République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
U N I V E R S I T E M O H A M E D K H I D E R D E B I S K R A

NUMÉRO D'ORDRE :

T H È S E

Pour obtenir le titre de

Docteur en Sciences
Spécialité : INFORMATIQUE

Présentée par :

Abdelouahab BELAZOUI

Modélisation d'un comportement coopératif d'un agent

Thèse dirigée par :
Pr. Okba KAZAR

Soutenue publiquement le 03 octobre 2016 devant le jury composé de :

Président :	Pr. Mohamed Tayeb LASKRI	Professeur	Université d'Annaba
Rapporteur :	Pr. Okba KAZAR	Professeur	Université de Biskra
Examinateurs :	Pr. Omar BOUSSAID	Professeur	Université Lyon 2 – France
	Pr. Nadjia BENBLIDIA	Professeur	Université de Blida
	Dr. Khaled REZEG	M.C.A	Université de Biskra
	Dr. Saber BENHARZALLAH	M.C.A	Université de Biskra
Invité :	Dr. Samir BOUREKKACHE	M.C.B	Université de Biskra

Année universitaire : **2015 – 2016**

AU NOM DU DIEU LE MÉSÉRICORDIEU LE CLÉMENT

Remerciements

C’est difficile de remercier ici toutes les personnes grâce à qui j’ai pu mener à bien toutes ces années de recherche ; une période de rencontre et d’échange qui ne peut se résumer à une simple liste de noms.

Néanmoins, je tiens tout d'abord à remercier vivement mon encadreur **Pr. Okba KAZAR**, Professeur à l'Université de Biskra et directeur du Laboratoire d’INFormatique Intelligente (LINFI), pour m'avoir dirigé tout au long de cette thèse. J'ai beaucoup bénéficié de ses conseils, de ses suggestions pertinentes pour la mise au point de ce travail.

Je remercie chaleureusement les membres du jury de thèse :

Pr. Mohamed Tayeb LASKRI, Professeur à l’Université d’Annaba, pour m’avoir fait l’honneur de présider mon jury de thèse, ainsi que **Pr. Omar BOUSSAID**, Professeur à l’Université Lyon 2 – France, **Pr. Nadjia BENBLIDIA**, Professeur à l’Université de Blida, **Dr. Khaled REZEG**, Maître de conférences (MCA) à l’Université de Biskra, **Dr. Saber BENHARZALLAH**, Maître de conférences (MCA) à l’Université de Biskra et **Dr. Samir BOUREKKACHE**, Maître de conférences à l’Université de Biskra, pour avoir accepté de juger ce travail.

Enfin, je tiens à remercier tous les membres de mon entourage qui ont, de près ou de loin, contribué à créer un contexte favorable au bon déroulement de ce travail.

*** * * * * * ***

Résumé

Ces dernières années, le développement des applications qui supportent les activités coopératives a attiré l'attention des chercheurs. Ainsi, plusieurs approches et méthodes ont été proposées pour atteindre cet objectif. Dans ce contexte, nous proposons une nouvelle solution pour la problématique des travaux collectifs, qui est le résultat de la combinaison de plusieurs approches de coopération existantes.

Au centre de notre approche un Framework de coopération tolérant aux fautes appelé *CoMAS* (Coopérative Multi-Agent System) qui permet en utilisant la préemption des ressources aux agents coopérants, en cas de défaillance, de maintenir le service et assurer la survie du système. Nous étudions le cas des systèmes auteurs coopératifs en e-Learning pour examiner le comportement du notre Framework face à une activité coopérative. La solution que nous proposons trouve application, surtout avec les activités coopératives ; C'est une solution à la fois générique et extensible pour tout le processus de coopération.

Mots-clés : Coopération ; Système Multi-Agent ; Résolution de Conflit ; Compétence ; Temps de coopération ; Système Auteur ; e-Learning.

*** * * * * * ***

Abstract

Nowadays, development of application that supports cooperative works has attracted recent attention. Thus, several approaches and methods were proposed to reach this aim. In this context, we propose a new solution to the problem of collective work that presents a result of hybridization of several cooperative approaches, as follows: the non-communicative approach to minimizing the required time to perform a cooperative activity and conflict resolution approach based competency for improving the quality of cooperative works.

Therefore, we propose a cooperation framework fault tolerant called *CoMAS* (Cooperative Multi-Agent System) brings preemption resources to cooperating agents in case of failure to prevent the blocking and ensure the survival of the system. Moreover, we study the case of cooperative authoring systems for e-learning to examine the behavior of our framework with a cooperative activity. Finally, our cooperation approach provides a generic and extensible solution that covers the whole cycle of a cooperation process.

Keywords: Cooperation; Multi-Agent System; Resolution of Conflict; Competence; Cooperation Time; Authoring System; e-Learning.

*** * * * * * ***

ملخــص

إن الأنظمة الداعمة للأعمال التعاضدية أثارت اهتمام الباحثين في الآونة الأخيرة مما أدى إلى ظهور مجموعة من المناهج والطرق لغرض تحسين نوعية الخدمة المقدمة من طرف هذه الأنظمة؛ في هذا الإطار، قمنا بتطوير مقاربة جديدة لحل مشكل العمل الجماعي ترتكز على تهجين مجموعة من المناهج، وهي كالتالي: المنهج الغير تحاوري لهدف تقليص الوقت اللازم لإتمام عمل جماعي ومنهج حل النزاعات المعتمد على الكفاءات لتحسين نوعية النتائج المحصل عليها.

نتيجة لهذا التصور، قمنا بتطوير إطار للعمل الجماعي متسامح مع العطب يقوم بسحب الموارد من الأعوان المشاركين في حالة تلفهم من أجل ضمان استمرارية النظام؛ بالإضافة إلى قيامنا بدراسة حالة الأنظمة التعاضدية للتأليف في التعليم عن بعد لأجل تقييم نوعية الخدمة المقدمة من طرف منهجنا في الأعمال التعاضدية. في الختام نقول إن مقاربتنا تمثل حلا عاما غير متعلقة بميدان خاص قابلة للتمديد وتغطي جميع أطوار العملية الجماعية.

الكلمات المفاتيح: نظام متعدد أعوان تعاضدي؛ حل النزاعات؛ الكفاءة؛ مدة التعاضد؛ نظام التأليف؛ التعليم عن بعد.

* * * * * * *

Table des matières

*** * * * * * ***

Liste des figures

Liste des tableaux

* * * * * * *

Liste des abréviations

ACC: Agent Communication Channel

ACL: Agent Communication Language

AEF: Automates à états finis

AEIO: Agent – Environnement – Interaction – Organisation

ADT: Abstract Data Types

AMS: Agent Management System

API: Application Programming Interface

AUML: Agent-based Unified Modeling Language

BD: Base de Données

CBL: Case Based Learning

CGI: Common Gateway Interface

CoMAS: Cooperative Multi-Agent System

CoMAS-AS: Cooperative Multi-Agent System for Authoring System

Contract-Net: Contractual Network

DF: Directory Facilitator

Doi: Digital Object Identifier System

QoS: Quality of Service

CDL: Conflict Driven Learning

CSCW: Computer−Supported Cooperative Work

CTL: Logique Temporelle Arborescente

ESTELLE: Extended State Transition Language

FIPA: Foundation for Intelligent Physical Agents

HTML: HyperText Markup Language

HTTP: HyperText Transfert Protocol

IAD: Intelligence Artificielle Distribuées

IDL: Interface Definition Language

IEEE: Institute of Electrical and Electronics Engineers Inc.

IIS: Internet Information Services

ISO: International Standard Organisation

ITU: International Telecommunication Union

JADE: Java Agent Development Environment

JDK: Java Development Kit

JDT: Java Development Tooling

JVM: Java Virtual Machine

KQML: Knowledge Query and Manipulation Language

LTL: Logique Temporelle Linéaire

NTIC: Nouvelles Technologies de l'Information et de la Communication

OMG: Object Management Group

OSI: Open Systems Interconnection

PADA: Perception – Analyse – Décision – Action

RdP: Réseau de Petri

RMI: Remote Method Invocation

SDL: Specification and Description Language

SMA: Système Multi-agents

TCAO: Travail Coopératif Assisté par Ordinateur

TIC: Technologies de l'Information et de la Communication

UML: Unified Modelling Language

URL: Uniform Resource Locator

VON BDI: Value – Obligation – Norm – Belief – Desire – Intention

W3C: World Wide Web Consortium

WYSIWYG: What You See Is What You Get

XML: eXtensible Markup Language

* * * * * * *

Introduction générale

A. Préliminaire

Dans ces dernières années, la coopération se définit comme étant l'un des comportements sociaux les plus actifs qui intervient dans un système multi-agent (*SMA*), la coopération se définit comme une nécessité quand un agent ne peut pas atteindre ses objectifs tout seul et sans l'aide des autres agents.

Minimiser le temps nécessaire pour la résolution d'un problème, tout en améliore la qualité des résultats, font que l'utilisation de la coopération est devenue indispensable, ajoutent à ça la nature multidisciplinaire des problèmes actuels qui rendront l'intervention de plusieurs experts essentiel et même incontournable.

Actuellement, le développement des applications basées sur les activités coopératives est reconnu pour être la solution là mieux adaptée par rapport à la nature des problèmes et aux objectifs visées par les solutions proposées. La coopération est devenue la solution miracle et elle à trouver place au sommet des intérêts de la recherche sur les *SMA*.

Dans ce contexte, plusieurs approches et méthodes ont été proposées, et de nombreux concepts subordonnés ont été identifiés, tels que : la communication, l'interaction, la coordination et la négociation. Cependant et malgré la diversité de ces méthodes on constate l'absence d'un cadre référentiel qui définit la relation de la coopération avec ces concepts subordonnés. Pour cela, nous proposons, une nouvelle solution pour l'optimisation de la coopération dans le contexte des travaux collectifs. La solution proposée est le résultat de la combinaison de plusieurs approches, on parle d'une approche hybride rassemblent les différentes techniques préexistantes et qui tire avantage de chacune d'elles.

Au cœur de notre travail un Framework de coopération tolérant aux pannes appelé *CoMAS* (Coopérative Multi-Agent System). *CoMAS* apporte une nette amélioration de l'allocation des ressources en utilisent la préemption comme moyen permettent la réaffectation des ressources aux agents suite à un réajustement de décessions provoquer par une défaillance, dans un objectif global de maintenir le système en un état de marche. Dans le cadre de notre travail, et pour valoriser cette proposition, nous étudions le cas des systèmes auteurs coopératifs en e-Learning. Les résultats montrent que *CoMAS* est une solution générique et extensible qui couvre tout le cycle d'un processus de coopération.

B. Eléments de la problématique

Notre principal objectif dans ce travail est la proposition d'une approche de coopération multi-objectifs qui vise à la fois : la minimisation du temps de convergence de la solution coopérative, tout en améliore la qualité des résultats obtenus.

C. Fondements de notre approche de coopération

Pour répondre aux objectifs visés par notre solution, nous avons essayé de tirer profits des solutions prés-existantes, chacune de ces solutions présente de son coté un avantage, et l'idée est de les rassemblés tous dans une même proposition, on parle de la plus vielle solution qu'est la combinaison. Les approches de base utilisées sont :

- *Approche non-communicative* : cette méthode repose sur l'interdiction de toutes formes de communication inter-agents coopérants, cette propriété est utilisée par *CoMAS* pour répondre à l'objectif de la minimisation du temps ;
- *Approche par compétence* : est une technique pédagogique utilise les compétences nécessaires dans un domaine dans la conception et le développement d'une activité pédagogique, elle définit et découpée en termes d'acquisition de capacités nécessaires pour effectuer une tâche [1]. En effet, nous adoptons cette approche pédagogique dans la spécification des compétences des agents coopérants pour plus de spécifications ce qui permet d'améliorer la décessions d'allocation des tâches.

- *Approche de résolution de conflit* : dans un système multi-agent, chaque agent est capable d'assurer certaines tâches en fonction de ces compétences, beaucoup d'agents ont les mêmes compétences et peuvent réalises les mêmes tâches. L'approche de résolution de conflit permet de résoudre tout éventuel conflit entre les agents par la définition d'une métrique de compétence requise pour assurer chaque tâche, *CoMAS* utilise cette propriété d'une manière sélective pour l'affectation des travaux coopératifs.

La définition et la répertoriassions des compétences de chaque agent, permet une meilleure affectation des tâches, et donc une exécution plus rapide du moment où la tâche est affectée à l'agent le plus compétant. L'intersection de plusieurs agents sur la même compétence peut provoquer un conflit d'affectation, et c'est à la métrique de compétence de résoudre ce problème avec le meilleur arrangement possible. Pas de communication entre les agents conduit directement à un gain de temps considérable, dans ce contexte et en se basent sur ces trois affirmation, notre solution gère l'interaction entre ces deux principaux acteurs : l'agent coopérant et le Framework de coopération, par la définition et l'intégration d'un *vecteur d'état* dans la spécification conceptuelle de chaque agent coopérant pour permettre dans un premier temps de déclarer explicitement ces compétences, et aider le Framework à allouer ses sous-problèmes aux agents coopérants les mieux adaptés dans un deuxième temps. La projection de cette technique sur le cas des systèmes auteurs coopératifs en e-Learning présente un cas d'étude très intéressent du moment où l'essentiel du travail de ces systèmes est coopératif.

D. Organisation de thèse

Ce manuscrit de thèse est composé de trois parties principales, la première partie cerne le cadre général de domaine d'étude. Nous présenterons dans le recueil de cette partie deux chapitres introductifs permettent la mise en contexte dans le domaine de *TCAO* en générale, puis, nous présentons le phénomène l'activité coopérative dans un *SMA*. En effet, nous présentons en premier chapitre la notion du collecticiel des travaux coopératifs en termes d'avantages et de classifications. En plus, nous présentions les différents formalismes de modélisation de la coopération et des systèmes coopératifs. En deuxième chapitre, nous introduisons explicitement notre domaine d'étude de la coopération au sein d'un *SMA*.

La deuxième partie présente l'état de l'art de notre domaine d'étude. Cette partie comporte une étude panoramique de quelques travaux connexes de la coopération dans un *SMA*. Ultérieurement, nous captons les inconvénients des approches de coopération actuelles qui faisant les motivations fondatrices de notre approche de coopération basée sur la compétence.

La troisième partie présente notre contribution dans la modélisation de la coopération dans un *SMA*. En effet, cette partie englobe trois chapitres, à savoirs :

En quatrième chapitre, nous montrons l'architecture conceptuelle de notre approche de coopération. Dans ce contexte, nous présentons le concept de vecteur d'état de l'agent coopérant et notre Framework de coopération nommé *CoMAS*. En plus, nous exhibons les diagrammes de séquences qui présentent toutes les interactions possibles entre l'agent coopérant et le Framework dans une activité coopérative ;

En cinquième chapitre, nous étudions le cas des systèmes auteurs coopératifs en e-Learning pour examiner le comportement du notre Framework face à une activité coopérative. En effet, nous proposons un système auteur nommé *AS-CoMAS* basé sur notre approche de coopération ; puis, nous expérimentons le comportement du système à l'aide d'un exemple. Nous évaluions les performances de notre approche en termes de minimisation du temps consacré à la mise en œuvre de la coopération en comparaison avec deux autres approches : appel d'offre et négociation.

En sixième chapitre, nous décrivons les outils de développement et leurs critères de choix en vue aider les développeurs dans la tâche de mise en œuvre du système.

Nous terminons cette thèse par une conclusion générale et les perspectives souhaitable et possible pour notre approche de coopération.

*** * * * * * ***

Première partie :
Contexte de travail

"Toute connaissance est une réponse à une question"

Gaston Bachelard

Sommaire

* * * * * * *

Chapitre 1 L'activité coopérative : principes et concepts de bases

1.1 Introduction

Depuis son apparition, l'informatique a transformé beaucoup de professions, c'est tout le monde du travail qui est presque touché par cette discipline et donc tout est profondément modifié et réorganisé. Le domaine de travail coopératif, lui, n'est pas resté insensible au phénomène informatique. Beaucoup de recherche s'est en effet effectuées dans le but de simuler sur ordinateur cette tâche aussi humaine qu'est le travail coopératif.

L'étude des travaux multidisciplinaire gravitant autour de l'activité de coopération nous permet de proposer, dans ce premier chapitre, une synthèse des notions essentielles permettant d'appréhender l'état des connaissances actuelles sur cette activité.

Dans la perspective d'introduire explicitement le domaine du travail coopératif, nous nous proposons maintenant de définir le concept *TCAO* et la notion de collecticiel (ou *Groupware*), puis nous mettons d'avantage l'accent sur la taxonomie collecticiel. Enfin, nous présentons les différentes approches de modélisation de la coopération et systèmes coopératifs.

1.2 Le TCAO : Travail Coopératif Assisté par Ordinateur

Le Travail Coopératif Assisté par Ordinateur (*TCAO* est la traduction en français du *CSCW*) est un récent domaine d'activités qui suscite l'intérêt de la recherche et de l'industrie [2]. L'objet du *TCAO* est d'adapter la technologie de l'information aux besoins des utilisateurs impliqués dans des activités de groupe. Les auteurs dans [3] proposent la définition du *TCAO* suivante :

> *"Le TCAO est le domaine de recherche répondant aux questions suivantes : quelles sont les caractéristiques spécifiques du travail coopératif comme opposé au travail effectué par des individus isolés ? Comment l'informatique peut-elle être appliquée pour soutenir les problèmes logistiques du travail coopératif ? Comment les conceptions abordent-elles les délicats et complexes problèmes des systèmes qui façonnent les relations sociales ?"*

Les solutions du *TCAO* sont appliquées dans de nombreux domaines à savoir : la production, l'enseignement, le commerce, les jeux, etc. Bien que ces domaines ne visent pas les mêmes objectifs, ils possèdent les mêmes caractéristiques qui doivent être prises en considération par les logiciels du travail coopératif [4]. La réalisation par les utilisateurs de tâches coopératives à partir de leurs stations de travail constitue une autre caractéristique des logiciels destinés au support du *TCAO* [5].

1.3 Notion de Collecticiel

Vu l'avancement considérable que connait actuellement le domaine des réseaux de télécommunication, ainsi que l'intégration immense en termes de matériels informatiques dans les secteurs professionnels, éducatifs et domestiques, le domaine du *TCAO* s'affirme de plus en plus dans notre vie quotidienne.

Le collecticiel ou *Groupware* désigne tout à la fois les processus de travail en équipe et les outils logiciels qui supportent ces processus [6]. En d'autres termes, le collecticiel peut être compris comme un ensemble de méthodes et de techniques de travail en équipe, ces méthodes et techniques étant instrumentées par des outils logiciels conçus pour améliorer les mécanismes de communication, de coopération et de coordination.

Dans la suite de cette section, nous reviendrons d'abord sur les principales définitions qui ont été proposées. Nous considérons ensuite les avantages visés à travers l'utilisation de ces systèmes. Enfin, nous discutons les principales approches de développements fournies dans la littérature afin de fournir plus de clarifications autour de la technologie collecticiel de façon générale.

1.3.1 Définitions

Dans le but de caractériser un collecticiel, une variété de définitions ont été proposées. Linguistiquement, le dictionnaire *Reverso*[1], définit le terme collecticiel comme étant un logiciel permettant à un groupe de réaliser un travail. Scientifiquement, nous citons la plus habituelle des définitions, celle donnée par [7] :

> *"Computer-based systems that support groups of people engaged in a common task (or goal) and that provide an interface to a shared environment".*

Une interprétation française de cette définition a été introduite par [8] :

> *"Les Groupware sont des systèmes informatiques qui assistent un groupe de personnes engagées dans une tâche commune (ou but commun) et qui fournissent une interface à un environnement partagé".*

Sur les plans organisationnels, le collecticiel désigne de nouvelles façons de travailler privilégiant la coopération entre les individus et une meilleure coordination des actions menées. Elles reposent généralement sur un ensemble d'outils génériques : de messagerie, de bibliothèque (stockage et partage de documents, édition conjointe, etc.), de réunion, de calendrier (agendas partagés), de gestion de tâches, etc.

1.3.2 Bref historique

L'idée d'utiliser les ordinateurs pour aider les humains à collaborer remonte au moins à la fin des années 50, avec l'invention de la notion de document hypertexte et hypermédia, et la description de son utilisation pour créer une véritable intelligence collective dans les organisations. Depuis, l'adoption d'outils tels que le courrier électronique et les forums de discussion ont modifié nos manières de travailler [6].

Plusieurs types de collecticiels existent pour les situations nécessitant un plus fort couplage entre participants. C'est le cas des négociations commerciales où l'influence directe entre personnes est essentielle. De même le contrôle aérien demande un haut niveau de sécurité et des opérateurs bien coordonnés [9]. Dans ce cas les recherches sur le *TCAO*

[1] Reverso, disponible en ligne sur l'adresse: http://dictionnaire.reverso.net/francais-definitions/collecticiel, date de la dernière consultation: 26/08/2012.

proposent des outils basés sur des protocoles de communication informatiques permettant de synchroniser des applications pouvant être situées à des milliers de kilomètres, et de transmettre de la voix ou de la vidéo sur les mêmes réseaux. Par exemple les éditeurs partagés permettent à plusieurs personnes de travailler simultanément sur un même document, texte ou dessin, De même les agendas partagés permettent à un groupe de personnes de mettre en commun leurs emplois du temps.

1.3.3 Avantages des Collecticiels

L'utilisation d'un collecticiel dans un travail de groupe présente un gain multidimensionnel. Dans cette section, nous présentons quelques avantages des collecticiels de travail de groupe.

1.3.3.1 Élimination des contraintes spatio-temporelles

Au sein d'un groupe, les participants peuvent collaborer autour d'un projet partagé sans être réellement dans le même endroit et au même instant. Ils peuvent rédiger des documents partagés, développer des idées, dialoguer à propos de problèmes, etc. De plus, ils peuvent participer au travail à partir de n'importe quel endroit. Cela signifie que nulle contrainte n'est posée sur le lieu ni sur le temps.

