Garijo F., Gomez J., Pavon G., Kearney P., Stark J. & Massonet P., *« Agent Oriented Analysis using MESSAGE/UML »,* Proceedings of the 2nd International Workshop on Agent-Oriented Software Engineering, AOSE'01. LNAI, vol. 2222. Springer Verlag, 2002, pp. 119-135

[Camazine, 2001] Camazine S., Deneubourg J.-L., Franks N., Sneyd J., Theraulaz G., Bonabeau E., *« Self-Organization in Biological Systems »,* Princeton University Press, 2001

[Capera, 2004a] Capera D., Gleizes M-P., Glize P., *« Mechanism Type Synthesis based on Self-Assembling Agents »,* Journal of Applied Intelligence Artificial 2004

[Capera, 2004b] Capera D., Gleizes M-P., Glize P., *«Self-Organizing Agents for Mechanical Synthesis»,* Engineering Self-Organising Systems Nature-Inspired Approaches to Software Engineering, LNAI 2977 State of the Art Survey Springer Verlag 2004, Di Marzo Serugendo G., Karageorgos A., Rana O.F., Zambonelli F. Editors, pages 169-185

[Cardon, 2000] Cardon, A., Guessoum, Z., *« Système multi-agents adaptatifs »,* Actes des JFIADSMA00, pages 100-116, oct. 2000

[Casteran, 2000a] Casteran J-C., *« Expérimentation et approche d'une méthodologie dans les systèmes complexes adaptatifs »,* DEA RCFR - UPS Toulouse, Juin 2000

[Casteran, 2000b] Casteran J-C., Gleizes M-P., Glize P., *« Des méthodologies orientées multi-agent »,* 8ièmes Journées Francophones d'Intelligence Artificielle Distribuée et des Systèmes Multi-Agents, Editions Hermès, Saint-Jean de la Vêtre France 2-4 octobre 200

[Castro, 2001] Castro J., Kolp M., and Mylopoulos J., *« A Requirements-driven Development Methodology »,* Proceedings of the 13th International Conference on Advanced Information Systems Engineering (CAiSE'01), Stafford, UK – June 2001.

[Cernuzzi, 2004] Cernuzzi L., Juan T., Sterling L., Zambonelli F., *« The Gaia Methodology: Basic Concepts and Extensions »* Methodologies and Software Engineering for Agent. The Agent-Oriented Software Engineering handbook., Bergenti F., Gleizes M-P., Zambonelli F. Editors, Kluwer Publishing, 1-4020-8057-3, July 2004.

[Chevrier, 2002] Chevrier, V., *« Contributions au domaine des systèmes multi-agents »,* Mémoire d'habilitation à diriger des recherches, soutenue le 3 janvier 2002 à l'Université Henri Poincaré, Nancy 1, 2002.

[Collinot, 1996] Collinot A., Ploix L., Drogoul A., *« Agent oriented design of a soccer robot team »,* Proceedings of the Second International Conference on Multi-Agent Systems - ICMAS 1996

[Cossentino, 2002] Cossentino M., *« Different Perspectives in Designing Multi-Agent System »,* AgeS'02 (Agent Technology and Software Engineering) Workshop at NodE'02, Erfurt, Germany, October 2002.

[Deloach, 2001] DeLoach S. A., *« Analysis and Design Using MaSE and agentTool »,* 12th Midwest A.I. and Cognitive Science Conference (MAICS01), Ohio, 2001.

[Demazeau, 2001] Demazeau, Y., *« VOYELLES »,* Habilitation à Diriger des recherches, INP Grenoble, 2001

[Di Marzo, 2003] Di Marzo Serugendo G., Karageorgos A., Rana O.F., Zambonelli F., *« The First International Workshop on Engineering Self-Organising Applications (ESOA »,* Melbourne, Australia, July 2003 LNAI 2977 State of the Art Survey Springer Verlag 2004

[Drogoul, 1993] Drogoul A., *« De la simulation multi-agents à la résolution collective de problèmes »,* Thèse de l'Université de Paris VI.

[Dotto, 1999] Dotto F., Travé-Massuyès L., Glize P., *« Acheminement du trafic d'un réseau téléphonique commuté par une approche multi-agent adaptative »,*Congrès CCIA, Girona - 1999

[Forrest, 1990] Forrest, S., *« Emergent computation: self-organizing, collective, and cooperative phenomena in natural and artificial computing networks »,* Proceedings of the Ninth annual CLNS conference, 1990.

[Foukia, 2003] Foukia N., Hassas S., *« Managing Computer Networks Security through Self-Organization : A complex System Perspectives »,* The First International Workshop on Engineering Self-Organising Applications (ESOA), Di Marzo Serugendo G., Karageorgos A., Rana O.F., Zambonelli F. Editors, Melbourne, Australia, July 2003 LNAI 2977 State of the Art Survey Springer Verlag 2004

[Frederiksson, 2003] Frederiksson M., Ricci A., Gustavsson R., Omicini A., *« The first International workshop on Theory and Practice of Open Computational Systems »,* TAPOCS 2003 at IEEE 12th International workshop on Enabling technologies: Infrastructure for collaborative enterprises WETICE 2003 Johannes Kepler Universitaet, Linz – Austria, 2003.06.09 – 11

[Frederiksson, 2004] Frederiksson M., Ricci A., Gustavsson R., Omicini A., *« The second International workshop on Theory and Practice of Open Computational Systems »,*

TAPOCS 2004 at IEEE 13th International workshop on Enabling technologies: Infrastructure for collaborative enterprises WETICE 2004 Modena Italy, 2004-06-14

[Garcia, 2003] Garcia A.F., Pereira de Lucena C.J., Zambonelli F., Omicin A., Castro J. (Eds.) (2003): *« Software Engineering for Large-Scale Multi-Agent Systems »,* Research Issues and Practical Applications]. Lecture Notes in Computer Science 2603 Springer 2003, ISBN 3-540-08772-9

[Georgé, 2003a] Georgé JP., *« L'émergence »,* Rapport interne IRIT n° 2003-12-R, 2003

[Georgé, 2003b] Georgé J.P., Picard G., Gleizes M-P., Glize P. (2003b), *« Living Design for Open Computational Systems »,* 1st International Workshop on Theory And Practice of Open Computational Systems TAPOCS at IEEE 12th International workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises WETICE 2003, Johannes Kepler Universitaet, Linz – Austria, 2003.06.09 – 11

[Georgé, 2004] Georgé J-P., *« Résolution de problèmes par émergence, Etude d'un Environnement de Programmation Emergente »,* Thèse de l'Université Paul Sabatier, Toulouse, Juillet 2004

[Giunchiglia, 2002] Giunchiglia F., Mylopoulos J., and Perini A., *« The Tropos Software Development Methodology: Processes, Models and Diagrams »,* AOSE'02, Bologna, July 2002

[Gleizes, 1999] Gleizes M-P., Marcenac P., *« Actes des septièmes Journées d'Intelligence Artificielle et des Systèmes Multi-Agents »,* Ingénierie des systèmes multi-agents », Editions Hermès

[Gleizes, 2000] Gleizes M-P., Glize P., *« ABROSE : Des systèmes multi-agents pour le courtage adaptatif »,* 8ièmes Journées Francophones d'Intelligence Artificielle Distribuée et des Systèmes Multi-Agents, Editions Hermès, Saint-Jean de la Vêtre France 2-4 octobre 2000

[Gleizes, 2003] Gleizes M-P., Millan T., Picard G., *« Adelfe: Using SPEM Notation to Unify Agent Engineering Processes and Methodology »,* Rapport interne IRIT n° IRIT/2003-10-R, Juin 2003

[Goldstein, 1999] Goldstein, J., *« Emergence as a Construct : History and issues »,* Emergence Volume 1, Issue 1, pages 49-71, 1999

[Grumbach, 1997] Grumbach, A., *« A propos d'émergence. Emergence ou explication »,* Intellectica Emergence and Explanation 1997/2 n°25, ISSN n°0984-0028 185-194, 1997.

[Gomez Sanz, 2002] Gomez Sanz J., and Fuentes R., *« Agent Oriented System Engineering with INGENIAS »,* in Fourth Iberoamerican Workshop on Multi-Agent Systems, Iberagents 2002.

[Guessoum, 2003] Guessoum Z., *« Modèles et Architectures d'Agents et de Systèmes Multi-Agents Adaptatifs »,* HdR de l'Université Pierre et Marie Curie, Décembre 2003

[Hassas, 2003] Hassas S., *« Systèmes Complexes à base de Multi-Agents Situés »,* HdR Université de Lyon 1 Décembre 2003.