1.3.3.2 Parallélisation de communications et prise de décision rapide

Usuellement, une réunion regroupant dix participants intervenants de manière équitable, avec un tour de parole de six minutes chacun, signifie que chaque personne doit rester à l'écoute des autres pendant 54 minutes (le temps de non-participation de chacun vaut à 90 %) ; ce qui baisse le rendement des réunions. A travers l'exploitation des technologies adéquates, le partage des opinions se fait de manière plus rapide parce que les communications peuvent se dérouler de façon parallèle. Par conséquent, les décisions peuvent être prises dans des temps réduits.

1.3.3.3 Amélioration de la participation

Les systèmes collecticiels poussent les participants à intervenir aux activités de manière plus décontractée. Par conséquent, la possibilité de se protéger derrière l'anonymat fournie

par les outils collecticiels, permet de dissocier l'identité et les idées des intervenants. De plus, les réunions traditionnelles dégagent chez les participants le sentiment d'être pressés volontairement ou involontairement par le groupe en vue de s'accordé aux idées dominantes. L'anonymat conduit à plus d'autonomie et de franchise dans les commentaires et favorise donc des réunions plus productives.

1.3.3.4 Structuration des activités de groupe

La création d'un guide structuré du travail partagé constitue l'une des possibilités d'un collecticiel. Par conséquent, il sera difficile aux intervenants de ne pas le suivre. De ce fait, chaque participant contribue de façon totale à l'achèvement d'activités structurées. La rétroaction de groupe, supportée par les outils collecticiels assure une transparence sur l'état intégral des tâches en cours et avise implicitement chaque intervenant sur le progrès des activités partagées.

1.3.3.5 Mémoire organisationnelle

Avec les systèmes collecticiels, toutes les informations sont sauvegardées de manière automatique et chaque participant peut interroger les informations saisies ou modifiées par les autres. Ceci crée une flexibilité telle que les collaborateurs peuvent attendre l'instant opportun pour participer au débat et ne risquent pas de perdre le fil même s'ils sont interrompus par une pause durant la réunion. Dans cette logique, la contrainte d'établir après chaque réunion un procès-verbal n'a plus lieu d'être.

Ainsi, puisque les collaborateurs peuvent prendre le temps de réfléchir, de s'imprégner des commentaires d'autrui, la synergie de groupe en utilisant les collecticiels devient plus forte que dans les réunions traditionnelles. Vu la possibilité de consulter à tout moment les documents partagés, la mémoire organisationnelle s'avérera précieuse surtout en cas d'oubli d'élément établis pendant la réunion.

1.4 Taxonomies Collecticiels

Un collecticiel se compose d'une variété d'outils informatiques agissant en groupe. Ces outils peuvent être mis en place de manière séparée comme l'email, les forums, les

agendas, etc., ou de façon globale comme ceux qui sont propres au travail collectif tels que : les éditeurs partagés, les jeux en réseau, les outils d'aide à la décision collaborative, etc. Plusieurs expériences ont eu lieu pour l'élaboration d'une taxinomie des systèmes collecticiels. Toutes ces expériences ont proposé des modèles intéressants, dont il est difficile d'en favoriser un plutôt qu'un autre. Certains aspects qui permettraient une classification des systèmes collecticiels sont résumés dans ce qui suit :

- Classification selon les modes de travail de groupe (temps/espace);
- Classification fonctionnelle (trèfle fonctionnel);
- Classification conceptuelle (trois univers conceptuels).

1.4.1 Classification espace-temps

La présence simultanée ou différée des participants à une activité de coopération permet de qualifier la coopération, respectivement de synchrone ou d'asynchrone [6]. Les auteurs dans [7] proposent une classification basée sur les notions de temps et d'espace. Les logiciels de coopération peuvent appartenir à l'une des quatre catégories de la taxonomie spatio-temporelle [10] (voir figure 1), à savoir :

a) *Activité de type même lieux / même moment* : représente l'interaction de type face-à-face ;
b) *Activité de type même lieux / différents moments* : représente l'interaction asynchrone ;
c) *Activité de type différentes lieux / même moment* : représente l'interaction synchrone distribuée ;
d) *Activité de type différentes lieux / différents moments* : représente l'interaction asynchrone distribuée.

La figure ci-après montre une matrice des modes de travail de groupe [11] :

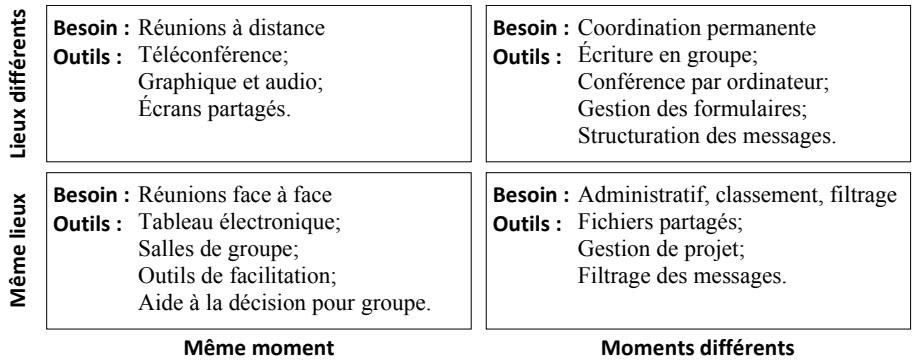


Figure 1. Matrice des modes de travail de groupe

1.4.2 Classification Fonctionnelle

Les systèmes collecticiels peuvent être classifiés, selon [12] en fonction de trois modèles :

a) *Le modèle ontologique*, qui donne une description claire des objets manipulés par le système, ainsi que les actions qui les manipulent ;

b) *Le modèle de coordination*, qui tient compte de la spécification des activités des intervenants ainsi que les relations entre elles ;

c) *Le modèle d'interface utilisateurs*, qui s'occupe de la définition des interfaces de communication.

La classification d'Ellis et Wainer, citée précédemment, a fait l'objet d'une critique soulevée par [13], qui adapte ce modèle en proposant une nouvelle variante (voir figure 2).

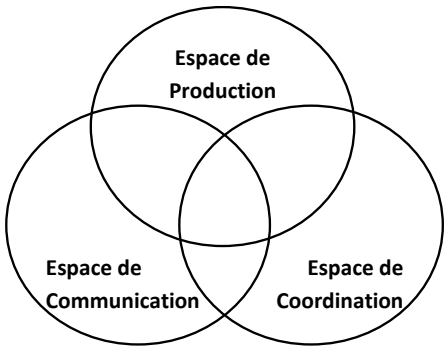


Figure 2. Trèfle fonctionnel des systèmes collecticiels

1) *L'espace de communication*, qui recouvre toutes les fonctionnalités relatives à l'aspect communication du système, ainsi qu'aux échanges d'informations ;

2) *L'espace de production*, qui tient compte de la spécification des fonctionnalités manipulant les données partagées ;
3) *L'espace de coordination*, qui s'occupe de la manière d'utilisation des ressources communes par les différents participants, ainsi que de l'aspect coordination des activités partagées à savoir : l'affectation des rôles, la spécification des droits d'accès, la répartition des tâches, etc.

1.4.3 Classification conceptuelle

Les auteurs dans [14] a suggérons une décomposition du système collecticiel en trois grands univers (voir figure 3) : l'univers de données, des utilisateurs (qui accèdent et manipulent les éléments de l'univers de données) et l'univers de l'organisation (qui structure les deux autres univers). Enfin, il faut préciser que les univers conceptuels définis permettent de spécifier une architecture logique pour la conception d'un système coopératif complet.

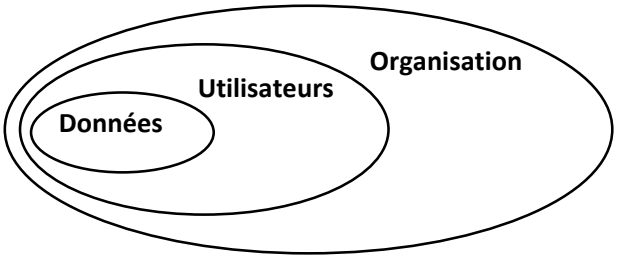


Figure 3. Univers conceptuels de la coopération

1.5 Modélisation de la coopération et systèmes coopératifs

La modélisation des systèmes coopératifs est nécessaire à leur bonne compréhension, leur analyse et leur vérification. Nous présentons dans cette section la formalisation par workflow, les graphes de coopération et un modèle orienté processus individuel de coopération. Enfin, nous évoquons le lien étroit existant entre coopération et systèmes multi-agents.

1.5.1 Le formalisme workflow

La formalisation par workflow vise l'étude des systèmes coopératifs [15, 16]. Pour cela, une analyse de ces systèmes est réalisée en plusieurs points :

1) *Analyse des différentes tâches* : il existe deux types de tâches : élémentaires et composites. Les tâches élémentaires sont dites indivisibles et les tâches composites sont dites divisibles. Elles peuvent être divisées en sous-tâches qui peuvent être composites ou élémentaires. Des conditions peuvent être utilisées afin de modifier l'exécution d'un processus en fonction d'une tâche. Il est ainsi possible d'énumérer toutes les tâches élémentaires et de représenter leur enchaînement.

2) *Analyse des acteurs du système* : l'objectif de cette analyse est d'énumérer les acteurs du système ainsi que leurs fonctions. Une fois les tâches représentées, il convient de spécifier les acteurs associés à ces tâches.

3) *Analyse des données échangées* : l'objectif de cette analyse est de définir les données échangées entre les différentes tâches. Il est aussi utile de spécifier la forme que prennent ces données.

4) *Analyse de l'infrastructure* : il s'agit dans cette partie de lister les choix technologiques effectués. Ceux-ci sont dépendants de la localisation des sites, des systèmes déjà en place et des capacités du réseau existant par exemple.

L'analyse d'un système coopératif par workflow permet un bon niveau de compréhension de celui-ci. Cependant, cette formalisation ne permet pas la définition du système global, ni une vérification ou simulation. Notre but est de comprendre, mais aussi de prévoir le fonctionnement des systèmes coopératifs. En ce sens, la formalisation par workflow ne répond pas au besoin de modélisation identifié pour les systèmes coopératifs.

1.5.2 Le modèle de trèfle fonctionnel

Le modèle de trèfle fonctionnel est un modèle fonctionnel pour les applications distribuées coopératives. Il distingue trois niveaux fonctionnels : Coopération, Coordination et Communication. Une application distribuée basé sur le modèle de trèfle fonctionnel est

donc composée d'entités distribuées qui coopèrent suivant des règles de coopération, en utilisant un médium de coordination et un médium de communication [17]. Ce modèle, très conceptuel est utilisé comme base à un ensemble de modèles formels dont nous allons décrire les plus importants.

1.5.3 Les graphes de coopération

Les travaux des auteurs dans [18, 19, 20, 21, 22] proposent un modèle basé sur le partage des données. La définition la plus fiable retenue pour caractériser une coopération entre deux agents[2] est celle de la mise à disposition d'information privée. Un agent coopère avec un autre s'il lui transmet une partie de son information privée. De ce fait, le modèle proposé exprime la coopération tout en se situant au-dessus de la couche de communication. Un agent coopère avec un autre s'il lui communique une partie de sa connaissance. Ce modèle utilise la logique modale afin de définir les relations et les propriétés entre les agents communicants. Ainsi, il a été défini un ensemble A d'activités, $A = A_i$. Chaque activité, A_i maintient un ensemble de valeurs d'information ou de prédicats pouvant être exportés $(P = P_i)$. A_i Coopère avec A_j si A_i permet à A_j d'accéder à certaines de ses valeurs, alors exportées, qui deviennent connues de A_j.

Le modèle basé sur les graphes de coopération est assez adaptatif pour caractériser de nombreuses situations de coopération. Il permet de modéliser la coopération entre plusieurs personnes dans des organisations peu dynamiques. En effet, les prédicats ainsi que leurs dépendances doivent être préalablement définis, et leur évolution dynamique rend l'utilisation du modèle complexe. Les aspects comportementaux des membres de la coopération, ainsi que l'aspect global du système coopératif ne sont cependant pas pris en compte.

[2] Dans ces travaux, le terme agent définit un utilisateur humain ou un composant logiciel qui reproduit une partie d'un comportement humain. Il ne s'agit donc pas d'agent constituant un *SMA*.

1.5.4 Les processus individuels de coopération

L'auteur dans [23] utilise un modèle de processus modélisé par des réseaux de Petri (*RdP*). Celui-ci recouvre le processus individuel classique d'une coopération : déclenchement, renseignement, conception, décision, évaluation et application. Les réseaux de Petri colorés temporisés sont utilisés pour modéliser les interactions coopératives entre membre. Selon l'avancement individuel de chacun, certaines transitions sont franchies. Ainsi, un membre en état de renseignement passe en état de conception tout en activant la transition d'un autre membre le faisant ainsi passer de l'état déclenchement à l'état renseignement. Les couleurs du RdP servent à représenter l'information véhiculée.

Le modèle basé sur les processus individuels de coopération permet de détecter des blocages éventuels au sein d'un processus coopératif. Il permet aussi de modéliser une partie importante de la sémantique utilisée lors des échanges menant aux différentes étapes du processus individuel et collectif. Ce modèle est beaucoup plus intéressant que le formalisme workflow dont on retient également le concept de processus coopératif. Ce modèle ne tient pas compte du système utilisé pour coopérer et donc de son influence sur le résultat de la coopération. Celle-ci est représentée par les états individuels correspondant aux différentes étapes de la résolution d'un problème.

1.5.5 Les systèmes multi-agents

Les systèmes multi-agents ont été utilisés avec succès pour modéliser l'évolution coopérative de systèmes naturels. La résultante globale des comportements individuels des fourmis par exemple apparaît de façon claire lors de simulations réalisées. La coopération se réalise de façon implicite, dès lors que des agents intelligents représentant des membres ayant une aptitude à la coopération, sont intégrés à une simulation multi-agent [24]. Les recherches réalisées sur les multi-agents leur donnent de plus en plus de possibilités de modélisation. Ainsi, un grand nombre de mécanismes humains peuvent être intégrés à des agents. Les travaux portant sur des aspects sociologiques tels que les actes de langage sont largement utilisés pour donner des capacités cognitives aux agents. La modélisation de système de coopération complexe impliquant des personnes au comportement et aux connaissances variées devient donc progressivement une réalité. En vue de mieux

comprendre la relation entre la coopération et les systèmes multi-agents, nous avons proposons une étude détaillée dans le prochain chapitre afin de clarifier la nature de relation entre ces deux concepts

1.6 Conclusion

L'activité coopérative présente le contexte général de notre travail de thèse. Pour cela, nous avons présentons une synthèse des notions essentielles permettant d'appréhender l'état des connaissances actuelles sur cette activité. Nous commençons par clarification des concepts de *TCAO* et collecticiel, puis nous avons recensons quelques gains engendrés par l'utilisation d'un système collecticiel dans un travail coopératif, ensuite, nous proposons trois façons de classification des systèmes collecticiels (spatio-temporelle, fonctionnelle et conceptuelle). Enfin, nous présentons quelques approches de modélisation de la coopération et des systèmes coopératifs. Nous proposons dans le suivant chapitre, une étude détaillée de l'activité coopérative dans un environnement multi-agents.

*** * * * * * ***

Chapitre 2 La coopération et les systèmes multi-agents

2.1 Introduction

La coopération est un des concepts majeurs des *SMA*, c'est également le thème visé par cette thèse. Il est donc important d'en avoir une compréhension précise. Malgré l'importance de ce concept, il n'existe pas véritablement de théorie de la coopération. De ce fait, le développement d'agents coopératifs doit être précédé d'une réflexion sur la coopération. De plus, la définition de la coopération est généralement subordonnée aux concepts de coordination, d'organisation, de communication et de négociation et l'insuffisante définition des relations entre ces concepts augmente la confusion sur la signification de la coopération.

Ce chapitre se compose de trois sections. La première a trait à la terminologie : il présente différentes définitions de la coopération, de la coordination et de la collaboration. La deuxième section comporte une présentation des *SMA*. Enfin, la troisième section étudie l'activité coopérative dans un *SMA* en termes d'indices, de formes et de méthodes de coopération.

2.2 Concepts subordonnés à la coopération

Le terme coopération revêt de nombreuses significations selon le contexte dans lequel il est utilisé. Il est fréquemment interconnecté à d'autres termes proches tels que la collaboration, la coordination, ou même la compétition. Afin de mieux comprendre les concepts liés à cette activité, nous proposons dans cette section une synthèse de définitions données dans la littérature.

Au sens strict, coopérer signifie opérer ensemble (préfixe : Co-). Le dictionnaire Larousse donne les définitions suivantes concernant la coopération :

- *Coopérer* : agir conjointement avec quelqu'un.
- *Coopération* : méthode d'action économique dans laquelle des personnes ayant des intérêts communs constituent une entreprise où les droits de chacun à la gestion sont égaux et où le profil est réparti entre les seuls associés au prorata de leur activité.

L'encyclopédie *wikipedia*[3] donne la définition suivante :

> *"Dans un système basé sur la coopération, les différents acteurs travaillent dans un esprit d'intérêt général de tous les acteurs. Cela suppose un certain degré de confiance et de compréhension. La coopération est antagoniste à la concurrence".*

Malgré le contexte très général de ces définitions, les concepts d'*intérêts communs*, de *droits* (au sens légal du terme). De *profil* et d'*activité* sont évoqués. Les notions de *confiance* et de *compréhension* sont également évoquées.

Actuellement, les travaux effectués sur le workflow définissent la coopération ainsi : La tâche de coopération considère le processus de division du travail (en autres tâches) et des responsabilités (rôles) parmi les coauteurs. Elle considère aussi la définition des ressources et des périodes pour chaque tâche [16]. La notion de planification qui ressort de cette définition est primordiale pour le déroulement de l'activité de coopération [24].

D'autres termes sont fréquemment utilisés lorsque la coopération est évoquée. Les mots collaboration et coordination apparaissent en effet régulièrement. Parfois, ils peuvent être mal interprétés. Nous proposons ici d'éclaircir cet aspect. La différence entre coopération et collaboration peut être justifiée ainsi [14] :

> *"Un groupe en collaboration a une tâche à réaliser et il partage la réalisation de cette tâche. Il s'agit d'un ensemble d'individus qui possèdent chacun une certaine vue d'un problème. Ces individus cherchent à converger vers un*

[3] Wikipedia, disponible en ligne sur l'adresse: http://fr.wikipedia.org/wiki/, date de la dernière consultation: 03/01/2013.

> *même but global, chacun participant à la tâche globale et effectuant une partie de la résolution du but. La collaboration implique le découpage d'une tâche en sous tâches qui seront exécutées par différentes entités".*

La coopération, quand à elle, concerne un ensemble d'entités ayant des buts individuels convergents vers un objectif commun. La collaboration est couramment utilisée dans un processus de coopération, mais elle n'est qu'une partie de celle-ci.

Ainsi, il convient de voir la collaboration comme une communauté d'activité et d'objets, et la coopération comme une communauté de sens et d'objectifs. Cette vision est confirmée dans [25] :

> *"Dans le cas où plusieurs tâches sont attribuées à un groupe dans lequel les membres n'effectuent pas nécessairement les mêmes tâches, il convient de parler de coopération. Alors que le terme collaboration doit être utilisé si l'ensemble des membres partage les parties d'une tâche afin d'aboutir à un but commun".*

Le terme de coordination mérite lui aussi d'être mieux cerné. La coordination est définie comme l'agencement de tâches dans un ordre permettant l'aboutissement de celles-ci et de la tâche globale sous-jacente. La coordination contient une notion de centralisation des décisions. Le meilleur exemple pour comprendre ce concept est la marche d'un homme. Celle-ci nécessite une capacité de coordination de la partie cerveau. Il est pourtant impossible de dire que les différents membres du corps coopèrent pour produire l'action de marcher !

La figure 4 classe ces différentes actions selon le degré de communication de groupe. Ceci confirme que la coopération se place sur un niveau plus complexe d'interaction que la collaboration [26].

Figure 4. Intensité du flux d'information au sein du groupe

Ces premières définitions nous rappellent que la coopération est un mécanisme complexe [27], difficile à cerner avec le vocabulaire existant, mais ayant une place bien définie au sein d'un ensemble d'autres termes évoquant des concepts proches.

2.3 Agent et système multi-agent

Les systèmes multi-agents constituent un des axes de l'intelligence artificielle distribuée, ses systèmes permettent de faire coopérer un ensemble d'agents dotés d'un comportement intelligent et de coordonner leurs buts et leurs plans d'actions pour résoudre un problème [28]. Dans ce qui suite, nous proposons le contexte théorique des agents et des systèmes multi-agents.

2.3.1 Le concept d'agent

Les agents sont des entités actives plus ou moins autonomes qui composent un *SMA*. Un agent est défini par [29] comme une entité physique ou virtuelle :

- Qui est capable d'agir dans un environnement ;
- Qui peut communiquer directement avec d'autres agents ;
- Qui est mue par un ensemble de tendances (sous la forme d'objectifs individuels ou d'une fonction de satisfaction, ou même de survie, qu'elle cherche à optimiser) ;
- Qui possède des ressources propres ;
- Qui est capable de percevoir son environnement (de manière limitée) ;
- Qui ne dispose que d'une représentation partielle de cet environnement (éventuellement aucune) ;
- Qui possède des compétences et offre des services ;
- Qui peut éventuellement se reproduire, mourir et changer d'état ;
- Dont le comportement tend à satisfaire ses objectifs, en tenant compte des ressources et des compétences dont elle dispose, et en fonction de sa perception, de ses représentations et des communications qu'elle reçoit.