[Holland, 1997] Holland J. H., *« Emergence – from Order to Chaos »,* Addison-Wesley, 1997.

[Hutzler, 1997] Hutzler G., Gortais B., Drogoul A., *« Le jardin des hazards, peinture abstraite et IAD réactive »,* Actes des cinquièmes journées francophones IAD&SMA, La Colle sur Editions Hermès, pages 295-306

[Jacobson, 1999] Jacobson Y., Booch G., Rumbaugh J., *« The Unified Software Development Process »,* Addison-Wesley, 1999

[Jennings, 1999] Jennings N.R., *« Agent-Oriented Software Engineering »,* Multi-Agent System Engineering, 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World, MAAMAW'99 Valencia Spain June/July 1999 Proceedings, Springer lecture Notes in Artificial Intelligence 1647 F.J. Garijo & M. Boman Eds.

[Iglesias, 1999] Iglesias C.M., Garijo M., Gonzalez J.C., *« A Survey of Agent-Oriented Methodologies »,* Intelligent Agents V, ATAL'98, LNAI 1555, Springer Verlag 1999.

[Kaplan, 2000] Kaplan F., *« Objets et Agents pour Systèmes d'Information et de Simulation* », Thèse du LIP6 Juin 2000

[Kinny, 1996] Kinny D., Georgeff M., and Rao A., *« A Methodology and Modelling Technique for Systems of BDI Agents »,* Agents Breaking Away: Proceedings of the Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World, Van Der Velde, W., Perram, J. (eds.), 1996. MAAMAW'96, (LNAI Vol. 1038). Springer-Verlag: Heidelberg, Germany

[Labbani, 1997] Labbani O., Müller J-P., Bourjault A., *« Analyse en vue d'une conception de comportemenst collectives émergents dans une colonie de robots »,* Actes des cinquièmes journées francophones IAD&SMA, La Colle sur Editions Hermès, pages 343-358

[Macchion, 2004] Macchion E., « Plate-forme système multi-agent adaptatif pour une exploration comportementale transcriptionnelle. Application à la levure Saccharomyces cerevisiae à partir de données métaboliques et de puces à ADN », Mémoire d'Ingénieur C.N.A.M. Informatique, Conservatoire National des Arts et Métiers de Toulouse, 2004.

[MARCIA, 1997] MARCIA, *« Auto-organisations : émergence de structures »,* Journée du PRC GDR IA : les systèmes multi-agents Toulouse Février 1996 et Journées du PRC GDR IA Grenoble Mars 1997 Editions Hermès

[Mamei, 2003] Mamei M., Zambonelli F., *« Self-Organization in Multi-Agent Systems : A Middleware Approach »,* First International Workshop on Engineering Self-Organising Applications (ESOA), Di Marzo Serugendo G., Karageorgos A., Rana O. F., Zambonelli F. Editors, Melbourne, Australia, July 2003 LNAI 2977 State of the Art Survey Springer Verlag 2004

[Marschall, 2003] Marschall F. and Braun P., *« Model Transformation for the MDA with BOTL »,* TR-CTIT-03-27. University of Twente, June 2003.

[M. R. Jean 1997] M. R. Jean : Nom collectif pour Batard, Brassac, Delépine, Gleizes, Glize, Labbani, Lenay, Marcenac, Magnin, Müller, Pesty, Quinqueton, Vidal, *« Emergence et SMA »,* Actes des cinquièmes journées francophones IAD&SMA, La Colle sur Editions Hermès, pages 323-341

[Müller, 1998] Müller, J., *« Vers une méthodologie de conception de systèmes multi-agents de résolution de problèmes par émergence »,* Actes des sixièmes Journées Francophones d'Intelligence Artificielle Distribuée et Systèmes multi-agents. Editions Hermès, 1998.

[Müller, 2001] Müller J-P., *« Systèmes multi-agents et résolution de problèmes par émergence »*, Séminiare LISI Lyon 2001

[Müller, 2002] Müller J-P., *« Des systèmes autonomes aux systèmes multi-agents : Interaction, émergence et systèmes complexes »*, Habilitation à diriger des recherches. Université de Montpellier II, Novembre 2002.