Wooldridge et Jennings dans [30] proposent une définition légèrement différente. Le terme agent est utilisé pour décrire un matériel ou (plus couramment) un logiciel possédant les propriétés suivantes :

- *Autonomie* : les agents opèrent sans intervention extérieure
- *Capacités sociales* : les agents interagissent entre eux et éventuellement avec des utilisateurs humains
- *Réactivité* : les agents perçoivent leur environnement et répondent aux changements de celui-ci par des actions spécifiques
- *Pro-activité* : les agents sont capables de prendre des initiatives

La structure d'un agent lui permet donc d'effectuer les actions du processus PADA (Perception-Analyse-Décision-Action). Un agent est en effet une entité capable de planifier de façon réfléchie une action en fonction de ses perceptions, de ses connaissances, et de son comportement. Les agents peuvent être définis en fonction de leur caractère réactif ou cognitif :

- Les agents *réactifs* sont des agents qui réagissent uniquement à leur perception de l'environnement. Ce type d'agent est qualifié de minimaliste. Le principe de fonctionnement est du type stimuli-action : à la perception d'un stimulus particulier, l'agent fourni une réponse déterministe. C'est grâce à ce type d'agent que certains chercheurs ont modélisé le comportement global d'insectes sociaux. Ces agents réactifs sont capables de réaliser des tâches complexes lorsqu'ils sont présents en nombre suffisant. Parmi les agents réactifs, il y a émergence de propriétés globales qui ne s'expliquent pas à partir des agents.
- Les agents *cognitifs* sont plus complexes, ils sont capables de percevoir leur environnement et d'agir dessus, mais ils ont en plus des capacités de réflexion élaborées. Ces capacités leur permettent de construire un raisonnement, une planification ou d'utiliser des modes de communication complexes. Ils pouvant interagir des connaissances ou savoir-faire. Les agents cognitifs disposent fréquemment d'une représentation explicite des autres agents. Cette représentation est désignée par le terme de réseau d'accointances.

Afin de modéliser au mieux les caractéristiques humaines des agents, un nouveau type d'agent cognitif à été introduit. Les agents *VON BDI* sont des agents cognitifs particuliers qui possèdent des caractéristiques humanisées :

- *Value* : les principes gouvernant le comportement de l'agent sont fédérés sous ce terme. Par exemple, l'agent communiquera en priorité avec les agents avec lesquels il a l'habitude de communiquer. La notion d'habitude est donc exprimée par ce concept ;
- *Obligation* : cette notion correspond aux obligations internes de l'agent. Par exemple, l'agent est obligé de fournir ses connaissances aux autres agents s'ils lui demandent. La notion de devoir est donc exprimée par ce concept ;
- *Norm* : la norme définit un comportement basique de l'agent. Exemple : il est raisonnable d'être poli, mais personne ne l'oblige ;
- *Belief (croyances)* : cette notion correspond à la connaissance de l'environnement et des autres agents. Les croyances peuvent être modelées au fur et à mesure de la vie de l'agent. Ce mécanisme est qualifié d'apprentissage ;
- *Desire* : cette caractéristique traduit un état de satisfaction que l'agent cherche à atteindre ;
- *Intention* : représente une planification des actions à réaliser pour atteindre le désir.

2.3.2 Les systèmes multi-agents

Un système multi-agent est composé : d'un environnement d'évolution des agents et d'un ensemble : d'agents, d'objets (manipulables par les agents), de relations permettant aux agents de dialoguer et ainsi de coopérer, d'opérations permettant aux agents de percevoir, et manipuler les objets et d'un ensemble d'opérations permettant de contrôler la réaction de l'environnement aux opérations des agents (lois de l'univers).

Il existe une approche qui décompose les *SMA* en quatre briques élémentaires :

- *Les agents* : qui concernent les modèles (ou les architectures) utilisés pour la partie active de l'agent ;
- *Les environnements* : qui sont les milieux dans lesquels sont plongés les agents ;

- *Les interactions* : qui concernent les infrastructures, les langages et les protocoles d'interaction entre agents ;
- *Les organisations* : qui structurent les agents en groupes, hiérarchiques, relations…

Cette approche est appelée l'approche voyelle en référence à la première lettre de chaque brique élémentaire (*AEIO*) [31].

2.4 La coopération dans les systèmes multi-agents

Les systèmes multi-agents sont fortement liés à la notion de coopération. L'intelligence artificielle distribuée sous-tendue par le concept des systèmes multi-agents, nécessite des échanges entre les agents. Ceux-ci coopèrent. Les systèmes multi-agents sont donc des systèmes coopérants. En ce sens, ils se prêtent bien à la simulation de situations coopératives complexes. Nous décrivons dans ce qui suite, les grands concepts et définitions de la coopération multi-agent.

2.4.1 Définitions

Les études dans le domaine de la sociologie ont montré que les humains doivent coopérer à cause de leurs capacités limitées. Nous coopérons parce que nous ne pouvons pas atteindre nos buts, ou parce que nous ne pouvons pas les atteindre de manière efficace et rapide [32].

En effet, la coopération est la forme générale d'interaction la plus étudiée dans les SMA, elle représente l'attitude sociale qui permet l'augmentation des performances de groupe [33, 34, 35]. De plus la coopération se fonde à la fois sur la complémentarité d'intérêt et la confiance [36].

Nous nous intéressons particulièrement à cette forme d'interaction, nous présentons dans ce qui suit deux définitions qui nous semblent répondre à ce dont nous avons besoin.

Définition 1 : Les caractéristiques de la coopération idéale sont aussi celles d'une coopération totale où la moindre activité est bénéfique pour autrui [37] :

- *Compréhension* : 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 ;
- *Raisonnement* : toute information (un signal interprété) doit avoir des conséquences logiques dans le système. En d'autres termes, toute information doit apporter de la nouveauté (en différence avec les informations actuellement mémorisées) ;
- *Action* : les conclusions du processus de raisonnement doivent être utiles à l'environnement du système.

Définition 2 : On dira que plusieurs agents coopèrent ou encore qu'ils sont dans une situation de coopération si l'une des conditions suivantes est vérifiée :

- L'ajout d'un nouvel agent permet d'accroître différentiellement les performances du groupe ;
- L'action des agents sert à éviter ou à résoudre des conflits potentiels ou actuels.

Ferber dans [38], propose un certain nombre de méthodes permettant de mettre en œuvre cette attitude coopérative entre agents comme :

- La communication pour échanger des informations ;
- Le regroupement physique des agents ;
- La spécialisation pour rendre certains agents plus adaptés à leur tâche ;
- La répartition des tâches, des informations et des ressources ;
- La coordination d'actions qui correspond à l'exécution de tâches supplémentaires permettant d'exécuter d'autres actions critiques dans les meilleures conditions.

De plus, certains chercheurs distinguent la coopération indirecte qui est due aux actions individuelles émises par les agents faisant évoluer l'environnement et la coopération directe résultante des signaux directs émis par les agents.

De ces définitions nous déduisons qu'un système coopératif doit avoir les caractéristiques suivantes :

- C'est un système composé d'agents coopératifs, ce sont des agents qui s'entraident, se complètent et se comprennent ;

- Ce sont des agents qui doivent accomplir la tâche pour laquelle ils ont été conçus dans un milieu coopératif ;
- Le maintien d'un degré de coopération élevé assure la survie du système.

2.4.2 Les indices de coopération

Cette section présente six indices de coopération. Ils représentent des éléments observables qui influencent notre jugement sur la coopération. Ces indices ont été déterminés à partir de l'observation de situations de coopération [39]. La liste présentée n'est pas exhaustive.

1) *La coordination d'action*, cet indice concerne l'ajustement de la direction des actions des agents dans le temps (synchronisation) et dans l'espace.
2) *La parallélisation*, cet indice est fonction de la répartition des tâches et de leur résolution concurrente.
3) *Le partage des ressources*, cet indice concerne l'utilisation des ressources et des compétences telles que l'information, les résultats, le matériel.
4) *La robustesse*, cet indice concerne l'aptitude du système à suppléer la défaillance d'un agent.
5) *La non-redondance*, cet indice témoigne du peu d'activités redondantes, par exemple d'une communication sélective.
6) *La non-persistance des conflits*, cet indice témoigne du peu de situations bloquantes; il est fonction de la capacité des agents à prévenir les conflits ou à défaut à les résoudre.

Tableau 1. Points de vue et indices de coopération

Point de vue Observateur	***Point de vue Agent***
– Coordination des actions – Parallélisation – Partage des ressources – Robustesse – Non-redondance – Non-persistance des conflits	– Augmentation de la vitesse de résolution – Augmentation de l'ensemble des tâches réalisables – Augmentation de la probabilité d'achever des tâches – Diminuer l'interférence entre les tâches

Ces indices peuvent servir de base pour établir des buts génériques de coopération. Les correspondances entre ces indices et ces buts de coopération sont données dans le tableau 1. Les indices de *parallélisation*, de *partage des ressources* et de *robustesse* constituent les

bases nécessaires aux trois premiers buts définis par [40] qui se rapportent à la vitesse de résolution, au nombre de tâches prises en compte et à la probabilité d'achever les tâches. Les indices de *coordination des actions* et de *non-persistance des conflits* se confondent dans le quatrième but de coopération qui consiste à diminuer les interférences entre les tâches.

2.4.3 Formes de coopération

Il existe plusieurs points de vue sur la coopération, selon que l'on considère que la coopération est une attitude des agents qui décident de travailler en commun [38] ou que l'on se pose comme un observateur qui interprète a posteriori les comportements en les qualifiant de coopératifs ou non à partir de critères sociaux (ou physiques), tels que l'interdépendance des actions ou le nombre de communications effectuées.

2.4.3.1 La coopération par point de vue observateur

D'un point de vue observateur, la coopération se rapporte à une valeur de jugement sur l'activité globale d'un ensemble d'agents [39]. Comme toute valeur de jugement, la coopération est une valeur subjective. Ainsi, plusieurs observateurs confrontés à une même situation peuvent estimer différemment la coopération et ils peuvent pondérer leur jugement par des qualificatifs tels que l'harmonie. Le jugement de coopération est influencé par plusieurs facteurs tels que le nombre et la persistance des conflits, la synchronisation des actions de différents agents. Nous appelons ces facteurs les indices de coopération. Les mécanismes qui permettent de pondérer ces facteurs sont appelés les processus de coopération. Parmi ces processus, les processus de coopération internes aux agents, c'est à dire relevant du comportement des agents, peuvent être distingués des processus externes aux agents, tels que ceux assurés par un superviseur. L'automatisation des processus de coopération requiert l'identification préalable des indices de coopération.

2.4.3.2 La coopération par point de vue agent

Examinons maintenant ce que désigne la coopération non plus du point de vue de l'observateur, mais du point de vue de l'agent. La question est alors de savoir ce que signifie pour un agent le fait de coopérer. Autrement dit sous quelles conditions un agent

peut-il considérer qu'il y a coopération entre lui et un autre agent. La section précédente a montré qu'à partir des indices de coopération, il était possible d'abstraire des buts de coopération. Ces buts définissent des règles générales de comportement et, les comportements coopératifs désignent les comportements des agents contribuant à leur satisfaction. Ainsi, du point de vue de l'agent, la coopération désigne une attitude qui constitue le cadre des comportements coopératifs et qui régit le comportement d'un agent vis-à-vis des autres agents. Cette attitude peut être définie par des règles de comportement telles que les buts génériques de coopération et les maximes de conversation. Cette attitude dépend d'un but commun [41].

2.4.3.3 Coopération intentionnelle et non-intentionnelle

Lors des discussions sur la coopération, la coopération intentionnelle est souvent opposée à la coopération non-intentionnelle sans pour autant que ces deux notions soient clairement définies. A l'issue de cette analyse générale de la coopération, il est donc intéressant d'examiner le rapport entre l'intentionnalité et la coopération. Les concepts d'intention et d'actions intentionnelles sont assez difficiles à appréhender en raison notamment de leurs significations multiples [42]. L'objectif de cette section n'est donc pas de définir ces concepts mais simplement de préciser les notions de coopération intentionnelle et de coopération non-intentionnelle. La coopération se rapporte à l'activité globale d'un ensemble d'agents. Par conséquent, les critères caractérisant l'action d'une façon générale s'appliquent également à la coopération. De la même façon que les actions sont différenciées suivant leur intentionnalité [43], la coopération intentionnelle peut être opposée à la coopération non-intentionnelle.

Dans le cas de la coopération non-intentionnelle (ex., la coopération des fourmis [44]), il n'est prêté aucune intention aux agents. Les actions des agents ne sont pas déterminées à l'issue d'une délibération, elles ne sont pas intentionnelles en ce sens qu'elles ne sont pas accomplies en vue de produire un effet particulier.

Dans le cas de la coopération intentionnelle, il y a une différenciation entre l'intention de coopérer et la mise en œuvre de la coopération. L'intention peut être définie comme un but à long terme persistant mais révocable qui guide les délibérations et les actions des agents

[43, 45]. Dans la coopération intentionnelle, l'intention de coopérer se rapporte à l'accomplissement d'un but commun partagé par l'ensemble des agents coopérants. Les actions des agents sont intentionnelles puisqu'elles sont accomplies en vue de produire des effets qui vont contribuer à la satisfaction du but commun à l'origine de la coopération. Les buts génériques de coopération énoncés précédemment sont caractéristiques de la coopération intentionnelle, contrairement aux indices de coopération qui peuvent s'appliquer aux deux formes de coopération. Le modèle d'agents coopératifs développé dans cette étude met en œuvre une coopération intentionnelle. Dans ce modèle, les intentions des agents se rapportant à la coopération sont déterminées par les engagements.

2.4.4 Méthodes de coopération entre agents

La coopération entre agents est un important concept vis-à-vis le fonctionnement d'un SMA. En effet, une résolution distribuée d'un problème est le résultat de l'interaction coopérative entre les différents agents du système. Dans le cadre d'une telle dynamique collective, un agent doit disposer en plus de la connaissance reflétant son degré d'implication dans cette dynamique (croyance, buts, intentions, engagements, modèle de soi et d'autrui) d'un certain nombre de compétences nécessaires pour la coopération. Il doit pouvoir :

- Agir dans un environnement ;
- Communiquer directement ou indirectement avec d'autres agents ;
- Percevoir partiellement son environnement ;
- Mettre à jour le modèle du monde environnant ;
- Intégrer des informations venant des autres agents ;
- Interrompre ses tâches pour aider d'autres agents ;
- Déléguer la tâche qu'il ne sait pas résoudre à un autre dont il connaît les compétences ;
- Satisfaire ses objectifs en fonction des éléments précédents.

Ces caractéristiques forment les qualités essentielles d'un agent coopératif. Nous présentons dans ce qui suit, quatre approches pour la coopération entre agents : la

négociation de contrats, l'échange de résultats intermédiaires, l'approche organisationnelle et la planification locale.

2.4.4.1 Coopération par négociation de contrat

La négociation est la technique la plus employée pour résoudre des vues incohérentes ou pour atteindre un accord sur la façon de travailler ensemble. Dans les approches fondées sur le principe de la négociation, des alliances temporelles sont définies entre agents dans un réseau. Deux formes d'allocation de sous-tâches fondées sur la négociation sont : le *Contract Net Protocol* [46, 47] et la *négociation multi-niveau* [48] qui implique plusieurs itérations du processus de négociation. Dans ces approches, les rôles et les tâches sont alloués de manière dynamique en fonction de la disponibilité des ressources et des capacités effectives des agents.

Du fait du manque de vue globale de l'état du réseau à un instant donné, des tâches similaires risquent également d'être exécutées dans deux contrats différents. Enfin, un risque de récursivité peut apparaître, du fait que les agents peuvent assurer des rôles différents pour différents contrats.

2.4.4.2 Coopération par échange de résultats intermédiaires

Ici, l'objectif est de traiter l'existence de vues inconsistantes. Dans des systèmes à base de connaissances importantes, il est en effet improbable qu'une base de données totalement consistante puisse être maintenue. L'existence d'inconsistances est ainsi inévitable et peut même permettre de considérer des interprétations différentes. Les inconsistances sont alors résolues en échangeant les résultats de haut niveau issus de l'interprétation de données de bas niveau. Cette résolution des ambiguïtés forme une partie intégrale de la tâche de résolution de problème. Permettre un libre-échange d'informations pour résoudre les ambiguïtés conduit rapidement à des coûts de communication excessifs [49]. Les auteurs dans [50] a montrent comment réduire la communication de solutions partielles à un petit nombre d'agents.

2.4.4.3 Coopération par approche organisationnelle

L'approche organisationnelle constitue un compromis entre la négociation de contrat et le partage de résultats intermédiaires en définissant des alliances parmi les agents appartenant à la même organisation : les agents possèdent ainsi une vue de haut niveau de leur rôle dans le processus de résolution et de leurs relations avec les autres. Ces rôles et relations sont pré-assignés d'une façon plutôt permanente, chaque fois qu'une nouvelle organisation est créée pour exécuter une tâche.

Des stratégies de contrôle hiérarchiques peuvent être développées comme la technique de tableau noir. Dans ce type de contrôle, les sources de connaissances peuvent communiquer à travers le tableau noir en utilisant un schéma hiérarchique des connexions existantes entre elles. Seuls des messages de contrôle peuvent être échangés entre les sources de connaissances ; les données doivent toujours être partagées à travers la base de données.

Le problème principal des agents travaillant en coopération est que des situations conflictuelles surviennent fréquemment à cause d'environnements imprévisibles et changeants constamment. La capacité à reformuler les buts et à réallouer les tâches pour résoudre les inconsistances qui peuvent survenir dans une interprétation est essentielle. C'est pourquoi certaines approches permettent de modifier dynamiquement la structure organisationnelle. Par exemple, les auteurs dans [51] utilisent des structures organisationnelles pour réduire les échanges d'informations entre les agents. Ces structures sont élaborées de manière dynamique et concurrente par les agents. En plus, les auteurs dans [52] suggèrent qu'un méta-niveau de contrôle est nécessaire pour guider la génération de nouvelles structures et pour identifier les échecs des structures existantes.

2.4.4.4 Coopération par planification locale

La planification dans les systèmes d'IA classique repose sur l'hypothèse d'un univers statique (seules les actions du planificateur sont considérées); cet aspect est incompatible avec l'approche multi-agents [53]. Les SMA, en revanche, offrent des possibilités de négociation autorisant une gestion locale des conflits et une planification dynamique. En effet, du fait de l'intervention d'autres agents, un plan peut être remis en cause. L'agent doit,

pour cela, alterner entre planification et exécution et réviser des parties de son plan. La planification dans les SMA est une planification distribuée : il n'existe pas de plan global et chaque agent construit son propre plan en coordination avec les autres (cas d'agents coopératifs). Certains SMA présentent une planification centralisée ; un organe central se chargera de la gestion des conflits et de l'élaboration d'un plan global.

2.5 Conclusion

Ce chapitre a présenté le phénomène de l'activité coopérative dans un environnement multi-agents. Il apparaît clairement que le concept coopération est généralement subordonné aux concepts de coordination, d'organisation, de communication et de négociation. En plus, nous avons étudiés cette activité en termes d'indices, de formes et de méthodes. Dans le chapitre qui suit, nous exhibons les approches de spécification de la coopération existantes et nous présentons un panorama de quelques travaux récents rapportant la coopération dans un environnement multi-agents.

* * * * * * *

Deuxième partie :
État de l'art

"The problem is not the problem. The problem is your attitude about the problem"

Captain Jack Sparrow

Sommaire

* * * * * * *

Chapitre 3 Le comportement coopératif des systèmes multi-agents

3.1 Introduction

Toute modélisation passe par une étape de spécification, basée sur un modèle, et utilise un formalisme de représentation. La tendance actuelle veut que les formalismes graphiques, sous forme de diagrammes, aient plus de succès que les représentations purement textuelles [54]. Cette prévalence se justifie dans notre cas par le fait que les protocoles de coopération constituent une perspective relativement abstraite, à l'échelle d'un agent, ce qui les rend propices à être décrits par des formalismes graphiques. Bien entendu un formalisme textuel peut toujours y être associé.

Nous avons parlé dès le début que notre principal objectif est développer un modèle de coopération entre agents, mais pour que la vision soit plus claire nous allons citer et expliquer toutes les approches de spécification de la coopération existantes sur terrain pas seulement les approches formelles.

Ce chapitre contient trois sections principales ; la première présente l'état des besoins dans les techniques de spécification ; la deuxième exhibe les approches de spécification existantes et la troisième présente un panorama de quelques travaux récents rapportant la coopération dans un environnement multi-agents.

3.2 Besoins relatifs à la formalisation des comportements

Le comportement d'un agent peut être considéré selon deux points de vue : interne ou externe. Le comportement interne est relatif à l'expression des capacités de l'agent de *perception*, de *décision*, de *planification* et d'*action*, alors que le comportement externe d'un agent correspond à l'observation d'une suite d'actions exécutées par celui-ci qui sont à

la fois le résultat du comportement interne et des interactions suite aux échanges d'informations et des influences pouvant exister entre les différents agents [55].

Nous commençons par énumération des différents besoins relatifs à la formalisation des comportements, puis nous exhibons les approches de spécification où nous distinguons ceux qui sont adaptés à la concurrence de ceux qui ne le sont pas.

3.2.1 Mode de coopération

Par mode de coopération, nous entendons les règles et le jeu d'opérateurs permettant de spécifier le comportement relatif aux agents. Ceci permet par la suite de déterminer l'action à exécuter parmi l'ensemble des actions offertes tout en respectant des contraintes d'exécutions pour aboutir à une cohérence globale du système. Ces contraintes se traduisent par exemple par des points de synchronisation, des échanges de messages, etc.

3.2.2 Modèles du comportement d'agent

La description du comportement d'un agent peut être basée soit sur les occurrences d'événements soit par les états atteints par l'agent.

- *Modèle d'événements* : un événement est une action atomique. Le comportement d'un agent est perçu comme une composition de séquences d'événements qui peuvent être simple ou plus complexes telles que des interactions. Un événement est dit simple lorsqu'il est local à l'agent. Par opposition, une interaction correspond à un échange entre plusieurs agents pour accomplir l'action représentée par l'événement comme par exemple une communication. Un événement est donc soit interne soit externe. Il est externe lorsque son exécution nécessite la participation de l'environnement (extra-agent).
- *Modèle d'états* : l'état d'un système multi-agents est représenté par l'ensemble des états de chacun de ses agents. L'état d'un agent est donné par les valeurs de ses connaissances, croyances, etc., ce qui n'est pas simple à formaliser. Mais un état peut être généralement considéré par l'ensemble des tâches ou actions explicites en cours d'exécution dans un processus de résolution de problème, de conversation, etc.

Le modèle d'états et le modèle d'événements sont liés. Nous pouvons voir un événement comme une relation entre deux états successifs d'un comportement d'agent. Les modèles d'états et d'événements sont généralement fusionnés dans un modèle général de transitions étiquetées.

3.2.3 Modèle de communication

Nous distinguons deux types de modèle de communication :

- *Communication synchrone* : le système est décrit comme composition de processus avec des interactions spontanées. Chaque communication synchrone va changer l'état interne d'au moins deux agents : l'émetteur et le(s) récepteur(s).
- *Communication asynchrone* : le système est décrit par une composition de processus et un ensemble de tampons d'interconnexion ayant chacun son propre état. Chaque communication a un effet sur l'état du tampon. Une réception de message (lecteur), lorsqu'elle est possible, va consommer le message du tampon et changer l'état de l'agent consommateur et du tampon. Un envoi de message (écriture), lorsqu'il est possible, change l'état du tampon et de l'agent producteur.

3.2.4 Expression des propriétés et raisonnement logique

Les principales propriétés à spécifier dans un *SMA* sont essentiellement les propriétés de sûreté et de vivacité, qui expriment respectivement qu'une situation ou configuration indésirable ne peut se produire telle que le blocage, et qu'une situation souhaitable finit par se produire telle que l'établissement d'un contrat après une négociation, etc. Un langage de spécification doit permettre la vérification de ce type de contraintes. Certains langages comme la logique temporelle permettent d'avoir des spécifications qui peuvent être directement vérifiés puisqu'elles représentent des formules logiques (si la formulation des propriétés est permise par le langage); dans d'autres langages nous sommes obligés de passer par une représentation intermédiaire, par exemple des modèles de graphe, sur lesquels il faudrait définir une fonction d'interprétation pour vérifier que des expressions logiques sont satisfaites ou non (si le problème est décidable) dans ces représentations.

3.3 Approches de spécification pour un SMA

Les approches permettant une formalisation des concepts tels que la concurrence ont été développées initialement pour fournir une sémantique aux langages de programmation parallèles. Par la suite, le développement des recherches dans d'autres domaines tels que l'*IAD* et les *SMA* présentant les mêmes concepts que les systèmes parallèles ont permis à ces approches d'être appliquées à l'étude de ces derniers.

D'après [56], le comportement d'un protocole décrit la séquence des messages qui peuvent être échangés, notre objectif maintenant est de modéliser le comportement d'un protocole de coopération. Ce comportement est spécifié par les quatre approches de spécification suivantes : techniques semi-formelles de spécification, modèles de spécifications formelles pour les *SMA*, les langages formels de description et la logique.

3.3.1 Techniques semi-formelles de spécification

Afin de gérer la variabilité du nombre d'agents dans le protocole d'interaction, les langages graphiques considèrent non plus seulement les agents mais leur rôle dans le cadre d'interaction [54]. Les formalismes graphiques les plus utilisés dans le domaine *SMA* sont le graphe de *Dooley* et *AUML*.

Le graphe de *Dooley* [57] est un formalisme graphique qui permet d'analyser les actes de langage dans le domaine des *SMA*. Selon Parunak, chaque agent a plusieurs états mentaux ou états de connaissance au cours de la communication et des sous-problèmes sont visualisés au cours de la résolution du problème.

Le langage de modélisation unifié *UML* est une notation universelle pour la modélisation d'objets. Toutefois, il ne permet pas de modéliser un *SMA* car il y a une grande différence entre l'objet et l'agent : les agents sont en activité permanente car ils agissent selon des buts qui émergent d'eux même. C'est dans cette perspective que le langage de modélisation unifié à base d'agents *AUML* a été proposé.

En effet, le choix des formalismes adéquats qui satisfait les besoins de représentation des protocoles d'interaction, est guidé par quelques grands critères généraux. En fait, ces

formalismes doivent avoir un pouvoir d'expression explicite des différents participants dans le protocole d'interaction, permettre une représentation des alternatives et de parallélisme et permettre d'exprimer des contraintes sur l'envoi de messages.

La représentation explicite des acteurs de protocole et de leurs interventions représente le principe essentiel de l'activité coopérative. Ainsi, cette représentation constitue une approche pour abstraire le fonctionnement ou le comportement d'un groupe d'agents en termes de messages échangés. A ce stade, les formalismes de *Dooley* et *AUML* sont intéressants car ils combinent à la fois une représentation explicite des rôles, et de leur dynamicité. En effet, ces deux formalismes spécifient les rôles source ou destinataire en les préfixant aux performatifs des messages.

Dans le formalisme *AUML*, la représentation explicite des rôles (et des messages échangés) est plus intuitive que celle de graphe de *Dooley*. Ce dernier ne dispose pas du connecteur qui permet la représentation des alternatives, ce qui rend le graphe illisible, surtout dans le cas des sous-conversations pour la résolution de sous problèmes (apparition de sous graphes non connectés).

L'utilisation des éléments supplémentaires pour la représentation des branches alternatives et du parallélisme (le cas d'*AUML*) permet de distinguer plus facilement les différents chemins dans le protocole et de pouvoir les suivre plus facilement. Si l'on désire connaître tous les chemins qui partent d'une alternative, il suffit de prendre le contenu de l'élément graphique associé (qui soit un *OU*, *XOR* ou *AND*) à celui-ci. Dans le graphe de *Dooley*, il est nécessaire de suivre tous les nœuds représentent les états mentaux des participants durant les échanges.

3.3.2 Modèles de spécifications formelles

Grâce à sa simplicité, le formalisme des automates à états finis peut être considéré comme l'un des premiers et l'un des plus employés pour représenter un protocole d'interaction ou de communication [38]. L'automate se représente sous la forme d'un graphe orienté dont les nœuds sont les états de l'automate et les arcs ses transitions. Sur les arcs, on indique à la fois les évènements qui le font passer d'un état à un autre et les actions qui doivent être

entreprises lors de cette transition ; alors il s'agit d'une succession d'état ou le changement d'un état à un autre correspond à l'émission ou la réception d'un message.

L'intérêt des automates à états finis et des grammaires en général est, d'une part, de pouvoir décrire très facilement le comportement d'un agent capable de mémoriser l'état dans lequel il se trouve et, d'autre part, d'être soutenus à la fois par un support théorique important et d'avoir été utilisés dans de nombreux domaines de l'informatique. Plusieurs travaux sur la modélisation des protocoles se sont basés sur ce formalisme [58, 59], etc. Cependant, ils s'avèrent très rapidement limités pour représenter des comportements plus complexes [55], et encore moins pour décrire l'évolution d'un système multi-agents essentiellement pour deux raisons : (i) Leur capacité de calcul est limitée, du fait que leur nombre d'états est fini, (ii) Ils ne peuvent décrire que des processus séquentiels. De ce fait, tous les calculs parallèles leur sont pratiquement interdits.

Il faut passer alors à des formalismes plus puissants et adoptés à la concurrence tels que les réseaux de Petri (*RdP*) [60]. Ces derniers sont fondés sur une représentation à la fois graphique et mathématique. L'aspect graphique permet de visualiser simplement les processus et de faciliter la conception et la communication des modèles de comportement, tandis que la formulation mathématique assure à ce formalisme des fondements théoriques solides.

Les *RdP* modélisent des comportements de systèmes décrits par un ensemble d'événements où les concepts de causalité et de concurrence sont bien représentés [55]. Ils ont apporté la notion de fusion de transition (rendez-vous) qui n'existait pas pour les automates à états finis et permettent en plus d'exprimer les choix non-déterministes. Enfin, il existe non pas un modèle de *RdP*, mais toute une famille permettant de modéliser des mécanismes aussi complexes que cela s'avère nécessaire : *RdP* colorés, *RdP* à prédicats, *RdP* temporels, *RdP* temporisés, *RdP* stochastiques, *RdP* objets, *RdP* hiérarchiques, *RdP* récursifs. Par exemple si nous souhaitons distinguer entre les jetons dans un *RdP*, il est possible d'utiliser des *RdP* colorés.

En effet, les automates à états finis (*AEF*) présentent le moyen le plus simple pour décrire les protocoles de coopération car ils sont capables de mémoriser l'état de l'agent ce

qui permet de décrire facilement son comportement. Le deuxième avantage des *AEF* est qu'ils facilitent la tâche des concepteurs du protocole de coopération car ils possèdent une représentation graphique qui donne aux concepteurs une vue globale et évidente du protocole. Passons au troisième avantage des *AEF* qui concerne la validation du protocole. Nous pouvons noter que l'aspect de validation des protocoles présente le point le plus fort qui caractérise les *AEF* par rapport aux autres formalismes car la majorité des algorithmes de validation nécessitent d'abord une traduction du protocole dans un formalisme donné vers les *AEF* avant validation. Malgré tous les avantages que nous avons cité pour les *AEF*, ils présentent aussi des inconvénients surtout pour la modélisation de l'évolution d'un système multi-agents où le comportement de ce système s'avère plus complexes que les autres. D'une part nous avons dit que parmi les avantages des *AEF* est qu'ils permettent de décrire un protocole de coopération de façon simple où chaque état de l'automate correspond à une étape de la conversation, cet avantage conduit à un grand problème ou inconvénient dont le nombre d'état est combinatoire avec le nombre d'agent impliqués dans une activité coopérative et puisque aussi ce nombre d'états d'automate doit être fini malgré qu'il peut être énorme, la capacité de calcul des *AEF* est limité. Alors les *AEF* ne peuvent donc se souvenir d'une suite d'événements arbitrairement longue. D'autre part, les *AEF* ne peuvent décrire que des processus séquentiels, de ce fait, tous les calculs, toutes les actions parallèles, la synchronisation leur sont pratiquement interdits ; sachons que le parallélisme et la synchronisation sont des caractéristiques importantes dans un protocole de coopération, donc ce formalisme (*AEF*) n'est pas adapté lorsqu'on veut représenter un système coopératif malgré tous ses avantages décrits précédemment.

Généralement les *RdP* possèdent presque les mêmes avantages des *AEF* et permettent aussi d'éviter les inconvénients de ces derniers. Au niveau de la conception du protocole : les *RdP* comme les *AEF* possèdent une représentation graphique qui facilite la tâche de conception. A l'encontre des *AEF* les *RdP* possèdent la notion de modularité par exemple dans les *RdP* hiérarchique où un *RdP* est constitué d'un ensemble des *RdP*. Les critères de synchronisation et parallélismes qui sont absents dans les *AEF* existent dans les *RdP* où nous trouvons que ces derniers ont été conçus initialement afin de permettre la synchronisation et la communication des processus concurrents. Maintenant au niveau de la

validation de la conception de protocole : les *RdP* peuvent être facilement validés car il existe une conversion possible de protocole vers les *AEF*, de plus il existe un ensemble de propriétés qui peuvent être directement validés sur les *RdP*, comme la propriété qui vérifie qu'un marquage donné est accessible à partir de l'état initial.

3.3.3 Les langages formels de description

Les langages formels présentent un moyen pour spécifier des comportements des systèmes distribué et des *SMA* mais, est ce qu'ils présentent le bon moyen ou non !

Pour le langage *Z* [61, 62], il a l'avantage de l'existence des notions de réutilisabilité et modularité qui facilitent la tâche de conception puisqu'il est possible de décomposer le protocole en un ensemble de modules (composants) qu'il peut les spécifier, valider et vérifier formellement à condition que ces composants sont pris isolement. Cependant, le langage *Z* ne possède ni une représentation graphique ni d'outils pour la conception et la validation des protocoles, et en plus de ça, il n'offre pas un modèle de communication et de synchronisation ce qui constitue une des faiblesses majeures, dans la modélisation des protocoles.

Pour le langage *SDL* [63], il se distingue par une représentation graphique qui facilite la conception des protocoles et qui présente aussi une raison à son développement dans l'industrie car il facilite la lecture même par des personnes non spécialistes du formalisme. *SDL* aussi permet la réutilisabilité et il possède aussi des outils pour la conception et la validation de protocole [64]. Mais, malgré tout ça, dans *SDL* il n'y a pas de notion de synchronisation ce qui élimine l'aspect des rendez-vous entre agent.

Malgré que le langage *ESTELLE* basé sur la notion de module qui permet au concepteur du protocole de réutiliser et d'avoir une architecture modulaire, il l'oblige de concevoir les protocoles d'une manière textuelle car il ne possède pas d'une représentation graphique mais seulement d'une représentation textuelle [65]. Cet inconvénient dégrade énormément les capacités de ce langage et c'est sans doute une des raisons pour lesquelles ce formalisme a très peu de notoriété dans le monde industriel, il est essentiellement utilisé

dans le milieu universitaire sachons aussi que ce langage ne dispose pas de mécanisme de synchronisation.

LOTOS est un langage qui garantit les notions de modularité et réutilisabilité car il y a la possibilité de décomposer un processus en sous processus [66], aussi le mode de coopération des processus est obtenu à l'aide d'un jeu d'opérateurs : parallélisme, synchronisation, chois, séquence et préemption le modèle de communication entre le système et son environnement et entre les processus d'un même système est synchrone pour *LOTOS*. Il existe aussi des outils et des algorithmes pour la validation des protocoles. Cependant, la représentation textuelle de *LOTOS* rend la conception du protocole difficile. Alors toutes les caractéristiques générales de *LOTOS* incluent la puissance d'expression des éléments de services, de protocole et des interfaces, mais sa notation abstraite, difficile à comprendre, et le style algébrique ne sont pas appréciés dans le monde industriel.

3.3.4 La logique temporelle

Les logiques temporelles permettent de représenter et de raisonner sur certaines propriétés de sûreté des systèmes. En ce sens, elles sont bien adaptées à la spécification et à la vérification des systèmes réactifs et concurrents [67]. Elles pourraient donc être adaptées à la spécification des protocoles de coopération. Néanmoins, il ne faut pas oublier que ces logiques ont été conçues pour raisonner sur des propriétés usuelles des systèmes concurrents comme l'invariance, la vivacité, ou encore la précédence.

On distingue deux grandes classes de logique temporelle selon la structure du temps considérée :

- Logique temporelle linéaire (*LTL*)
- Logique temporelle arborescente (*CTL*)

Dans le cadre des logiques temporelles linéaires le temps se déroule, comme son nom l'indique, linéairement. En clair, on spécifie le comportement attendu d'un système sans réelle possibilité de spécifier plusieurs futurs possibles [54].

Dans le cas de la logique *CTL*, deux dimensions sont à prendre en considération sur les ensembles de traces : l'aspect non déterministe des choix des branches en chaque état (issus

des désynchronisations), et l'aspect linéaire d'une trace (les propriétés concernant la succession des états le long d'un chemin).

- Sur la première dimension, à partir d'un état initial donné, on peut demander qu'une propriété soit vraie pour au moins un choix de branche ou bien qu'elle soit vraie pour tous les choix, c'est-à-dire tous les futurs possibles ;
- Sur la seconde dimension, partant d'un état initial le long d'un chemin donné, on peut demander qu'une propriété soit vraie : à l'instant juste après l'état initial, au moins à un instant donné sur le chemin, tout le temps ou jusqu'à ce qu'une autre propriété devienne vraie.

L'intérêt de cette approche est de pouvoir axiomatiser le comportement et ensuite de définir les propriétés à vérifier à l'aide de formules en logique temporelle. La spécification en logique temporelle des comportements d'agents a fait l'objet de plusieurs travaux dont [68, 69] et dans [70] pour la description du protocole *Contract-Net*.

Pour le formalisme de la logique, les notions de réutilisabilité, modularité et synchronisation n'existent pas. La validation est présente par l'intermédiaire de vérification de modèle. La conception n'est pas facile avec la logique temporelle car il n'existe pas de représentation graphique du protocole et la conception nécessite de travailler avec des concepts abstraits, ajoutons aussi qu'il n'existe pas d'outils pour la conception mais des outils pour la validation, tous ces inconvénients rend la spécification avec la logique une tâche difficile.

3.4 Travaux connexes à la coopération multi-agents

D'un point de vue statique, un système multi-agent peut être considéré comme un ensemble de compétences distribuées [71]. Il est donc légal de supposer au premier lieu que l'efficacité du travail collectif d'une société d'agents est relative à la quantité d'agents ayant pris part à la résolution. Ce phénomène provient du fait que des agents trop individualistes provoquent d'autant plus de situations indésirables pour la collectivité (conflits, concurrences, ...) qu'ils sont nombreux. Dès lors, on comprend bien l'intérêt de la coopération.

Dans cette section, nous étudions quelques travaux gravitant autour de l'activité de coopération au sein un *SMA* afin de permettre d'appréhender l'état des connaissances actuelles sur cette activité. En effet, nous étudions la coopération multi-agent par opinion de Lesser [72], par investigations de Ferber, par modèle de raisonnement hétérogènes [73, 74], par adaptation du comportement [75, 76], par classification des fonctions [29], par algorithmes d'apprentissage [77], par des satisfactions personnelle et interactive [78] et par Framework de facilitation de la coopération multi-agent robotique [79, 80].

3.4.1 L'opinion de Victor Lesser

Victor Lesser en [72], propose une vue personnelle des domaines d'application clés pour les *SMA* coopératives, les problèmes intellectuels majeurs dans la construction de tels systèmes, les principes sous-jacents régissant leur conception, et les orientations majeurs et les défis pour les développements futurs dans ce domaine.

3.4.2 Les investigations de Jacques Ferber

Jacques Ferber en [38], définit la coopération comme étant un outil qui participe à la résolution des problèmes de collaboration entre les agents. En effet, il ne présente pas de définition précise de la coopération, mais plutôt une relation entre la coopération et la paire : collaboration / coordination. Cette relation, selon le point de vue de Jmaiel en [81], est due au fait que l'auteur, dans son approche, établi le concept d'interaction beaucoup plus que la coopération.

3.4.3 Les modèles de raisonnement hétérogènes

L'auteur dans [73, 74] propose un modèle de coopération nommé *MOCAH*[4] permet à plusieurs modèles de raisonnement hétérogènes de coopérer pour résoudre un problème commun. Chaque modèle de raisonnement manipule un type de connaissances particulières (le domaine) et un mode de raisonnement (les méthodes) approprié à ces connaissances. A cause des limites individuelles des modèles de raisonnement, aucun n'est capable de maîtriser un problème dans sa totalité. Cela concerne autant le nombre de problèmes

[4] MOCAH : Modélisation de la Coopération entre Agents Hétérogènes

résolus que la qualité même d'une solution. D'où la nécessité d'une intégration de modèles de raisonnement dans un même système. *MOCAH* permet une intégration distribuée de modèles de raisonnement grâce à la coopération entre un ensemble d'agents d'un *SMA*.

Dans *MOCAH*, tout modèle de raisonnement est représenté par un agent particulier. Chaque agent est caractérisé par un comportement coopératif. Un tel agent est capable de raisonner sur ses propres compétences et sur celles des autres. Il raisonne sur les caractéristiques de ses méthodes afin de détecter leurs limites et par conséquent, le moment opportun pour faire appel à d'autres agents pouvant suppléer à ses méthodes. Il raisonne également sur les compétences des autres. Il peut aussi être nécessaire de détenir des informations sur les compétences des autres afin de s'assurer que les tâches issues de requêtes émises soient réalisables par ces agents. Un tel raisonnement est appelé, raisonnement coopératif.

Une solution est construite dynamiquement par les modèles de raisonnement tout au long du processus de coopération entre les agents. Les agents communiquent entre eux, coopèrent et créent ainsi, une synergie entre les modèles de raisonnement, ce qui permet alors d'augmenter le nombre de problèmes résolus et d'améliorer les solutions obtenues.

3.4.4 L'adaptation du comportement

L'adaptation du comportement fait l'objet de plusieurs travaux, par exemple, les auteurs en [75, 76, 82] proposent une approche d'adaptation qui consiste à maintenir un degré acceptable de coopération dans le système se basant sur l'évolution de l'organisation et l'évaluation des interactions entre les différents agents. A un niveau local, ces derniers observent les perturbations que peut rencontrer le système et ce par évaluation des interactions avec leurs accointances et procèdent par conséquent à un ajustement des liens qui les unissent, un processus de réorganisation est alors observé. Cette vision locale s'avère parfois insuffisante, comme il est le cas des systèmes communautaires. Les auteurs en [75], par exemple, proposent un niveau d'adaptation global complémentaire basé sur les algorithmes génétiques et dont le but est de produire une structure organisationnelle coopérative.

3.4.5 La classification des fonctions

Les trois grandes fonctions de la coopération définies par [29] sont l'accroissement des performances, la résolution de conflits et l'amélioration de la survie. La première fonction se mesure de manière qualitative et quantitative tant d'un point de vue individuel que collectif. Les deux dernières fonctions reflètent la capacité d'adaptation d'un individu ou d'un groupe lui permettant de maintenir son intégrité fonctionnelle. Ceci est particulièrement utile en monde ouvert. Une société d'agents autonomes sera amenée à faire face aux éventuelles perturbations ou aux éventuels dysfonctionnements tendant à la détériorer.

Les mécanismes sociaux mis en place pour assurer la viabilité du groupe restreignent encore les possibilités individuelles des agents. Ils contraignent leurs comportements et les poussent à se socialiser encore plus c'est-à-dire à devenir de plus en plus spécialisés et dépendant des autres.

> *"Les organisations émergent des interactions sociales entre individus et contraignent en retour leur comportement"* [29].

3.4.6 Les algorithmes d'apprentissage

Les algorithmes d'apprentissage présentent l'objet de plusieurs travaux de modélisation de la coopération dans un environnement multi-agents [83, 84, 85], par exemple, les auteurs en [77] proposons deux algorithmes d'apprentissage qui visent à supprimer les problèmes engendrés par le fait que les agents ne possèdent qu'une vue partielle de l'espace de recherche et par la même éviter les conflits inhérents à cette lacune.

Le premier algorithme d'apprentissage proposé s'appelle *CDL*. Lorsqu'un agent désire étendre ou critiquer un modèle partiel, il doit détecter que le modèle transgresse certaines de ses contraintes locales. Ses contraintes peuvent être soit explicites soit implicites. Les contraintes explicites peuvent être partagées, auquel cas l'agent qui détecte des transgressions de contraintes explicites génère un feed-back aux agents à l'origine du modèle partiel qui a généré le conflit. Un tel feed-back permet aux agents de développer une approximation de l'espace global de recherche qui inclut à la fois leur perspective

locale (qui leur est propre) et une contrainte explicite qui est construite à partir des autres agents, ce grâce à la coopération.