[Nagpal, 2003] Nagpal R., *« A Catalog of Biologically-Inspired Primitives for Engineering Self-Organization »*, The First International Workshop on Engineering Self-Organising Applications (ESOA), Di Marzo Serugendo G., Karageorgos A., Rana O. F., Zambonelli F. Editors, Melbourne, Australia, July 2003 LNAI 2977 State of the Art Survey Springer Verlag 2004

[Neptune, 2001] Neptune, *« Guidelines of a Process for the Use of (27/07/01) »*, Reference IST Project n° 1999-2001- http://neptune.irit.fr 2001

[Odell, 2000] Odell J., Parunak H.Bauer B., *« Extending UML for Agents »*, Proceedings of the Agent Oriented Information Systems (AOIS) Workshop at the 17th National Conference on Artificial Intelligence (AAAI), 2000.

[Odell, 2001]Odell J., Parunak H.V., and Bauer B., *« Representing Agent Interaction Protocols in UML »*, Oriented Software Engineering, P. Ciancarini and M. Wooldridge Editors, Springer-Verlag, Berlin, pages 121-140, 2001

[Padgham, 2002] Padgham L., and Winikoff M., *« Prometheus : A Pragmatic Methodology for Engineering Intelligent Agents »*, Workshop on Agent-Oriented Methodologies at OOPSLA 2002

[Parunak, 1997a] Parunak H. V.D., *« Go to the Ant: Engineering Principles from Natural Multi-Agent Systems »*, Annals of Operations Research 75 69-101 Special Issue on Artificial Intelligence and Management Science

[Parunak, 1997b] Parunak H. Van Dyke, Vanderbok R.S., *« Managing emergent behavior in distributed control systems »*, Proceedings of ISA-Tech'97

[Parunak, 2001] Parunak, H. Van D., Brueckner, S., *« Entropy and self-organization in multi-agent systems »*, Proceedings of Autonomous Agents'01, p. 124-130, 2001.

[Pesquet, 1999] Pesquet B., Gleizes M-P., Glize P., *« Une équipe de robots footballeurs auto-organisées : les SMACkers »*, Intelligence Artificielle Située, Cerveau, corps et environnement, Drogoul A. & Meyer J-A. Coordonnateurs Editions Hermès, pages 243-256

[Pesty, 2000] Pesty S., Sayettat-Fau C., Eds. « *8ièmes Journées Francophones d'Intelligence Artificielle Distribuée et des Systèmes Multi-Agents »*, Editions Hermès, Saint-Jean de la Vêtre France 2-4 octobre 2000

[Petit, 2004] *« Essai sur l'inférence adaptative dans JRules fondée sur la théorie des AMAS »*, Mémoire de Maîtrise IUP Ingénierie des Systèmes Informatiques, Université Paul Sabatier, 2004.

[Peyruqueou, 2002] Peyruqueou S., *« Approche et applications de l'ingénierie des systèmes multi-agents adaptatifs »*, Diplôme de Recherche Technologique, Spécialité : Ingénierie des Systèmes Informatiques et Automatiques, Université Paul Sabatier, 25 septembre 2002

[Picard, 2004] Picard G., Gleizes M-P., *« The ADELFE Methodology - Designing Adaptive Cooperative Multi-Agent Systems »,* Chapter 8, pages 157-176, In F. Bergenti, M-P. Gleizes, and F. Zambonelli Editors, Methodologies and Software Engineering for Agent Systems. The Agent-Oriented Software Engineering handbook. Kluwer Publishing, 1-4020-8057-3

[Régis, 2002] Régis C., Sontheimer T., Gleizes M-P., Glize P., *« STAFF : un système multi-agent adaptatif en prévision de crues »,* 10ièmes Journées Francophones d'Intelligence Artificielle Distribuée et des Systèmes Multi-Agents, Editions Hermès, 28-30 octobre 2002, Lille, France.

[Russel, 1995] Russel S., Norvig P., *« Artificial Intelligence: A Modern Approach »,* Prentice Hall Series, 1995.

[Searle, 1995] Searle J., *« La redécouverte de l'esprit »,* Gallimard 1995

[Servat, 1999] Servat, D., Leonard, J., Perrier, E., Treuil, J.P., *« The Rivage Project : a new approach for simulating runoff dynamics »,* Modelling of transport processes in soils, 24-26 Novembre 1999, Leuven, J.Feyen & K.Wiyo.Eds, Wageningen Pers, The Netherlands, pages 592-601

[Shehory, 2001] Shehory O.and Sturm A., *« Evaluation of modeling techniques for agent-based systems »,* Proceedings of the Fifth International Conference on Autonomous Agents, pages 624-631. ACM Press, May 2001.