Le second algorithme d'apprentissage est appelé *CBL*. Durant la phase d'apprentissage, les agents effectuent leur recherche en utilisant *CDL*. A la fin de chaque recherche, un agent mémorise la spécification du problème et les contraintes non locales qu'il a reçues par feed-back des autres agents. Ainsi, lorsqu'une nouvelle instance de problème est présentée à l'ensemble des agents, ce dernier retrouve l'ensemble des contraintes non locales qui sont sauvegardées et liées à la spécification d'un problème antérieur qui est similaire à la spécification du problème actuel. Il les ajoute à l'ensemble des demandes locales au début de la recherche. Ainsi les agents peuvent éviter des communications pour atteindre des approximations de l'espace de recherche global.

Les résultats obtenus montrent que les deux algorithmes améliorent à la fois la qualité de la solution et le temps de calcul par rapport à une recherche à l'aveugle. De plus, une fois que l'apprentissage est terminé, la méthode *CBL* diminue le temps de communication.

3.4.7 Les satisfactions personnelle et interactive

Le problème de l'évaluation, par l'agent, de ses actions et interactions constitue le cadre de recherche des auteurs dans [78]. Ils proposent un modèle des satisfactions différenciant actions individuelles et interactions avec les agents voisins. La satisfaction personnelle est calculée incrémentalement dans le temps suivant la perception de la progression de la tâche en cours. La satisfaction interactive est une évaluation de la présence des agents voisins et de leurs actions : gêne, coopération, indifférence. Cette évaluation est transformée en signaux d'intentions (attractions ou répulsions) agissant comme des champs de potentiels dynamiques. Ils proposent alors un module de sélection d'actions basé sur la compétition continue entre la satisfaction personnelle de l'agent et les signaux qu'il perçoit. Ils montrent comment l'application de cette architecture permet de résoudre des conflits spatiaux entre agents mobiles autonomes.

3.4.8 La facilitation de la coopération dans un SMA robotique

La facilitation de la coopération dans un environnement multi-agents robotique fait l'objet

de plusieurs recherches, par exemple, le travail proposé par [79, 80] présente un Framework multi-agent qui facilite la coopération dans les systèmes robotiques multi-agents. Il utilise une approche multidimensionnelle basée sur les réseaux de Petri Coloré (*RdPC*) pour modéliser les conversations complexes et simultanées entre les agents. Dans cette approche chaque agent emploie un modèle *RdPC* qui permet aux agents de suivre un plan précisant leurs interactions. Il permet également aux programmeurs de planifier pour la caractéristique concurrente. Le Framework aide les agents à identifier et à adapter différentes stratégies de sélection des équipes et des tâches d'une manière dynamique. Les agents peuvent changer leurs stratégies dans le cadre d'environnements dynamiques pour améliorer leur performance. Les auteurs ont examiné la performance des agents dans ce Framework par le développement de certaines stratégies de sélections des tâches et des équipes dans un scénario catastrophe.

3.4.9 Discussion de ces travaux

De l'ensemble de ces travaux résulte une certaine confusion sur la nature et les propriétés de la coopération. Ces travaux concernent des aspects de la coopération assez différents, ils ne se situent pas les uns par rapport aux autres, et ne s'accordent pas sur un vocabulaire donné. De ce fait, il est difficile d'en abstraire une conception générale de la coopération. Les différents travaux étudiés, montrent à l'adhésion, que la coopération est bénéfique pour obtenir un comportement proche de l'optimal. Ce résultat n'est pas trivial, car il est possible de croire que l'augmentation du nombre d'agents (non coopératifs certes) dans une société est suffisante pour accroître les performances du système. Cependant, divers travaux ont montré que le caractère individualiste des agents allait à l'encontre de ce phénomène. Dès lors, nous comprenons immédiatement le caractère dominant de la coopération. Faire coopérer les agents en leur attribuant une attitude sociale coopérative nous semble également une bonne méthode. Les agents n'ont alors pas le choix d'avoir un comportement autre que celui imposé par leurs attitudes sociales. Le résultat qui en découle est donc à coup sûr bénéfique pour la collectivité.

3.5 Notre opinion sur la coopération multi-agents

D'après l'étude des différents travaux rapportant la coopération dans un *SMA*, nous constatons l'absence d'une véritable identification des caractéristiques de l'activité coopérative dans un *SMA*, ces caractéristiques nous aidées à résoudre parfaitement cette activité sociale. Par conséquent, nous définissons un agent coopérant comme étant entité (logiciel ou matériel) qui agir dans un environnement (éventuellement d'autres agents), qui a des compétences et des ressources lui permettant d'effectuer des tâches individuelles et à coopérer avec d'autres agents pour atteindre un but commun. Ce but commun est atteint par la réalisation des buts partiels des agents coopérants. D'après cette vision d'agent coopérant, nous pouvons extraire trois caractéristiques essentielles d'un agent, à savoir : les *compétences*, les *ressources* et les *buts*. Dans notre opinion, les compétences et les buts sont étroitement liés à un agent, alors que les ressources sont considérées comme des outils nécessaires pour la réalisation de ces buts partiels.

3.6 Conclusion

Nous avons essayé le long de ce chapitre de présenter les différents formalismes existants qui nous permettent de spécifier et décrire la coopération dans un *SMA*. Tous les différents formalismes ont été appliqués aux *SMA* pour une prise en compte des actions, de l'activité des agents et de l'évolution des *SMA*. En plus, nous avons présenté une étude panoramique de quelques travaux récents gravitant autour l'activité coopérative au sein d'un *SMA*. Enfin, nous avons présenté notre point de vue sur la coopération multi-agents. Dans le prochain chapitre, nous présentons l'architecture proposée basée sur la minimisation du temps consacré à la mise en œuvre de la coopération.

* * * * * * *

Troisième partie : Contributions

"Chaque science, chaque étude a son jargon inintelligible, qui semble n'être inventé que pour en défendre les approches"

Voltaire

Sommaire

* * * * * * *

Chapitre 4 CoMAS : Framework de coopération dirigé par contraints

4.1 Introduction

Pour diminuer le temps consacré à la mise en œuvre de la coopération, pour la mise en place d'un système d'aide pour la résolution des problèmes pluridisciplinaires, pour améliorer la qualité des résultats de coopération, pour exploiter les expertises des gents situés géographiquement distants dans une activités coopérative et pour avoir une meilleure productivité dans la résolution des problèmes multi-domaines, il est nécessaire de disposer d'un ensemble de standards généraux afin d'accomplir certains objectifs parmi lesquels on pourrait noter : faciliter le travail du groupe, offrir un environnement favorable aux experts intervenants pour résoudre leurs problèmes, assurer la cohérence des données échangées, assurer le sauvegarde des problèmes résolus et constituer des *BD* relatives aux résultats de résolutions, etc. Pour atteindre ces objectifs, nous avons proposons un Framework de coopération nommé *CoMAS*[5] dédié aux agents coopérants basé sur la minimisation du temps consacré à la mise en œuvre de la coopération. Notre Framework supporte la résolution des problèmes multidisciplinaires d'une manière coopérative devant les agents coopérants.

Dans ce chapitre nous allons commencer par une description générale du notre approche de coopération en spécifions les objectifs du notre approche de coopération d'un côté, et exhibons les différentes approches hybridées dans notre Framework pour atteindre leurs objectifs de l'autre côté. Ensuite, nous présentons l'architecture générale de notre approche de coopération en déterminons le rôle de chaque composante de notre système. Enfin, nous

[5] CoMAS: Cooperative Multi-Agent System

décrivons par diagrammes de séquences les phases d'une activité coopérative sous notre Framework.

4.2 Notre approche de coopération

Dans cette section, nous spécifions les contraintes qui constituent la problématique de la recherche décrite dans le manuscrit de notre thèse. Ensuite, nous exhibons les différentes approches hybridées dans la solution de coopération proposée.

4.2.1 Éléments de la problématique

L'objectif principal visé par notre recherche est la proposition d'une nouvelle solution qui supporte l'activité coopérative dans un environnement multi-agents, cette solution doit répondre aux métriques de la qualité de service, à savoirs :

- La contrainte de minimisation du temps nécessaire pour accomplir une activité coopérative ;
- La contrainte d'amélioration de la qualité des résultats de coopération.

4.2.2 Approches hybridées

En vue satisfaire les contraintes énoncées précédemment, nous proposons une nouvelle solution pour la problématique de la coopération dans un environnement multi-agents ; cette solution présente un résultat d'hybridation de plusieurs approches, à savoirs : l'approche non-communicative, l'approche par compétence et l'approche de résolution de conflit.

Nous adoptons l'approche non-communicative dans notre architecture de coopération pour la prise en charge de la contrainte de minimisation du temps nécessaire pour accomplir une activité coopérative puisque cette approche fondée sur le principe d'interdiction toutes formes de communication inter-agents coopérants qui engendre une réduction considérable du temps consacré à la mise en œuvre de la coopération.

En plus, nous adoptons l'approche par compétence dans la spécification du savoir-faire des agents coopérants pour augmenter la qualité des résultats de coopération en termes de

spécialisation parce que cette approche fondée sur le principe d'utilisation des compétences nécessaires dans un domaine dans la conception et le développement d'une activité pédagogique, elle définit et découpée en termes d'acquisition de capacités nécessaires pour effectuer une tâche [1].

Enfin, nous résolvons les conflits par l'adoption d'une technique de sélection basée sur l'évaluation des profils des agents coopérants pour augmenter l'espérance de fonctionnement du système et éviter la situation de blocage.

4.3 Architecture générale de notre approche de coopération

Dans cette section du manuscrite de thèse, nous présentons l'architecture de coopération proposée pour un *SMA*. Notre approche adopte deux acteurs principaux : la société des agents coopérants et le Framework de coopération *CoMAS*. Nous introduisons le concept de *vecteur d'état* de l'agent coopérant pour faciliter la tâche de la gestion de l'activité coopérative assurée par le Framework. La figure 5 montre l'architecture de coopération proposée.

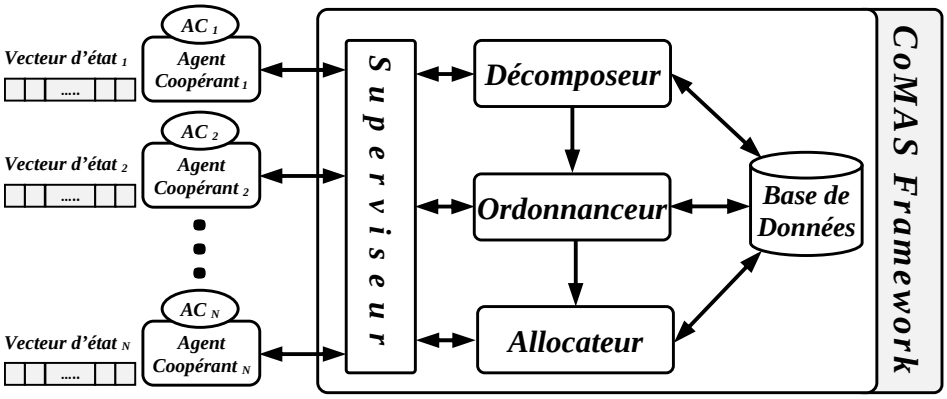


Figure 5. Architecture de coopération proposée

4.3.1 L'agent coopérant

L'engagement d'un agent dans une activité coopérative est relativement dirigé par ces compétences. Cette idée est mise en évidence par *CoMAS* via un *vecteur d'état* qui permet d'une manière explicite la spécification conceptuelle des compétences d'un agent coopérant. Ce *vecteur d'état* est composé de :

- L'identificateur de l'agent coopérant ;
- L'état actuel de l'agent coopérant (0 : passif et 1 : actif) ;
- Le compteur des compétences de l'agent coopérant ;
- Une suite des champs pour déclarer les *facteurs d'expériences* de ces compétences.

Nous incrémentons le *facteur d'expérience* d'une compétence pour chaque accomplissement d'une tâche exige la compétence en question. Formellement, nous proposons la définition suivante d'un *vecteur d'état* de l'agent coopérant :

$$\overrightarrow{SAC_i} \overset{def}{=} (Nam_{AC}, SA, Nbr_{Comp}, \{C_j\}) \quad \textbf{(1)}$$

L'état passif d'un agent coopérant signifie qu'il est libre de tout engagements et en attente l'allocation d'une nouvelle tâche, Cependant, l'état actif signifie que l'agent coopérant est occupé par la résolution d'une tâche donnée, et il indisponible actuellement.

4.3.2 Le Framework de coopération CoMAS

Notre Framework de coopération comporte trios modules essentiels : décomposeur, ordonnanceur et allocateur ; En plus d'un superviseur qui coordonne les travaux de ces derniers et une base de données.

4.3.2.1 Le décomposeur

La mission principale de ce module est la décomposition du sujet de coopération qui constitue le but global (B_G) du *SMA*, en un ensemble de sous-problèmes élémentaires qui constituants les buts partiels (B_P) des agents coopérants. On note :

$$B_G \overset{def}{=} \{B_{P1}, ..., B_{Pm} \mid B_P \in BP, m \geq 1\} \quad \textbf{(2)}$$

Où *BP* est l'ensemble des buts partiels.

4.3.2.2 L'ordonnanceur

L'agent coopérant doit disposer des compétences et des ressources pour atteindre ces buts, les compétences constituant une partie structurelle d'agent coopérant mais les ressources considèrent comme des outils offerts par le *SMA* ou par d'autres agents coopérants.

En plus, nous constations que les buts partiels dépendent étroitement avec les ressources qui nous conduisent à ordonner l'exécution de ces buts partiels en fonction de la disponibilité des ressources. Pour ce faire, l'ordonnanceur génère une *requête de précédence* pour chaque but partiel en spécifiant la compétence demandée et la liste des buts partiels qui doivent être achevés pour constituer les ressources requises.

$$RP_i \overset{def}{=} B_{Pi}\ need\ C_x\ pred\{B_{Pj}\} \text{ Where } j \geq 0 \quad \textbf{(3)}$$

4.3.2.3 L'allocateur

Comme son nom l'indique, ce module assure l'allocation des buts partiels aux agents coopérants, cette opération faite en se basant sur le critère de la compétence demandée. Pour remédier à cette problématique d'optimisation, l'allocateur adopte une stratégie similaire à un processus électorale pour choisir l'un des agents coopérants candidats, cette stratégie se résume dans la suivante figure :

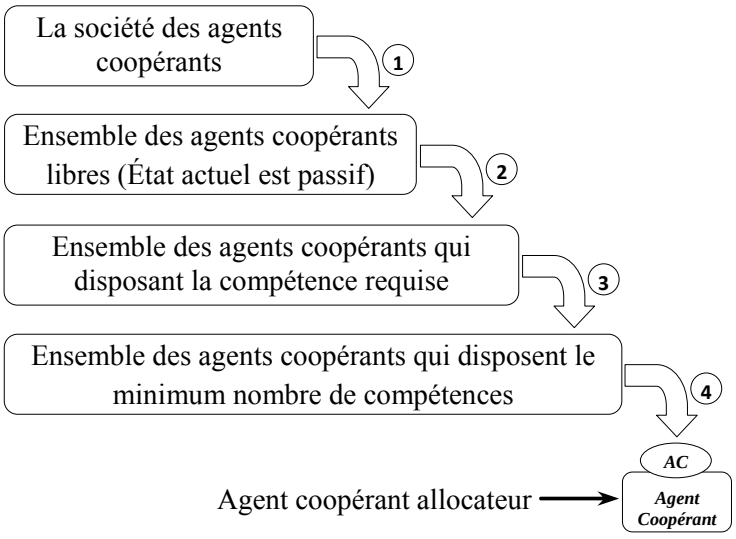


Figure 6. Politique d'allocation

Dans le tableau ci-après nous décrivons les différentes étapes de sélection en vue identifier l'agent coopérant allocateur.

Tableau 2. Notre politique d'allocation

Étapes	*Description*
1ière étape	Sélection des agents libres (État actuel est passif) ;
2ième étape	Sélection des agents coopérants qui disposant la compétence requise ;
3ième étape	Sélection des agents coopérants qui disposent le minimum nombre de compétences ;
4ième étape	Choix de l'agent coopérant allocateur qui dispose la valeur maximale du facteur d'expérience de la compétence requise.

Dans la politique d'allocation proposée, la résolution des conflits de la troisième étape de sélection est réalisée par l'adoption d'une technique de sélection basée sur le choix des agents coopérants ayant le moins nombre de compétences pour préserver les agents les plus compétents aux autres buts afin d'assurer la sureté de fonctionnement du système et d'éviter les situations de blocage ; cette proposition est sous supposition que l'agent coopérant à les compétences nécessaires pour mener à bien les travaux lui t'ont affecter. Pour cela, et dans la dernière phase de l'étape de la sélection, nous choisissons l'agent ayant la valeur maximale du *facteur d'expérience* de la compétence demandée en vue améliorer la qualité des résultats de coopération.

4.3.2.4 Le Superviseur

La tolérance aux pannes est l'une des critères de qualité de service introduite par notre Framework de coopération pour assurer le bon fonctionnement de ce dernier. Nous adoptons le module superviseur pour assurer le contrôle des agents coopérants durant l'exécution des buts partiels, en cas de disfonctionnement ou de panne de l'un de ces agents, le superviseur apport la préemption des ressources et de but partiel sur l'agent en question et provoque une nouvelle allocation. Ainsi, ce module assure la coordination des travaux des différents composants du Framework d'un part, et la gestion des communications entre le Framework et la société des agents coopérants d'autre part.

4.3.2.5 La base de données

L'échange des données et les résultats intermédiaires entre les différentes composantes du Framework fait la mission principale de notre base de données.

4.4 Processus de coopération

D'après notre approche de coopération, nous définissons le processus de coopération sous notre Framework en trois phases consécutifs, à savoir : buts, ressources et compétences. La figure 7 montre l'enchainement des phases d'un processus de coopération.

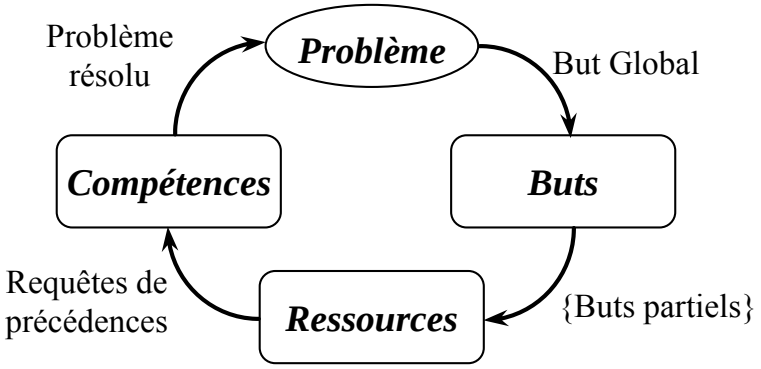


Figure 7. Processus de coopération

4.4.1 La phase des buts

Dans cette phase, nous décomposons le but global qui présente le sujet de coopération en un ensemble non vide des buts partiels qui seront résolus par la société des agents coopérants.

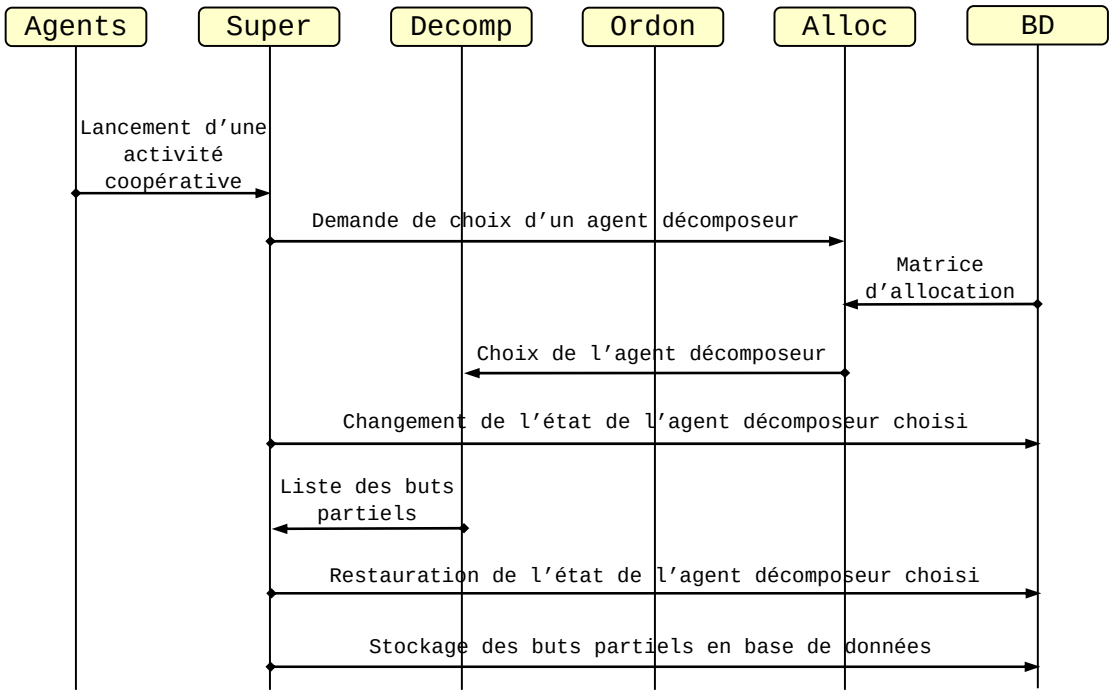


Figure 8. Diagramme de séquence décrivant la phase des buts

4.4.2 La phase des ressources

Maintenant, pour chaque but partiel identifié précédemment, nous générons à l'aide du module ordonnanceur une *requête de précédence* qui englobe l'identité de la compétence requise et la liste des ressources nécessaires pour la résolution du but partiel. Nous visons par les ressources celles qui constituent des données ou des résultats d'exécutions obtenus depuis d'autres buts partiels.

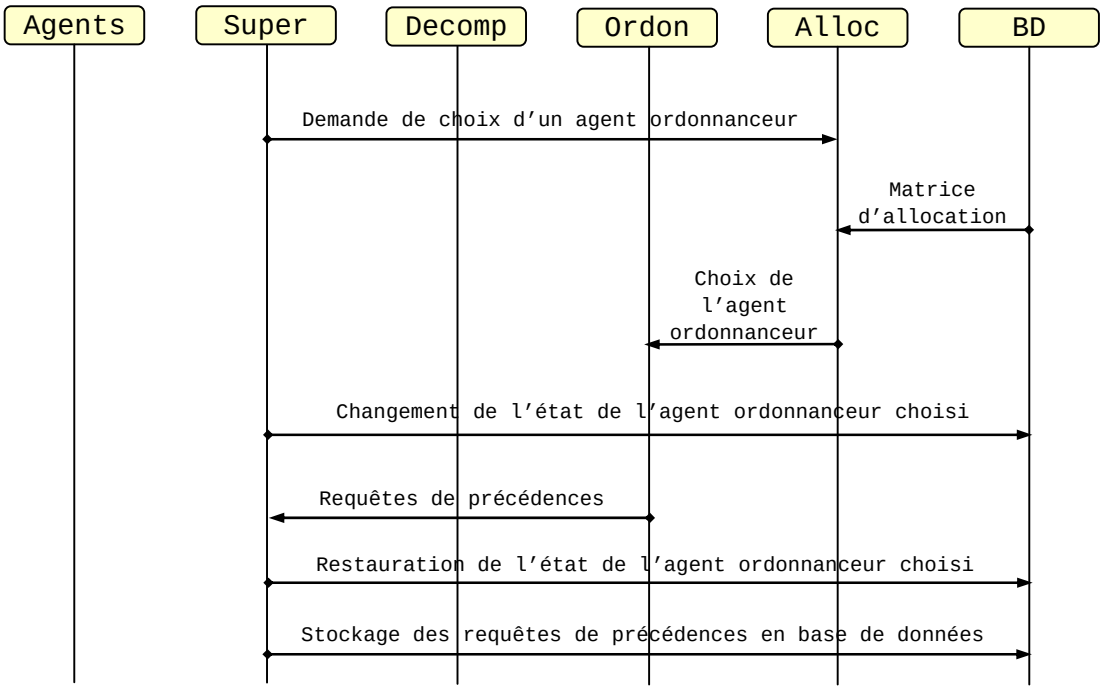


Figure 9. Diagramme de séquence décrivant la phase des ressources

4.4.3 La phase des compétences

Un but abordable est un but partiel qui dispose de toutes les ressources nécessaires à son exécution. Pour cela, l'allocateur agit sur les *requêtes de précédences* pour son identification, puis apporter une série de transformations sur la *matrice d'allocation* (collection de vecteurs d'état des agents coopérants) pour sélectionner un agent coopérant dit candidat à qui on affecte la résolution de ce but partiel.

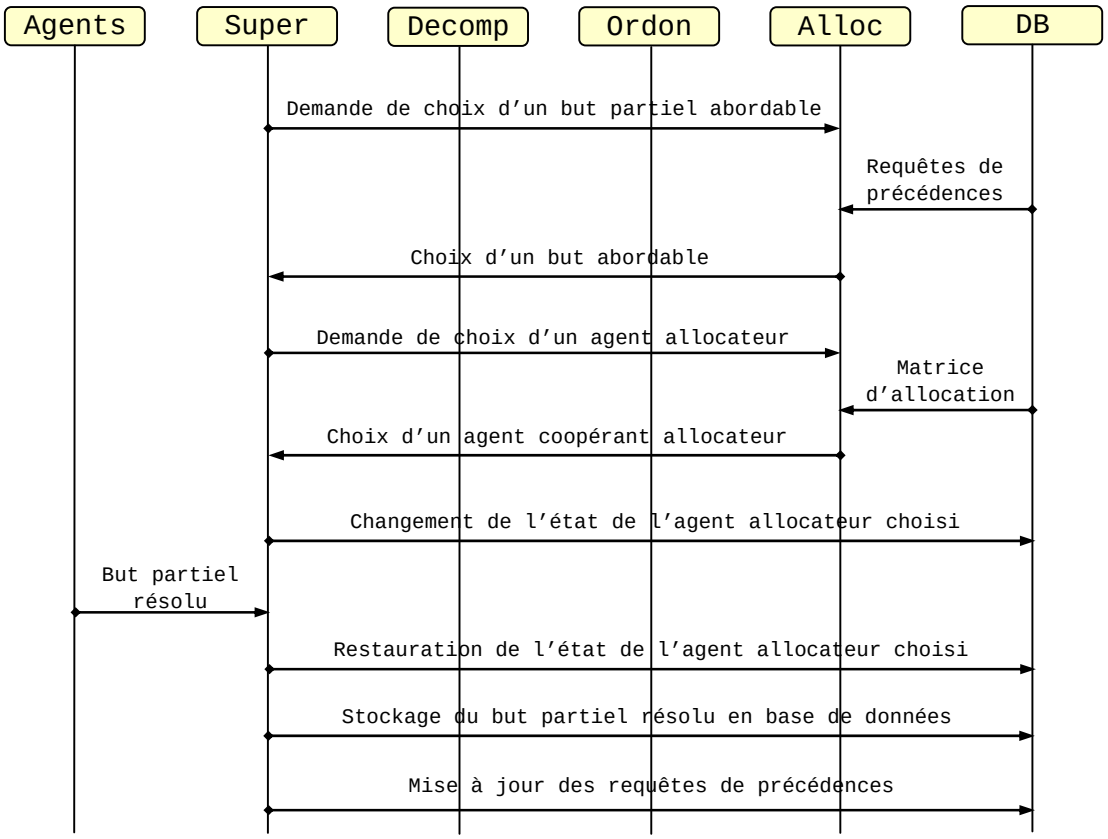


Figure 10. Diagramme de séquence décrivant la phase des compétences

4.5 Conclusion

Dans ce chapitre nous avons décrit l'architecture générale de notre Framework de coopération nommé *CoMAS*. Ce dernier permet à des agents géographiquement distants de coopérer pour résoudre un problème complexe et pluridisciplinaire d'une manière coopérative. En plus, notre politique de sélection des agents coopérants engendre une meilleure exploitation des expertises des agents coopérants en termes de spécialisation et minimisation du temps consacré à la mise en œuvre de la coopération en termes de disponibilité.

Le prochain chapitre est consacré à décrire le principe de fonctionnement de notre Framework à l'aide d'un cas réel. Pour ce faire, nous choisissons le cas des systèmes auteurs coopératif en e-Learning.

* * * * * * *

Chapitre 5 Étude de cas : Systèmes Auteurs Coopératifs en e-Learning

5.1 Introduction

En vue d'illustrer le principe de fonctionnement de notre approche de coopération à l'aide d'un cas réel, nous choisissons l'exemple du système auteur coopératif en e-Learning. Ce genre des systèmes offrent un environnement favorable au collectif d'auteurs pour élaborer leurs contenus éducatifs d'une manière coopératif. Dans ce contexte, nous proposons un système auteur basé sur notre Framework de coopération nommé CoMAS-AS. En plus, nous proposons une étude numérique pour examiner les performances de notre approche de coopération en termes de minimisation du temps consacré à la mise en œuvre de la coopération, tout en améliorant la qualité des résultats obtenus.

5.2 Les systèmes auteurs en e-Learning

Un système auteur peut être défini comme un « environnement de développement logiciel de haut niveau », entendant par-là qu'il permet en théorie, grâce à une interface graphique, de réaliser l'essentiel ou la totalité d'une application multimédia sans utiliser un langage de programmation. Nous Excluons le cas où l'auteur veut aboutir à une interaction un peu plus complexe [86].

Avec un système auteur, en principe, le temps d'apprentissage et de développement d'un logiciel multimédia est inférieur de beaucoup en temps requis comparativement à un autre système de programmation [87].

Un système auteur, selon [88], offre au concepteur pédagogique le moyen de concevoir un système d'apprentissage en fonction d'une expertise pédagogique. Ces systèmes s'élaborent généralement en tenant compte d'une théorie du design pédagogique.

Une des classifications possibles de systèmes auteurs revient à [89]. Le suivant tableau, résume les sept catégories de la classification de Murray.

Tableau 3. Classification des systèmes auteurs fondateurs

N°	*Catégories des systèmes*	*Exemples de systèmes faisant référence*
1	Systèmes de séquencement et de planification du curriculum	DOCENT, IDE, ISD Expert, Expert CML.
2	Systèmes à stratégies pédagogiques	Eon, GTE, REDEEM (et COCA), SmartTrainer.
3	Systèmes de simulation et d'entraînement	DIAG, RIDES, MITT-Writer, ICAT, SIMQUEST, XAIDA.
4	Systèmes experts et tuteurs cognitifs	Demontr8, D3 Trainer, Training Express.
5	Systèmes à connaissances multiples	CREAM-Tools, DNA, ID-Expert, IRIS, XAIDA.
6	Systèmes à usages spécifiques	IDLE-Tools/IMap, LAT.
7	Systèmes hypermédia intelligents / adaptatifs	CALAT, GETMAS, Interbook, MetaLinks.

Une autre classification présentée dans [90] classifie ces systèmes en deux catégories à savoir : les outils spécifiques au e-Learning, tels qu'*Authorware* ou *Director*, et les outils plus généralistes comme les éditeurs *HTML*, tels que *FrontPage* ou *Dreamweaver*, ou les éditeurs de simulation tels que *Flash*. Face à une évolution technologique que l'on sait de plus en plus rapide, le développement des systèmes auteurs dans une optique éducative est devenu un domaine de recherche à part entière. En effet, au cours des dernières années il y a eu des progrès significatifs dans le développement de tels systèmes de façon à concevoir des outils permettant aux utilisateurs de créer des contenus pédagogiques multimédias utilisables sur *CD-ROM* et/ou en ligne. Parmi ces systèmes citons *Toolbook* et *Serpolet Auteur*. Dans ce qui suit, nous présentons les critères et les fonctionnalités qu'un système auteur doit avoir et offrir.

5.2.1 Caractéristiques des systèmes auteurs

Quel que soit le type et le niveau de l'utilisateur du système auteur, plusieurs attributs doivent être étudiés soigneusement et aussi considérés par le concepteur d'un système auteur. Nous allons présenter ci-dessous les critères considérés comme principaux, d'après [91] :

5.2.1.1 La convivialité

La première approche avec un logiciel a un rôle psychologique fondamental pour tout utilisateur, même chez des informaticiens. En effet, si l'écran ou la présentation du logiciel n'est pas soigné, l'utilisateur peut être plus ou moins bloqué pour la suite. L'importance de l'interface homme-machine justifie la généralisation des icônes et l'intégration du multimédia dans les systèmes auteurs pour pouvoir projeter son idée, faire comprendre, expliquer, fabriquer ou modéliser. Le concepteur doit par conséquent attacher une attention particulière à la présentation de son produit.

5.2.1.2 La transparence

La gestion des données, et le fonctionnement interne du point de vue relationnel entre les différents éléments (variables, objets, etc.) doivent être complètement transparents à l'utilisateur. L'utilisateur ne doit pas se soucier ni du moment ni de l'endroit, ni du type de fonctions et librairies à inclure et à lancer pour l'exécution de ces diverses tâches. En effet, si l'utilisateur doit comprendre le fonctionnement des registres de données, la notion de système auteur n'a plus lieu d'être !

5.2.1.3 L'assistance

Comme un système auteur est supposé, entre autres, être un outil de programmation pour les non-informaticiens, il est fondamental d'assurer un minimum d'aide en ligne ou au moins une partie explicative des principales tâches ou commandes ou icônes du système. Même pour des tâches devenues banales comme la saisie de données textuelles, le système doit au moins indiquer à l'utilisateur dans quelle fenêtre et quand cette peut se faire.

5.2.1.4 L'interactivité

La communication entre le système auteur et son utilisateur, souvent sous la forme d'une assistance de test et de contrôle automatique des tâches de l'utilisateur, doit être particulièrement soignée car elle est vitale pour l'utilisateur et pour la réussite du produit sur le marché. Un système auteur doit intéresser son utilisateur et non l'ennuyer.

5.2.1.5 La fiabilité

Le système doit être le moins bloquant possible quelle que soit l'action menée par l'utilisateur sur le logiciel. Ce dernier doit pouvoir revenir en arrière, modifier ou supprimer sans difficulté tout composant de son logiciel. De même, il doit pouvoir assembler les pièces qui constituent son application dans un ordre qui ne soit pas trop contraignant, indépendamment des uns et des autres et ce à n'importe quel moment du développement.

5.2.2 Fonctionnalités des systèmes auteurs

Il existe plusieurs fonctionnalités des systèmes auteurs ; on présentera celles qui sont communes [89] :

5.2.2.1 Fonctionnalités basiques

Les fonctionnalités telles que copier/coller, trouver et annuler sont basiques mais doivent être implémentées dans un système auteur. La majorité des systèmes les possède même si annuler une action peut être parfois un problème complexe.

5.2.2.2 Utiliser des paradigmes familiers

Le fait d'utiliser des paradigmes familiers aux utilisateurs de logiciels est un principe plus qu'une fonctionnalité mais néanmoins un principe essentiel. En effet, il s'agit d'avoir un outil ergonomique. Il faut que les boutons et/ou menus soient aisément identifiables et pour cela autant reprendre les paradigmes classiques. Il est aussi important de garder des similarités avec les outils de création de contenus pédagogiques traditionnels.

5.2.2.3 WYSIWYG

Comme dans les environnements de développement rapide (*Delphi*, *Visual C++*...), l'utilisateur doit pouvoir rapidement créer et tester son système. Un rapide aller-retour édition-test est souhaitable.

5.2.2.4 Conception graphique

L'utilisateur des systèmes auteurs, est censé être un enseignant ou un formateur sans connaissances informatiques préalables. La conception graphique ou visuelle est donc incontournable. Le formalisme est représenté par des icônes ou dessins et il doit être aisément compréhensible et mémorisable. Une vision claire et intuitive de son travail doit être offerte à l'utilisateur.

5.3 CoMAS-AS : Système auteur basé sur CoMAS Framework

Dans cette section de thèse, nous illustrons le principe de fonctionnement de notre Framework de coopération à l'aide d'un exemple. Pour ce faire, nous choisissons l'exemple des systèmes auteurs coopératifs en e-Learning. Ce genre des systèmes offre un milieu de travail favorable aux auteurs qu'ils sont répartis géographiquement distant pour élaborer leurs contenus éducatifs d'une manière coopératif. La métrique de la qualité de service est assurée par le système auteur proposé en termes du temps et de qualité scientifique des contenus éducatifs car ce système est basé sur *CoMAS* Framework. La suivante figure montre l'architecture client/serveur du système auteur proposé nommé *CoMAS-AS*.

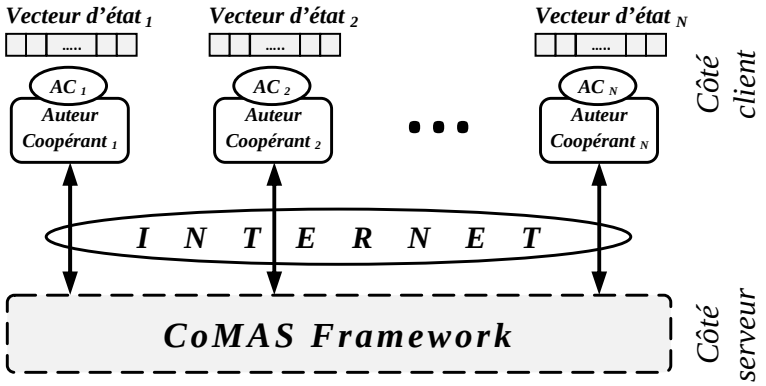


Figure 11. CoMAS-AS : Système auteur basé sur CoMAS Framework

L'architecture proposée s'articule sur deux acteurs principaux : auteur coopérant du coté client et notre Framework de coopération du coté serveur. En vue d'expliquer le principe de fonctionnement du système auteur proposé, nous supposons que la société des auteurs coopérants comporte huit (8) auteurs ayant chacun au maximum cinq (5) compétences.

Pour cela, la taille du *vecteur d'état* des auteurs coopérant égale à 8 (valeur maximale des compétences + champ de nom de l'auteur + champ du nombre des compétences de l'auteur + champ d'état actif actuel de l'auteur), formellement :

$$\left|\overrightarrow{SAC_i}\right| \stackrel{def}{=} Nam_{AC} + SA + Nbr_{Comp} + \max_j C_j \quad \textbf{(4)}$$

Les dimensions de la *matrice d'allocation* est égale à 8 lignes et 8 colonnes (nombre des lignes égale au nombre des auteurs coopérants et le nombre des colonnes égale à la taille du *vecteur d'état*) :

$$\left|\overrightarrow{MA}\right| \stackrel{def}{=} Nbr_{AC} \times \left|\overrightarrow{SAC_i}\right| \quad \textbf{(5)}$$

La suivante matrice présente notre exemple d'une matrice d'allocation.

$$\overrightarrow{MA} = \begin{bmatrix} 1 & 0 & 5 & 3 & 8 & 2 & 9 & 5 \\ 2 & 1 & 3 & 4 & 0 & 7 & 0 & 3 \\ 3 & 0 & 3 & 0 & 2 & 6 & 0 & 1 \\ 4 & 0 & 3 & 0 & 2 & 1 & 0 & 8 \\ 5 & 1 & 2 & 4 & 0 & 0 & 0 & 2 \\ 6 & 0 & 1 & 0 & 7 & 0 & 0 & 0 \\ 7 & 0 & 3 & 0 & 5 & 2 & 0 & 4 \\ 8 & 0 & 2 & 3 & 0 & 7 & 0 & 0 \end{bmatrix} \quad \textbf{(6)}$$

Maintenant, nous supposons qu'une activité coopérative est survenue au niveau de notre Framework. Ainsi, le processus de coopération se déroule comme suit :

5.3.1 Phase des buts

Premièrement, le Framework de coopération doit choisir l'un des auteurs coopérants pour décomposer l'activité coopérative à un ensemble des sous-problèmes (buts partiels). Pour cela, le décomposeur évoque une demande de sélection pour l'allocateur en vue choisir l'un des auteurs coopérants qui divise cette activité coopérative (supposons que la compétence de décomposition est : C_5). Le travail d'un allocateur dans la phase des buts comporte trois étapes consécutives : sélection, réquisition et libération.

5.3.1.1 Sélection

L'allocateur apporte une série de transformations sur la matrice d'allocation pour identifier l'auteur coopérant décomposeur, le suivant tableau montre ces transformations.

Tableau 4. Étape de sélection pour la décomposition

N°	*Description*	*Transformation de la matrice d'allocation*	
		Avant	*Après*
1ière étape	Élimination des auteurs coopérants actifs (voir la colonne 2)	1 0 5 3 8 2 9 5 2 1 3 4 0 7 0 3 3 0 3 0 2 6 0 1 4 0 3 0 2 1 0 8 5 1 2 4 0 0 0 2 6 0 1 0 7 0 0 0 7 0 3 0 5 2 0 4 8 0 2 3 0 7 0 0	1 0 5 3 8 2 9 5 3 0 3 0 2 6 0 1 4 0 3 0 2 1 0 8 6 0 1 0 7 0 0 0 7 0 3 0 5 2 0 4 8 0 2 3 0 7 0 0
2ième étape	Sélection des auteurs qui disposent la compétence de décomposition C_5 (voir la colonne 8 où le facteur d'expérience de la compétence demandée doit être non nul)	1 0 5 3 8 2 9 5 3 0 3 0 2 6 0 1 4 0 3 0 2 1 0 8 6 0 1 0 7 0 0 0 7 0 3 0 5 2 0 4 8 0 2 3 0 7 0 0	1 0 5 3 8 2 9 5 3 0 3 0 2 6 0 1 4 0 3 0 2 1 0 8 7 0 3 0 5 2 0 4
3ième étape	Sélection des auteurs qui possèdent la valeur minimale du nombre de compétences (voir la colonne 3)	1 0 5 3 8 2 9 5 3 0 3 0 2 6 0 1 4 0 3 0 2 1 0 8 7 0 3 0 5 2 0 4	3 0 3 0 2 6 0 1 4 0 3 0 2 1 0 8 7 0 3 0 5 2 0 4
4ième étape	Choix de l'auteur qui dispose la valeur maximale du facteur d'expérience dans la compétence demandée (voir la colonne 8)	3 0 3 0 2 6 0 1 4 0 3 0 2 1 0 8 7 0 3 0 5 2 0 4	4 0 3 0 2 1 0 8

5.3.1.2 Réquisition

À ce stade, l'allocateur change l'état de l' $Auteur_4$ dans leur *vecteur d'état* vers l'état actif (mettre la valeur 1 dans l'indicateur de l'état actif de l'auteur coopérant choisi).

Tableau 5. Étape de réquisition pour la décomposition

Description	***Transformation de la matrice d'allocation***	
	Avant	***Après***
Activation de l'auteur coopérant allocateur (voir la colonne 2)	4 0 3 0 2 1 0 8	4 1 3 0 2 1 0 8

5.3.1.3 Libération

Dès que l'auteur coopérant ($Auteur_4$) accompli son activité de décomposition, l'allocateur doit restaure son état vers l'état passif et incrémente le facteur d'expérience de la compétence utilisée (voir le tableau 6). On suppose que l'auteur décomposeur a défini 9 sous-problèmes (buts partiels).

Tableau 6. Étape de libération pour la de décomposition

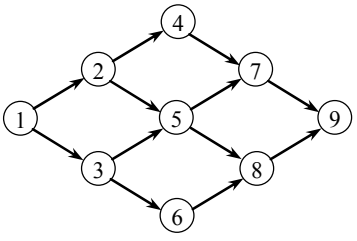
Description	***Transformation de la matrice d'allocation***	
	Avant	***Après***
Restauration de l'état passif et incrémentation du facteur d'expérience du la compétence demandée (voir les colonnes 2 et 8)	4 1 3 0 2 1 0 8	4 0 3 0 2 1 0 9

5.3.2 Phase des ressources

En deuxième lieu, l'ordonnanceur se déroule d'une manière similaire que le décomposeur pour choisir l'auteur coopérant ordonnanceur. On suppose que l'auteur coopérant ordonnanceur génère les *requêtes de précédence* suivantes :

```
BP1 Need C2 Predecessor {};
BP2 Need C4 Predecessor {BP1};
BP3 Need C1 Predecessor {BP1};
BP4 Need C5 Predecessor {BP2};
BP5 Need C3 Predecessor {BP2, BP3};
BP6 Need C2 Predecessor {BP3};
BP7 Need C1 Predecessor {BP4, BP5};
BP8 Need C3 Predecessor {BP5, BP6};
BP9 Need C5 Predecessor {BP7, BP8}.
```

(a)



(b)

Figure 12. Requêtes de précédence et leur représentation graphique

5.3.3 Phase des compétences

Finalement, nous affectons tous les buts partiels identifiés précédemment aux agents coopérants selon une stratégie composée de cinq étapes, à savoir : spécification d'un but abordable, sélection, réquisition, libération et mise à jour des requêtes de précédence.