[Sontheimer, 2002] Sontheimer T., *« Modèle adaptatif de prévision de crues par systèmes multi-agents auto-organisateurs »,* Diplôme de Recherche Technologique, Spécialité : Ingénierie des Systèmes Informatiques et Automatiques, Université Paul Sabatier, janvier 2002

[Steels, 1999] Steels, L., *« The Spontaneous Self-organization of an Adaptive Language »,* Koichi Furukawa and Donald Michie and Stephen Muggleton, editors, Machine Intelligence 15, pages 205—224, St. Catherine's College, Oxford: Oxford University Press.

[Topin, 1999] Topin X., Régis C., Gleizes M-P., Glize P., *« Comportements individuels adaptatifs dans un environnement dynamique pour l'exploitation collective de ressources »,* Intelligence Artificielle Située, Cerveau, corps et environnement, Drogoul A. & Meyer J-A. Coordonnateurs Editions Hermès, pages 223-242, 1999

[Treur, 1998] Treur J., *« Report of the First SIG Meeting, Contribution to Methodologies/software engineering SIG »,* First SIG Meeting, 1998 http://www.cs.vu.nl/~treur/SIG.meth0.3.html

[Tveit, 2001] Tveit A., *« A Survey of Agent-Oriented Software Engineering »,* NTNU Computer Science Graduate Student Conference. Norwegian University of Science and Technology, May 2001

[Van De Vijver, 1997] Van De Vijver, G., « *Emergence et explication »,* Intellectica : Emergence and explanation, 1997/2 n°25, ISSN n°0984-0028 185-194, 1997.

[Vasirani, 2003] Vasirani, M., Mamei, M., Zambonelli, F., *« Morphogenesis of cooperative robots with minimal capabilities: preliminary experiments »,* Proceedings of The First European Workshop on Multi-Agent Systems (EUMAS03), Oxford University, 18-19 december 2003

[Wood, 2000] Wood Mark, DeLoach Scott A., *« An overview of the Multiagent Systems Engineering Methodology »,* First International Workshop on Agent-Oriented Software Engineering (AOSE 2000), Limerick 2000

[Wooldridge, 2001] Wooldridge M. & Ciancarini P., *« Agent-Oriented Software Engineering: the State of the Art »*, AOSE 01, P. Ciancarini & M. Wooldridg, Editors, Springer Verlag LNAI 1957, 2001.

[Zambonelli, 2000] Zambonelli F., Jennings N.R., Wooldridge M., *« Organisational Abstractions for the Analysis and Design of Multi-Agent Systems »,* AOSE'00 LNCS, Springer-Verlag, 2000

Conclusion et Perspectives

Ce mémoire retrace mon parcours de recherche au cours duquel la problématique dominante a été la résolution de problèmes par un système artificiel constitué de plusieurs entités en interaction. Après avoir résumé les principales questions abordées concernant la résolution de problèmes, mon projet de recherche à court et à long terme est décrit.

Dans le cadre de la résolution distribuée de problèmes, c'est l'étude du contrôle de la résolution qui a été au centre de mes travaux. La question posée était de savoir comment contrôler la résolution de problèmes réalisée par plusieurs entités. Les réponses ont été apportées en terme de modèle d'agent et d'études des interactions.

Dans le cadre des systèmes multi-agents, la question centrale est de savoir comment résoudre des problèmes dans des environnements dynamiques où les agents n'en ont qu'une représentation partielle. Il faut donner au système des capacités d'adaptation pour lui permettre de faire face à des imprévus. Les systèmes à concevoir doivent être autonomes, robustes et permettre d'appréhender la complexité des applications. L'étude des systèmes que nous souhaitions concevoir et l'étude de la coopération comme mécanisme de résolution et d'adaptation, nous ont permis de proposer une théorie des systèmes multi-agents adaptatifs.

Dans le cadre de la résolution de problèmes par émergence, deux axes ont été poursuivis : d'un côté l'étude de la notion d'émergence et des mécanismes permettant de la maîtriser de manière autonome ; de l'autre des outils méthodologiques pour concevoir ces systèmes.