5.3.3.1 Spécification des buts abordables

Nous définissons un but abordable comme étant le but qui dispose tous le ressources nécessaires pour leur achèvement, pour ce faire, l'allocateur sélectionne les buts partiels ayant un ensemble vide des prédécesseurs, dans notre cas le premier but partiel (B_{P1}) avec la compétence demandée (C_2).

5.3.3.2 Sélection

Pour chaque but abordable spécifié précédemment, l'allocateur procède à une série de transformations sur la matrice d'allocation pour identifier leur auteur coopérant allocateur, le tableau 7 montre ces transformations :

Tableau 7. Étape de sélection du premier but partiel

N°	*Description*	*Transformation de la matrice d'allocation*	
		Avant	*Après*
1ière étape	Élimination des auteurs coopérants actifs (voir la colonne 2)	1 0 5 3 8 2 9 5 2 1 3 4 0 7 0 3 3 0 3 0 2 6 0 1 4 0 3 0 2 1 0 8 5 1 2 4 0 0 0 2 6 0 1 0 7 0 0 0 7 0 3 0 5 2 0 4 8 0 2 3 0 7 0 0	1 0 5 3 8 2 9 5 3 0 3 0 2 6 0 1 4 0 3 0 2 1 0 8 6 0 1 0 7 0 0 0 7 0 3 0 5 2 0 4 8 0 2 3 0 7 0 0
2ième étape	Sélection des auteurs qui disposent la compétence de décomposition C_2 (voir la colonne 5 où le facteur d'expérience de la compétence demandée doit être non nul)	1 0 5 3 8 2 9 5 3 0 3 0 2 6 0 1 4 0 3 0 2 1 0 8 6 0 1 0 7 0 0 0 7 0 3 0 5 2 0 4 8 0 2 3 0 7 0 0	1 0 5 3 8 2 9 5 3 0 3 0 2 6 0 1 4 0 3 0 2 1 0 8 6 0 1 0 7 0 0 0 7 0 3 0 5 2 0 4

3ième étape	Choix de l'auteur coopérant qui possède la valeur minimale du nombre de compétences (voir la colonne 3)	1 0 5 3 8 2 9 5 3 0 3 0 2 6 0 1 4 0 3 0 2 1 0 8 6 0 1 0 7 0 0 0 7 0 3 0 5 2 0 4	6 0 1 0 7 0 0 0

5.3.1.3 Réquisition

Enfin, l'allocateur change l'état de l' $Auteur_6$ dans leur *vecteur d'état* vers l'état actif (mettre la valeur 1 dans l'indicateur de l'état actif de l'auteur coopérant choisi)

Tableau 8. Étape de réquisition du premier but partiel

Description	***Transformation de la matrice d'allocation***	
	Avant	***Après***
Activation de l'auteur coopérant allocateur (voir la colonne 2)	6 0 1 0 7 0 0 0	6 1 1 0 7 0 0 0

5.3.1.4 Libération

Dès que l'auteur coopérant ($Auteur_6$) accompli son activité de résolution du premier but partiel (B_{P1}), l'allocateur doit restaure son état vers l'état passif et incrémente le facteur d'expérience de la compétence utilisée C_2 (voir le tableau 9).

Tableau 9. Étape de libération du premier but partiel

Description	***Transformation de la matrice d'allocation***	
	Avant	***Après***
Restauration de l'état passif et incrémentation du facteur d'expérience du la compétence utilisée (voir les colonnes 2 et 5)	6 1 1 0 7 0 0 0	6 0 1 0 8 0 0 0

5.3.1.5 Mise à jour des requêtes de précédence

Finalement, l'allocateur efface la requête de précédence du but partiel achevé (B_{P1}) et ces apparences dans les prédécesseurs des autres buts partiels (voir la figure 13).

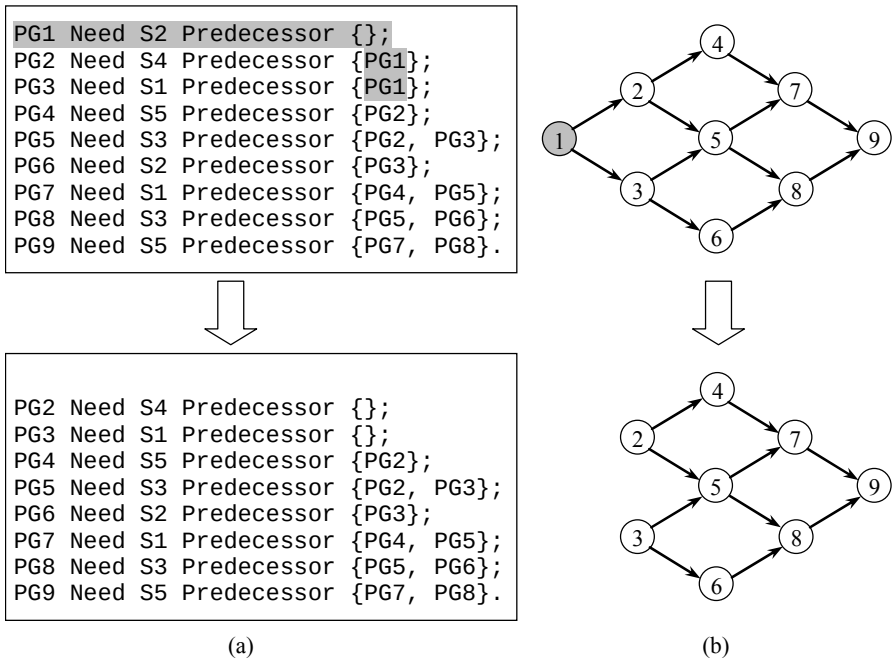


Figure 13. Requêtes de précédence après l'achèvement du premier but partiel

L'allocateur ré-exécute cette phase jusqu'à la pénurie des requêtes de précédences qui indique l'accomplissement de l'activité coopérative.

5.4 Évaluation

Nous avons comparé les performances de notre approche de coopération par rapport quelques approches préexistants par rapport au temps consacré à la mise en œuvre de la coopération.

5.4.1 Mesures d'évaluation

En effet, nous définissons trois raisons majeures qui peuvent engendre une augmentation du temps nécessaire à la réalisation de l'activité coopérative, à savoirs :

5.4.1.1 Temps d'acquisition

Nous définissons le temps d'acquisition comme étant la durée qui sépare le début de notre activité coopérative et le moment où notre but partiel devient abordable (dispose toutes les ressources nécessaire de leur accomplissement).

5.4.1.2 Temps de sélection

Nous définissons le temps de sélection comme étant la durée qui sépare le moment où notre but partiel devient abordable (dispose toutes les ressources nécessaire de leur accomplissement) et le moment de leur allocation à un auteur coopérant.

5.4.1.3 Temps d'attente

Enfin, nous définissons le temps d'attente comme étant la durée qui sépare le moment d'allocation du but partiel et celui de déclenchement de sa résolution par un auteur coopérant à cause d'accomplissement d'une tâche pendante.

5.4.2 Approches étudiées

L'élément le plus important de notre approche est la réduction du temps consacré à la mise en œuvre de la coopération. En se basent sur ce critère et afin d'évaluer les performances de notre proposition, nous avons choisi de la comparer avec les deux approches basées sur l'appel d'offre et sur la négociation.

5.4.2.1 Appel d'offre

L'approche basée sur l'appel d'offre fondée sur le principe de diffusion d'une proposition de participation pour tous les membres de la société des agents coopérants et affecte la tâche de coopération au premier agent demandeur.

5.4.2.2 Négociation

L'approche basée sur la négociation fondée sur le principe d'évaluation des propositions de coopération de tous les membres de la société des agents coopérants et affecte la tâche de coopération à l'agent coopérant qui présente la meilleure proposition.

5.4.3 Caractéristiques de la population étudiée

Nous choisissons le même exemple énoncé dans la section précédente. Dans ce contexte, nous suppose que tous les buts partiels constituant notre activité coopérative nécessitant le même temps d'exécution pour leurs achèvements.

Nous constatons que le temps d'exécution d'une tâche est généralement plus grand que le temps de consultation d'un agent coopérant. En vue mesuré les performances des approches de coopération, nous posons que le temps d'exécution est égale à *10 T* et le temps de consultation est égale à *1T*.

5.4.4 Résultats

Les données mentionnées dans le tableau ci-dessous représentent les résultats de déroulement des approches de coopération étudiées sous l'exemple des auteurs coopérants énoncé précédemment.

Tableau 10. Comparatif des approches de coopération

Tâches (Compétences)	***Approche d'appel d'offre***				***Approche de négociation***				***Notre approche***			
	NAS	***TAC***	***TS***	***TAT***	***NAS***	***TAC***	***TS***	***TAT***	***NAS***	***TAC***	***TS***	***TAT***
GP1 (C2)	6	00	01	00	6	00	07	00	6	00	01	00
GP2 (C3)	1	10	01	00	8	10	07	00	8	10	01	00
GP3 (C2)	3	11	01	00	6	11	07	00	6	11	01	00
GP4 (C3)	1	21	01	00	8	21	07	00	8	21	01	00
GP5 (C1)	7	22	01	00	2	22	07	00	2	22	01	00
GP6 (C4)	1	23	01	08	1	23	07	00	1	23	01	00
GP7 (C3)	8	41	01	00	8	33	07	00	8	33	01	00
GP8 (C1)	7	42	01	00	7	34	07	00	7	34	01	00
GP9 (C5)	4	52	01	00	4	44	07	00	4	44	01	00

Légende : NAS : Numéro de l'Agent Sélectionné; TAC : Temps d'Acquisition
TS : Temps de Sélection; TAT : Temps d'Attente.

Nous définissons le temps d'achèvement d'un bute partiel comme étant la somme des temps : d'acquisition, de sélection, d'attente et de traitement. Formellement, on note :

$$T_{BP} = T_{Acq} + T_{Sel} + T_{Att} + T_{Trait} \quad \textbf{(7)}$$

La suivante figure montre les performances des approches de coopération étudiées en termes de minimisation du temps consacré à la mise en œuvre de la coopération.

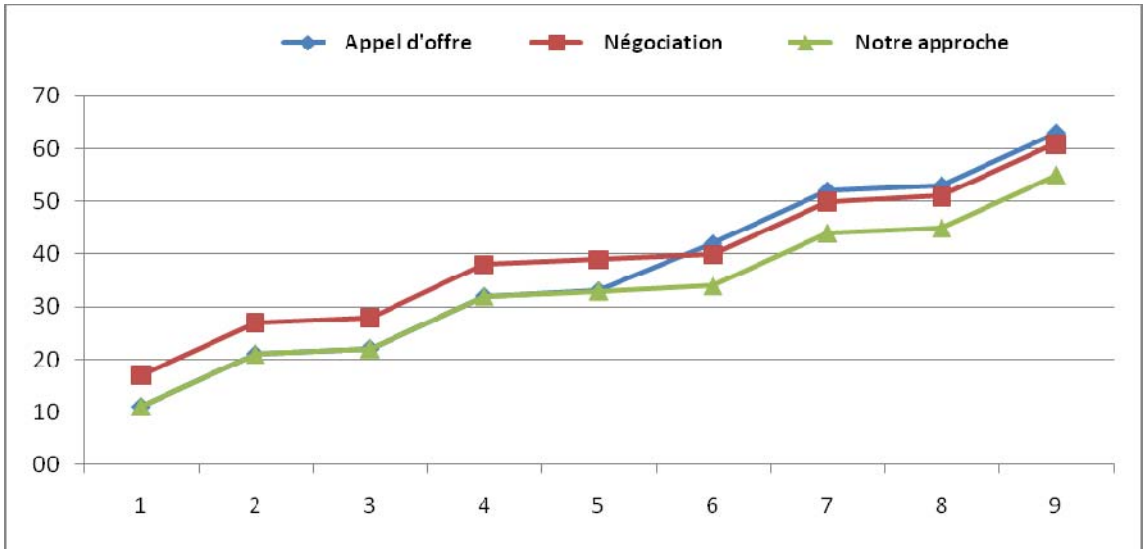


Figure 14. Performances des approches de coopération

5.4.5 Discussion

Le défit majeur du l'approche de coopération basée l'appel d'offre est l'augmentation du temps d'attente parce qu'elle affecte la tâche de coopération au premier agent demandeur qui peut engendre des situations de blocage du système. La suivante figure montre le problème de temps d'attente dans cette approche de coopération.

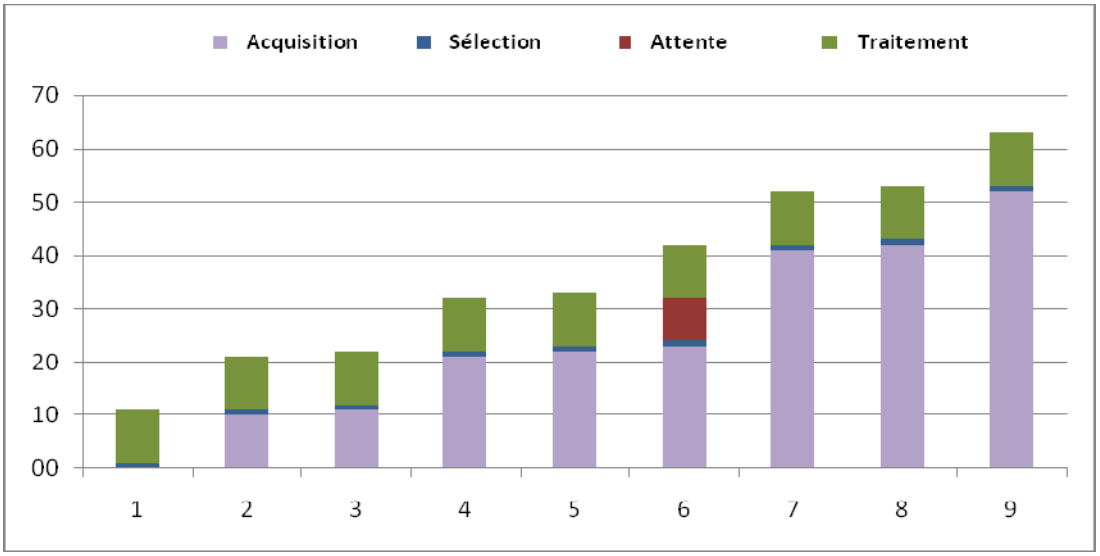


Figure 15. Le problème du temps d'attente dans l'approche basée sur l'appel d'offre

La problématique du temps d'attente est prise en charge dans l'approche de coopération basée sur la négociation puisqu'elle est fondée sur l'évaluation des propositions de coopération des agents coopérants, mais nous constatons des pertes considérables concernant le temps de sélection à cause que dans une société de *n* agents coopérant nous avons besoin *(n-1)* négociations. La suivante figure montre le problème de temps de sélection dans cette approche de coopération.

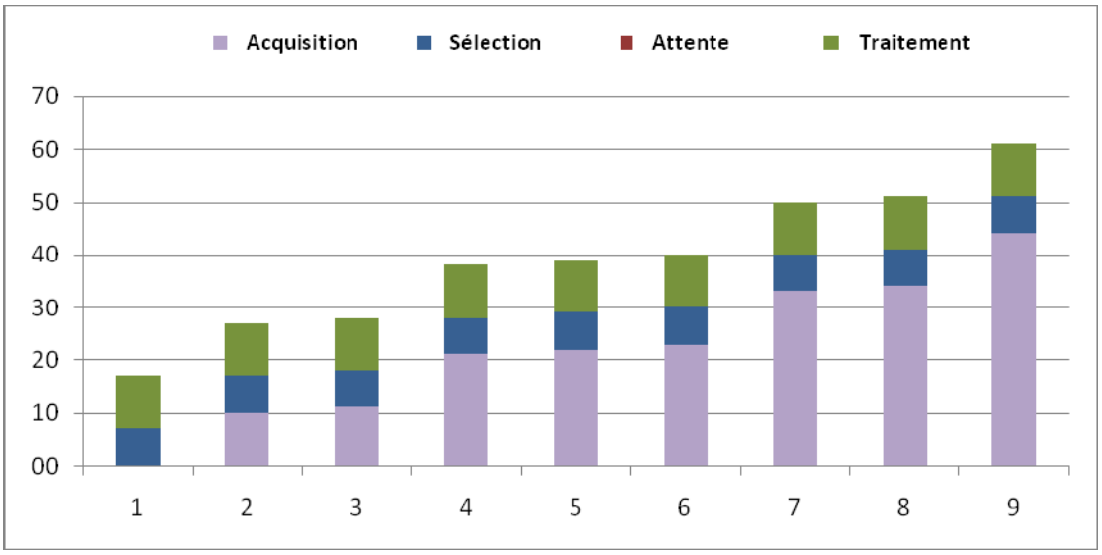


Figure 16. Le problème du temps de sélection dans l'approche basée sur la négociation

Il est bien clair que notre approche de coopération réponde aux raisons de perte énoncées précédemment. En effet, nous minimisons le temps de sélection par l'adoption de l'approche non-communicative qui interdite toutes formes de communications inter-agents. En plus, nous minimisons le temps d'attente par l'adoption d'une technique de sélection basée sur le choix de l'agent ayant le moins nombre de compétences pour préserver les agents les compétents aux autres buts afin augmenter l'espérance de fonctionnement du système et éviter la situation de blocage.

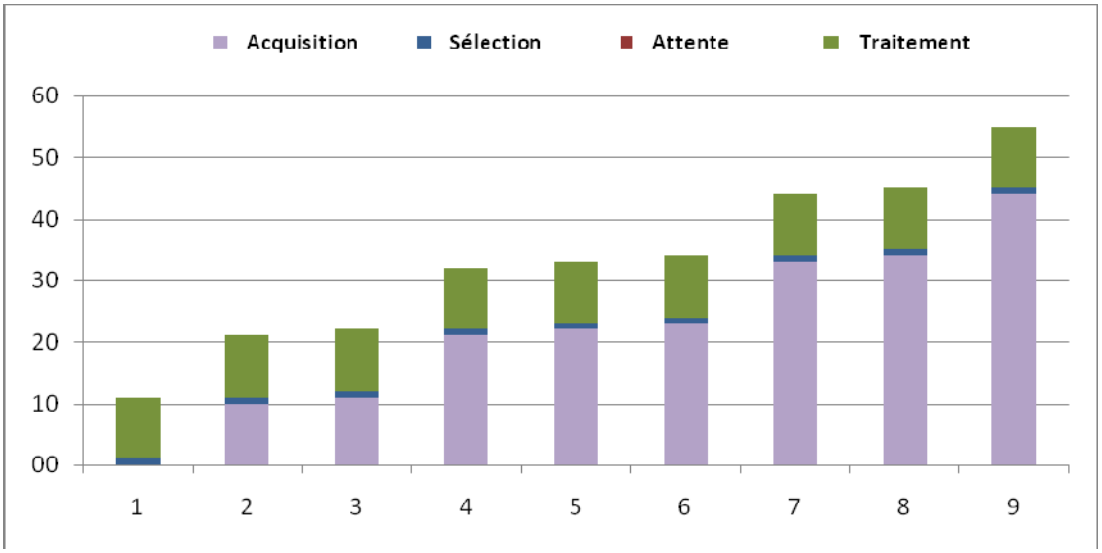


Figure 17. Efficacité de coopération de notre approche

5.5 Conclusion

Notre approche de coopération présente une meilleure stratégie à cause qu'elle minimise le maximum possible le temps nécessaire pour accomplir une activité coopérative. En effet, nous évitons le phénomène de lenteur du temps de sélection parce que nous adoptons la solution non-communicative dans notre approche. Aussi, nous évitons le phénomène du temps d'attente parce que nous adoptons une politique de sélection basée sur la compétence.

Le prochain chapitre est consacré à la phase d'implémentation de notre système de coopération. Pour ce faire, nous présentons les critères de choix des outils de développement. En plus, nous proposons une architecture logicielle de notre système pour aider les développeurs dans sa mise en œuvre.

* * * * * * *

Chapitre 6 Environnement de développement

6.1 Introduction

La phase d'implémentation de tous systèmes informatique correspond d'une part à choisir les outils de développement conformes aux spécificités du système, d'autre part, la traduction du modèle conceptuel en utilisant les outils choisis sous forme d'un logiciel opérationnel.

Dans ce chapitre, nous présentons les critères de choix des outils de développement tels que : le serveur web, le serveur de bases de données, l'environnement de développement de notre approche de coopération. Nous présentons aussi, l'architecture logicielle du système proposé en vue aider les développeurs dans la mission de sa mise en œuvre. Enfin, nous exhibons les caractéristiques de notre système en spécifiant leurs intérêts en termes d'assistance des travaux coopératifs.

6.2 Les architectures client/serveur web

Dans l'informatique moderne, de nombreuses applications fonctionnent selon un environnement client/serveur; cette dénomination signifie que des machines clientes (faisant partie du réseau) contactent un serveur (une machine généralement très puissante en termes de capacités d'entrées/sorties) qui leur fournit des services. Nous allons voir comment cette technologie permet d'exploiter au mieux les réseaux, et permet un haut niveau de coopération entre différentes machines sans que l'utilisateur se préoccupe des détails de compatibilité.

6.2.1 Le serveur web

Le vocable serveur web défini dans *wikipedia* comme étant :

> *"Un ordinateur sur lequel fonctionne un serveur HTTP est appelé serveur web".*

Le terme serveur Web peut aussi désigner le serveur *HTTP* (le logiciel) lui-même. Les deux termes sont utilisés pour le logiciel car le protocole *HTTP* a été développé pour le web et les pages web sont en pratique toujours servies avec ce protocole. D'autres ressources du web comme les fichiers à télécharger où les flux audio ou vidéo sont en revanche fréquemment servis avec d'autres protocoles.