A partir de mes travaux antérieurs, l'objectif général de mon projet de recherche futur reste la conception de système plongé dans un environnement dynamique et composé d'agents en interaction, le tout devant avoir un comportement cohérent. Cet objectif est celui affiché de nos jours par les chercheurs du domaine des systèmes multi-agents, bien que jusqu'à présent leurs systèmes résolvaient des problèmes pour lesquels le concepteur connaissait la manière de les résoudre. Je souhaite continuer à étudier la notion de résolution de problèmes en m'intéressant particulièrement aux mécanismes qui permettent au système d'avoir un comportement émergent et ce sans que le concepteur puisse expliciter la manière de résoudre collectivement le problème. Du point de vue du concepteur, ce dernier ne saura pas forcément à quoi les agents qu'il construit seront

utilisés. Ce projet de recherche se bâtit sur des objectifs à long terme et se met en œuvre, sur des objectifs plus précis à court terme.

Bien entendu le côté expérimental est le socle de mon activité scientifique et c'est pourquoi je participe activement à l'élaboration de nouveaux projets. Certains ont pour objectif de continuer à valider la théorie des AMAS, d'autres sont plus en rapport avec l'ingénierie des systèmes multi-agents adaptatifs.

1. Les recherches à court terme

La théorie des AMAS est appliquée depuis un certain nombre d'années sur des applications très variées. Elle est à un stade où une formalisation est nécessaire. La méthode de développement est à continuer puisque les phases de développement et de déploiement ne sont pas encore opérationnelles. Pour cela nous nous proposons d'aller vers une automatisation du passage d'une conception à une plate-forme cible en utilisant les techniques MDA (Model Driven Architecture) du génie logiciel.

1.1. Formalisation

Nous devons désormais modéliser les systèmes sur lesquels nous travaillons c'est-à-dire des systèmes ayant un environnement et composé d'agents qui ont des compétences et des représentations d'eux-mêmes, d'autres agents et de l'environnement. Cette formalisation doit permettre d'exprimer les comportements locaux des agents et le comportement collectif global, ainsi que des propriétés telles que les conditions nécessaires et suffisantes pour la convergence vers un état coopératif par le système.

C'est ainsi, qu'en partenariat avec Jean Fanchon du LAAS, nous avons commencé un travail de formalisation de la théorie des AMAS combinant des produits d'automates et les topologies de réseau d'interconnexions.

L'objectif de ce travail est d'établir un modèle général d'un système multi-agent et de son environnement puis de le particulariser afin d'étudier des propriétés de ceux qui sont à fonctionnalité émergente fondée sur l'auto-organisation coopérative. Les deux questions de base qui se posent sont :

- Est-il possible de définir les propriétés que les agents doivent posséder pour que le système multi-agent satisfasse une propriété globale particulière ?
- Si les agents satisfont une certaine propriété, que peut-on en déduire au niveau global ?

Ces objectifs et cette collaboration ont donné lieu au dépôt du projet MASE (Modélisation par Automates de Systèmes multi-agents à Emergence) dans le cadre de FERIA en septembre 2004.

1.2. Vers des automatisations du codage au sein d'ADELFE

Le but est d'apporter une aide à la conception des systèmes multi-agents adaptatifs dans les phases de conception et développement et apporter des outils génériques pour supporter le passage d'un modèle indépendant de toute plate-forme, à un modèle spécifique à une plate-forme donnée. Pour cela, deux pistes peuvent être suivies.

La première piste consiste à concevoir des outils qui permettront au concepteur d'enrichir le code de son système en observant son fonctionnement. Dans ce cadre, nous souhaiterions que le concepteur code les agents en même temps qu'il observe le fonctionnement du système global. C'est ce que nous avons appelé le « living design » par analogie aux systèmes naturels qui se construisent au cours de leur vie. Ces travaux ont un lien fort avec les travaux de simulation. Dans une perspective plus éloignée, à partir des travaux sur la programmation émergente et de ce travail d'environnement de programmation pour faire du « living design » on peut considérer le concepteur comme l'environnement du système et obtenir un système qui s'auto-construit.

La deuxième piste concerne la conception d'un outillage MDA[29] générique comportant un langage de transformation de modèle à base de règles, un éditeur de règles et divers outils de transformation associés à ce langage ainsi que la définition d'une bibliothèque de transformations pour passer du modèle agent à une plate-forme de prototypage dédiée aux agents ou de développement telle que EJB ou CORBA. Les objectifs sont d'améliorer le travail des concepteurs en fournissant une plate-forme de prototypage dédiée aux agents et une bibliothèque de transformations pour passer du modèle agent à cette plate-forme. Ainsi à partir de la spécification du logiciel, la génération de code se ferait sur une plate-forme de prototypage. L'observation par le programmeur du prototype lui permettra ensuite de modifier le comportement des agents et donc la spécification. Ce cycle de conception se réitère jusqu'à ce que le prototype satisfasse le concepteur.

Pour atteindre ces objectifs, le projet MDAdelfe a été élaboré en partenariat avec IRIN de Nantes, l'équipe MACAO de l'IRIT et les industriels TNI Valiosys et Artal Technologies. Il doit être soumis à un prochain appel d'offres RNTL[30].

2. Les recherches à long terme

Dans une perspective à long terme, je souhaiterais orienter mes travaux de recherche toujours autour de la conception, de l'étude des concepts d'auto-organisation et d'émergence et d'expérimentations dans des domaines d'application nouveaux.

[29] MDA : Model Driven Architecture
[30] RNTL Réseau National des Technologies Logicielles

2.1. Autour de la méthode de conception : fragments, méta-modèle

Au sein des groupes de travail sur les méthodes de conception auxquels je participe, au niveau européen, nous avons défini des perspectives. En partant du constat que beaucoup de méthodes orientées agent existaient qu'elles avaient toutes une spécificité, il nous a semblé important de guider un concepteur non spécialiste du domaine pour utiliser ces différentes méthodes. L'idée principale est dans un premier temps de permettre à un concepteur à partir des méthodes existantes de se construire une méthode ad-hoc en fonction de son application et dans un deuxième temps de réaliser un outil qui en fonction de critères de l'application pourra construire la méthode la plus adaptée. Pour cela, il est nécessaire d'exprimer sous une forme unifiée toutes les méthodes. A cette fin, le groupe TC Methodology de la FIPA a défini la notion de fragments et a initié pour chaque méthode une définition du méta-modèle du système multi-agent associé avec la méthode.

2.2. Les ontologies : construction et maintenance

L'utilisation du Web par des humains ou par des systèmes artificiels repose beaucoup sur de la communication à partir de textes. Une ontologie est l'outil le plus utilisé pour obtenir cette intercompréhension. Parce que les informations sur le Web évoluent très rapidement, il nous semble important de donner des outils pour créer mais aussi pour maintenir les ontologies. Avec Nathalie Aussenac-Gilles (équipe CSC[31] de l'IRIT), nous voulons évaluer l'apport de Systèmes Multi-Agents adaptatifs à l'analyse de textes pour le repérage d'éléments d'ontologie.

Un premier travail a fait l'objet du DEA de Kévin Ottens soutenu en juin 2004. Il s'agissait de définir en quoi les modalités d'une analyse lexicale et syntaxique peuvent être revues si l'on utilise des agents réactifs comme supports à des propositions d'analyse. Ce travail a permis de dresser un état de l'art des rares travaux faisant appel aux systèmes multi-agents pour faire du traitement automatique des langues. Il a confirmé la nécessité d'avoir recours à un minimum de ressources sur la langue analysée.

Pour montrer la faisabilité de l'utilisation de systèmes multi-agents adaptatifs pour le recueil et la gestion dynamique d'ontologies, en collaboration avec les équipes des URFIST de Toulouse et de Nice et Thomas Lebarbé du LIDILEM[32] de Grenoble, le projet DynamO (Dynamic Ontologies), a été élaboré. Ses objectifs sont de construire et de maintenir des ontologies en contexte dynamique en s'appuyant sur des textes. Le domaine d'expérimentation choisi est celui des Sciences de l'Information et de la Communication (URFIST) avec une application à la formation à la recherche d'informations car nous avons déjà coopéré avec les URFIST sur le projet FORSIC. Ce projet se poursuit grâce au travail de thèse de Kévin Ottens qui débute en septembre 2004.

Dans la phase de conception, le logiciel assiste *l'ontologue* pour construire une ontologie à partir de textes. La création de l'ontologie est basée sur un système multi-agent adaptatif au sein duquel les unités linguistiques des documents sont *agentifiées*. Le réseau

[31] CSC : Conception des Systèmes Coopératifs
[32] LIDILEM : Laboratoire LInguistique et DIdactique des Langues Etrangères et Maternelles

conceptuel produit par les agents est fourni à *l'ontologue* qui propose d'éventuelles modifications. Ainsi, le système multi-agent apprend aussi à partir de l'observation de ces modifications. Pour la phase de maintenance, le logiciel travaille de manière autonome : il assure l'évolution de l'ontologie en fonction de son utilisation. Le système multi-agent qui a appris durant la phase de conception, continue à apprendre en cours d'utilisation.

L'apport scientifique repose essentiellement sur l'articulation originale entre le domaine multi-agent, le traitement de la langue et les ontologies. Les propriétés de coopération, d'auto-organisation et d'émergence des systèmes multi-agents adaptatifs seront exploitées de manière innovante pour atteindre deux objectifs :

- ajouter aux ontologies une dimension dynamique qui correspond à la capacité d'évolution des concepts en fonction des situations ;
- sortir de la séquentialité des traitements linguistiques en intégrant l'interaction entre syntaxe et structures conceptuelles.

2.3. L'informatique diffuse et la computation autonome

Dans l'équipe SMAC, nous avons comme objectif de recherche de développer encore plus l'aspect informatique diffuse et la computation autonome car cela représente un vrai challenge dans les années à venir au vu du développement croissant d'appareils électroniques communicants (PDA, téléphone portable….). Des modèles de systèmes totalement nouveaux doivent être développés.

Dans le contexte de l'informatique diffuse, les systèmes doivent permettre de faire travailler ensemble des entités a priori conçues au départ pour rendre un service précis à un utilisateur mais qui en les faisant travailler ensemble peuvent rendre d'autres services non prévus au départ. Par exemple, de minuscules capteurs peuvent être répartis dans votre maison et vous avertir par téléphone d'une éventuelle inondation. Lors de vos courses, en passant à proximité de magasins proposant des offres promotionnelles susceptibles de vous intéresser, des messages peuvent être envoyés sur votre téléphone portable, si vous le désirez.

Dans le cadre de la computation autonome, les systèmes artificiels devront avoir des capacités d'auto-réparation et d'adaptation à des changements environnementaux qui modifient les ressources dont ils ont besoin tels que la bande passante, la capacité de calcul…

L'objectif est donc de faire interagir et travailler ensemble de nombreux logiciels qu'ils soient de granularité fine ou forte en ne connaissant pas a priori le comportement collectif de tous ces logiciels. Notre proposition est de définir un modèle de programmation et une infrastructure pour développer ce type de logiciels. Le déploiement des applications dans le cadre de cette infrastructure devrait se faire de manière non supervisée et adaptative. Un mécanisme d'auto-organisation devrait permettre à cet ensemble de logiciels d'avoir un comportement cohérent.

3. Les orientations

Dans tous les projets cités, mes travaux futurs sont une continuité et une extension de mes recherches actuelles, ils se déclinent en objectifs plus précis :

- la formalisation de la théorie des AMAS et de ses propriétés,
- l'amélioration de la modélisation de l'environnement,
- la création d'outils d'aide à la conception de systèmes complexes : étendre ADELFE aux phases codage test maintenance, utilisation de plates-formes, d'infrastructures, création de nouveaux outils.

De manière plus générale, on peut observer que les applications informatiques et les systèmes informatiques actuels évoluent en complexité et en passage à l'échelle. Que ce soit à l'échelle planétaire avec le Web ou à un niveau macroscopique avec les nanotechnologies, de nombreuses applications nécessitent l'interaction dynamique d'un grand nombre de composants et interdisent un contrôle centralisé. Au macro niveau le système est vu comme le résultat des interactions entre les composants du micro niveau. Ces futurs logiciels devront montrer des caractéristiques plus proches des systèmes naturels que des systèmes informatiques traditionnels. Pour cela, il est nécessaire de trouver de nouvelles théories, techniques, de nouveaux modèles et de nouvelles méthodes de conception de ces systèmes. Cela représente une problématique intéressante et innovante à laquelle je souhaite activement contribuer. En effet, les systèmes multi-agents et les systèmes basés sur l'émergence représentent une de ces nouvelles technologies prometteuses pour modéliser et concevoir ces applications complexes.