En réalité, on trouve plusieurs technologies de serveurs Web tel que: Apache[6] *HTTP* Server de la *Apache Software Foundation*, Internet Information Services (*IIS*) de *Microsoft*, Sun Java System Web Server de *Sun Microsystems*.

6.2.2 La technologie client/serveur

L'encyclopédie universel *wikipedia* défini la technologie client/serveur comme étant :

> *"Un mode de communication entre plusieurs ordinateurs d'un réseau qui distingue un ou plusieurs postes clients du serveur : chaque logiciel client peut envoyer des requêtes à un serveur. Un serveur peut être spécialisé en serveur d'applications, de fichiers, de terminaux, ou encore de messagerie électronique".*

On peut recenser trois acteurs principaux d'une architecture client/serveur qu'ils sont :

- *Le client* : processus demandant l'exécution d'une opération à un autre processus serveur par l'envoi d'un message contenant le descriptif de l'opération à exécuter et attendant la réponse à cette opération par un message en retour;
- *Le serveur* : processus accomplissant une opération sur demande d'un client et transmettant la réponse à ce client;
- *Le middleware* : ensemble des services logiciels construits au-dessus d'un protocole de transport afin de permettre l'échange de requêtes et des réponses associées entre client et serveur de manière transparente.

[6] Apache ne doit pas son nom à la communauté amérindienne mais initialement à la transposition phonétique de *a patch* qui correspond à un ajout logiciel fait au départ sur le serveur du NCSA.

Pour définir la technologie client/serveur, il faut ajouter à cela un ensemble de propriétés.

- *Le service* : c'est le travail fourni par le serveur suite à la requête du client. Le client est donc consommateur et le serveur fournisseur de services;
- *Les ressources partagées* : le serveur est capable de servir de nombreux clients simultanément et réguler leur accès;
- *Les protocoles asymétriques* : de nombreux clients demandent un service à un serveur qui attend leurs requêtes;
- *La transparence* : le serveur peut résider ou non sur la même machine que le client, sans différence pour ce dernier;
- *La compatibilité* : l'application client ou serveur est indépendante du matériel;
- *Le faible couplage* : le client et le serveur communiquent uniquement par envoi de messages;
- *L'encapsulation des services* : le serveur choisit la manière dont il réalise le service demandé; le client se borne à définir ce qu'il désire obtenir. Ainsi l'implémentation du serveur peut être changé sans préoccuper le client;
- *L'adaptabilité* : des machines peuvent être ajoutées ou retirées du réseau; les performances des machines peuvent aussi évoluer.

6.2.3 Types d'architecture client/serveur Web

D'après Delestre en [92], la technologie client/serveur web connaît trois formes d'architectures principaux : l'architecture classique, architecture basée sur l'utilisation des programmes *CGI* et architecture basée sur l'utilisation des *servlets*.

6.2.3.1 L'architecture classique

Une architecture classique client/serveur web peut être décomposée en trois parties distinctes :

- Les fichiers *HTML* contenant les informations qui vont être présentées à l'utilisateur.

- Le serveur *HTTP*, qui réceptionne les requêtes des clients, sélectionne le bon fichier (*HTML*, image, etc.) et le renvoie au client en le préfixant d'une information appelée type mime.
- Le client qui est en fait un programme spécifique, appelé navigateur. A l'aide de ce navigateur, l'utilisateur active des liens hypertextes. Ces actions sont alors interprétées comme des requêtes et par conséquent transmises au serveur *HTTP* destinataire.

6.2.3.2 Architecture basée sur l'utilisation des programmes CGI

L'architecture que nous venons de voir brille par sa simplicité de fonctionnement et de mise en œuvre, surtout avec les outils graphiques et *WYSIWYG* d'aujourd'hui, mais elle a le gros défaut d'être très statique, très figée. En effet, tout d'abord les informations qui sont présentées à l'utilisateur sont stockées sous forme de fichier (avec tous les inconvénients que cela entraîne : problème d'indexation, de maintenance, de sécurité, etc.). Ensuite, l'utilisateur n'a que les liens hypertextes (ou *URL* saisie manuellement) pour effectuer des requêtes auprès des serveurs.

Les programmes *CGI* sont alors apparus au début des années 90. Ce sont en fait des programmes qui récupèrent la requête émise par l'utilisateur, effectuent un traitement et retournent une information au client (cette information est le plus souvent une page *HTML*, qui est crée ou sélectionnée par le programme *CGI*). De plus, la requête transmise au serveur peut être assez complexe, ce n'est plus seulement une *URL* classique, mais des informations peuvent l'accompagner. Ces informations peuvent provenir des champs d'un formulaire (que l'utilisateur aura au préalablement remplis), des cookies (données sauvegardées par certains sites) ou bien de fonctions JavaScript.

6.2.3.3 Architecture basée sur l'utilisation des servlets

Les *servlets* peuvent être vues comme une remise au goût du jour de l'architecture incluant des programmes *CGI*. En effet, les services rendus par les *servlets* sont équivalents à ceux rendus par les programmes *CGI*, mais les *servlets* tirent parti du langage qui leur a donné le jour, c'est-à-dire le langage *JAVA*. Les *servlets* ont entre autres les trois avantages suivants sur les programmes *CGI* :

- Tout d'abord, alors qu'un programme *CGI* est un processus qui est lancé à chaque requête reçue, une *servlet* est un processus qui est persistant au niveau du serveur (il peut être lancé lors du lancement du serveur *HTTP*, ou lors de la réception de la première requête).
- Ensuite les servlets lancent un thread (processus fils) par requête, alors que les programmes *CGI* lancent des processus indépendants. Cela permet par exemple de pouvoir créer une mémoire commune à toutes les requêtes, ou encore permettre aux threads de communiquer entre eux.
- Enfin, le langage *JAVA* est un langage orienté objet, alors que les programmes *CGI* sont les plus souvent programmés en *C* ou *perl*. De plus *JAVA* est un langage multiplateforme. Par exemple, une *servlet* écrite sous un environnement *Windows NT*, fonctionnera parfaitement sous un environnement *UNIX*.

6.3 Choix techniques

Notre système doit permettre au collectif d'experts coopérants la résolution des problèmes pluridisciplinaires d'une manière coopérative et à distance via le réseau internet. Cette résolution doit assure la métrique de qualité de service en termes de réduction du temps consacré à la mise en œuvre de la coopération d'une part et d'amélioration de la qualité des résultats de coopération de l'autre part. Afin de construire un système ayant les spécificités énoncées précédemment, il doit choisir des éléments de développements qui supportent le télétravail coopératif. Pour ce faire, nous avons présenté dans cette section les éléments de développement choisis, ainsi que ces critères de choix.

6.3.1 L'environnement de développement Eclipse

Eclipse est un environnement de modélisation et de développement générique, ouvert et extensible.

- *Eclipse* est générique puisqu'il permet le développement peu importe le langage utilisé (*Java*, *C/C++*, *Cobol*, *XML/XSL*, *UML*...) sur de nombreux systèmes d'exploitation (*Linux*, *Windows*, *Solaris*, *QNX*, *AIX*, *HP-UX*, *Mac OSX*);

- *Eclipse* est une plateforme ouverte puisqu'elle est offerte sous licence Common Public License, c'est-à-dire que le code source est libre de redevance n'importe qui peut le redistribuer et les travaux dérivés sont autorisés;
- *Eclipse* est également extensible puisque c'est un Framework permettant de construire et d'intégrer des outils de développement de toute nature.

Eclipse vient avec un ensemble de modules et de bibliothèques servant à la gestion des ressources. On peut créer des projets, éditer et sauvegarder des fichiers, imprimer, partager des ressources, gérer les versions à l'aide d'une interface *CVS* intégrée, etc. Eclipse offre aussi de façon standard un environnement de développement *Java* et des outils de développement de modules d'extension (voir la suivante figure).

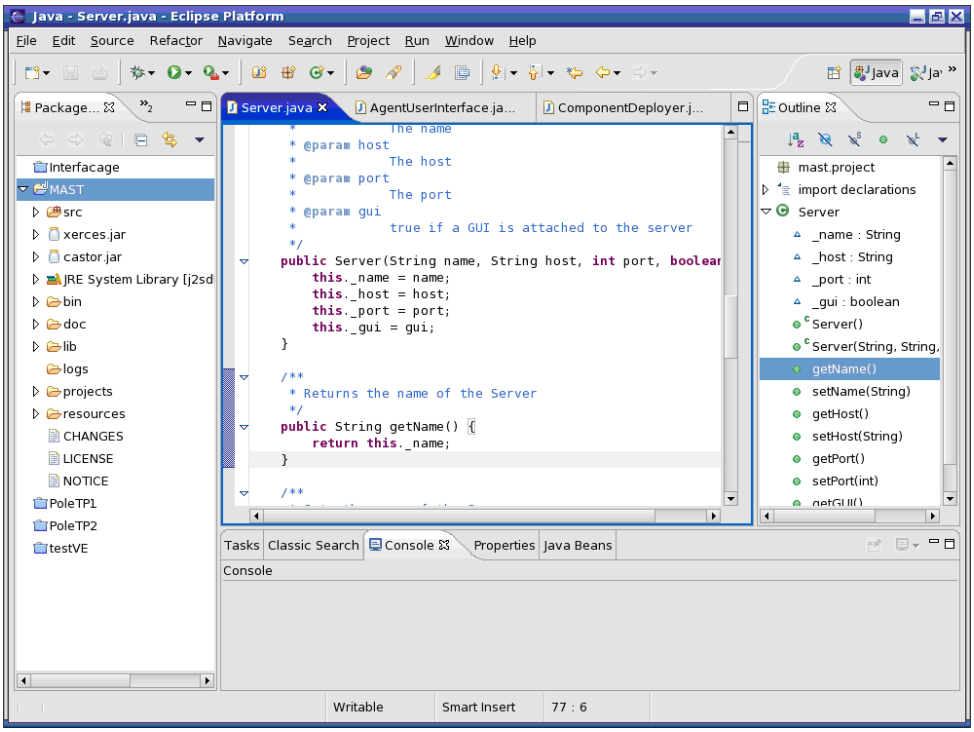


Figure 18. L'environnement de développement Eclipse

L'environnement de développement *Java*, qui n'est qu'un ensemble de modules d'extension (appelé *JDT*), offre un éditeur spécialisé, une compilation incrémentale, un débogueur et différents services tels que le *code completion*, des *code templates* et le *refactoring*.

Ce qui distingue *Eclipse* des autres *IDE* est l'extensibilité de son environnement. *Eclipse* a été conçu de manière à pouvoir facilement étendre ses fonctions à l'aide de modules d'extension tout en conservant une interface graphique cohérente. On peut ainsi charger différents modules dans *Eclipse* pour le développement en tout genre comme *Java*, *C/C++*, *Cobol*, *XML/XSL*, *UML*, etc.

Développé initialement par la compagnie *IBM*, *Eclipse* est maintenant pris en charge par un consortium de plusieurs grandes compagnies qui s'engagent à utiliser la plateforme *Eclipse*, à construire des modules d'extension gratuits et commerciaux pour *Eclipse* et à contribuer au développement du projet *Eclipse* et le supporter publiquement. Les membres initiaux du projet (novembre 2001) sont *Borland*, *IBM*, *MERANT*, *QNX Software Systems*, *Rational Software*, *Red Hat*, *SuSE*, *TogetherSoft* et *Webgain* auxquels se sont ajoutés entre autres *Oracle*, *SAP*, *HP*, *Hitachi*, *Fujitsu* et *Sybase*.

En vue de mieux comprendre la manière d'installation de la plateforme *Eclipse* et la configuration de sa traduction française, ainsi que ces mises à jours, il est conseiller de consulter le site officiel d'*Eclipse*[7].

6.3.2 Le langage Java pour implémenté notre système

Développé par *Sun Microsystem*, *Java* est un environnement de programmation orienté objet, composé de trois éléments : un langage de programmation, une machine virtuelle *JVM* et un ensemble de classes standard réparties dans différentes bibliothèques *API*. Dans ce qui suit, nous utilisons le terme *Java* pour désigner le langage de programmation. Ce langage dispose de nombreuses caractéristiques intéressantes, dont les principales sont:

- *La portabilité des programmes* : c'est-à-dire la possibilité de les faire fonctionner sur différentes plateformes (*PC*, *Mac*, *PDA*, …) mais aussi avec différents systèmes d'exploitations (*Windows*, *Linux*, *Mac OS*, …) ;
- *La richesse en bibliothèques de base* : *DOM* pour le traitement des fichiers *XML*, *XML-RPC* pour une communication inter processus indépendants ;

[7] Eclipse, le site officiel de la plateforme Eclipse disponible en ligne sur l'adresse: www.eclipse.org, date de la dernière consultation: 11/06/2014.

– *L'adaptation aux applications réparties* : *Java* est un langage qui est particulièrement adapté au développement d'applications communiquant entre elles par l'intermédiaire d'un réseau. *Java* propose de nombreuses *API* liées aux architectures distribuées : les *applets Java*, *Java RMI*[8] (une API Java standard permettant à une application Java s'exécutant sur une JVM d'invoquer les méthodes d'un objet hébergé dans une JVM distante), *Java IDL*[9] (Une API Java permettant l'interopérabilité d'une application Java), etc.

Suit à nous besoins et les contraintes exprimées jusqu'ici d'une part et les possibilités offertes par *java* de l'autre part, nous avons proposé de programmer notre système de coopération en *Java*.

6.3.3 La technique des Threads pour assister la préemption des ressources

Le meilleur moyen pour construire un *SMA* est d'utiliser une plate-forme multi-agent [93]. Une plate-forme multi-agents est un ensemble d'outils nécessaire à la construction et à la mise en service d'agents au sein d'un environnement spécifique. Ces outils peuvent servir également à l'analyse et au test du *SMA* ainsi créé. Ces outils peuvent être sous la forme d'environnement de programmation (*API*) et d'applications permettant d'aider le développeur.

Malgré les avantages consignés par la plate-forme *JADE*[10] [94] en termes des facilités offertes pour élaborer un *SMA*, nous avons préférons d'adopté la technique des *threads* de *JAVA* puisque l'exécution des comportements devant l'agent n'est pas préemptive en *JADE* c'est-à-dire lorsqu'un comportement commence, donc il s'exécute jusqu'à la fin de l'exécution du contenu de son comportement. En plus, conformément à la spécification conceptuelle de notre approche de coopération qui est fondée sur le principe de préemption

[8] Java RMI : disponible en ligne sur l'adresse : http://java.sun.com/products/jdk/rmi/, date de la dernière consultation: 17/09/2014.
[9] Java IDL : disponible en ligne sur l'adresse : http://java.sun.com/products/jdk/idl/, date de la dernière consultation: 17/09/2014.
[10] JADE – JAVA Agent DEvelopment Framework : le site officiel de la plateforme JADE disponible en ligne sur l'adresse: http://jade.tilab.com/, date de la dernière consultation: 17/09/2014.

des ressources en cas de défaillance, nous avons choisissons la technique des *threads* qui favorise la préemption de l'exécution des comportements de l'agent.

6.3.4 Le serveur Web Apache

Apache est né d'un projet réunissant via Internet un groupe de volontaires (connus sous le nom d'Apache Group) disséminés de part le monde. Ce projet visait à implémenter un serveur *HTTP* puissant, pouvant supporter une utilisation commerciale, et dont le code source et la documentation devait être librement accessible par l'intermédiaire du web. Des centaines d'utilisateurs ont par la suite contribué au projet.

En février 1995, *Apache* n'était encore constitué que du daemon *HTTP* (appartenant au domaine public) développé par *Rob McCool*, au *National Center for Supercomputing Applications*, *Université d'Illinois*. Le 1er décembre 1995, *Apache* 1.0 était mis à la disposition du public.

Aujourd'hui Apache est le serveur *HTTP* le plus utilisé dans le monde Internet et ce succès est dû d'une part à sa robustesse, et d'autre part à l'évolution actuelle des logiciels gratuits sous l'impulsion d'Internet. D'après une étude du cabinet *Netcraft*[11] sur le marché des serveurs web publiée en janvier 2016 (voir la suivante figure), *Apache* équipe 34% des serveurs web dans le monde. Son seul concurrent sérieux, avec 29% des parts du marché, est *IIS* de *Microsoft*. Le cabinet d'analyses *Giga Informations Group* recommande d'ailleurs *Apache* pour les utilisateurs soucieux de qualité sur plateformes *Intel*.

[11] Netcraft, l'étude du cabinet netcraft sur les serveurs web le plus utilisé dans le monde Internet disponible sur l'adresse: http://news.netcraft.com/archives/2016/01/26/january-2016-web-server-survey.html, date de la dernière consultation: 29/01/2016.

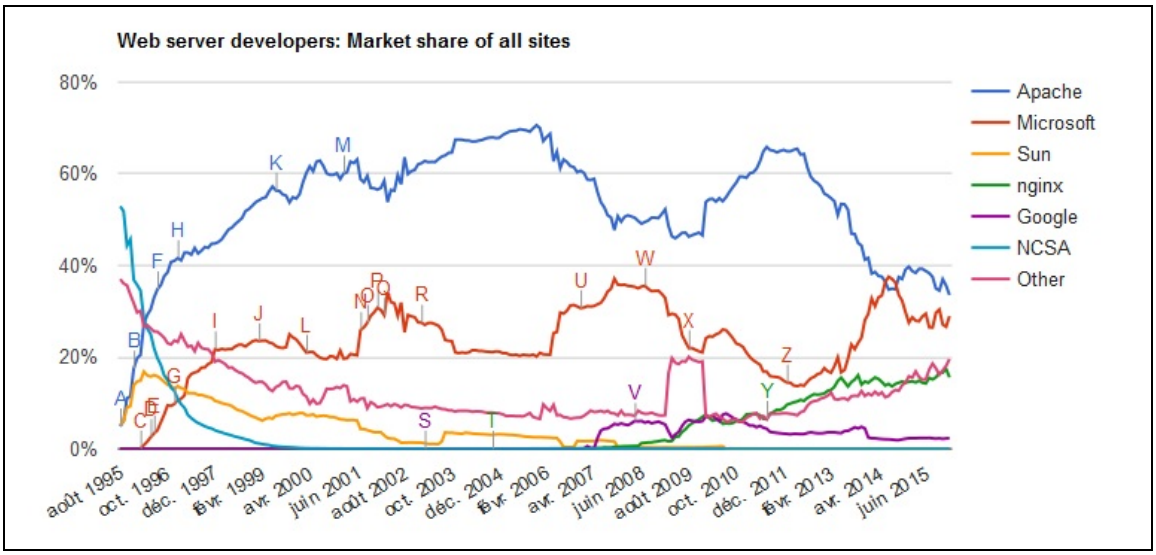


Figure 19. Les serveurs Web les plus utilisés

6.3.5 La base données eXist

eXist est une base de données *XML* native Open Source qui peut non seulement s'intégrer dans une application *Java* (grâce à la servlet *XquerServelet*), mais elle peut également être utilisée dans de nombreux autre langages via ses *API* : *REST* et *XML-RPC*.

eXist définit un modèle logique pour l'enregistrement et l'extraction des documents, ce modèle considère la totalité du document *XML* comme unité de stockage, de plus *eXist* ne repose pas sur un modèle physique particulier pour le stockage. Elle peut par exemple être bâtie aussi bien sur une base relationnelle, hiérarchique, orientée objet, ou bien utiliser des techniques de stockage propriétaires comme des fichiers indexés ou compressés.

eXist permet de formuler des requêtes *XPath* et *XQuery* et *XMLSchema* grâce à l'intégration des deux noyaux *XPath 2.0* et *XQuery 1.0*. Elle fournit également des extensions à *XPath*, comme des fonctions adaptées recherche textuelle. *eXist* offre la possibilité d'écrire des modules *XQuery* complémentaires soit directement en *XQuery* soit en *Java*. Elle supporte également quelques composants des technologies *XMLInclude*, *XPointer* et *XUpdate*. De plus *eXist* utilise un système d'indexation très performant pour améliorer la vitesse des requêtes, cette tâche est accomplie mutuellement par des indexes générés par le système ou configurés par l'utilisateur.

La flexibilité et la portabilité de ce projet font de lui l'environnement idéal pour la réalisation d'un *Data Warehouse* 100% *XML* native, de plus le fait qu'il implémente *XQuery*, et *XMLSchema*, met à notre disposition les deux outils dont en pourrait besoin, d'une part pour le questionnement de la base de données, d'autre part pour la validation des données avant leur enregistrement dans la base de données. Dernier point et non le moindre, *eXist* est Open Source, il peut donc être adapté et intégré dans une autre application.

Enfin, Il est conseiller de consulter le site officiel de la base de donnés *eXist*[12] pour connaître la façon de configuration de cette base de données sous multiples systèmes et environnement de développement.

6.4 Architecture logicielle de notre système

Malheureusement, les différentes architectures client/serveur web que nous venons de voir ne répondent par aux impératifs de notre système de coopération. En effet, au niveau fonctionnel, les exigences de notre système sont les suivantes :

- Tout d'abord nous devons fournir aux agents coopérants un outil pour spécifier leurs différentes compétences;
- Ensuite, nous devons fournir aux agents coopérants une interface graphique pour résoudre le(s) problème(s) affecté(s) sous forme d'un document XML.
- Ensuite, nous devons fournir des outils de coopération en vue de résoudre un problème d'une manière coopérative, ce qui ne peut convenablement être obtenu que par l'utilisation des mécanismes de gestion des versions et des contextes. Ces mécanismes exigent des échanges de messages entre le serveur et le client, ce qui ne peut être assuré que par l'utilisation des navigateurs web côté clients.
- Enfin, nous avons à sauvegarder les différentes versions de problèmes résolus. Nous allons donc utiliser une base de données.

Tout ceci implique donc :

[12] eXist, la base de données eXist disponible en ligne sur l'adresse: http://exist.sourceforge.net/index.html, date de la dernière consultation: 11/06/2014.

- Au niveau du client, il faut avoir un voisinage entre un navigateur et quelques applications.
- Au niveau du serveur, d'avoir outre le serveur web et le système de gestion de base de données, un outil de communication et de persistance adapté.

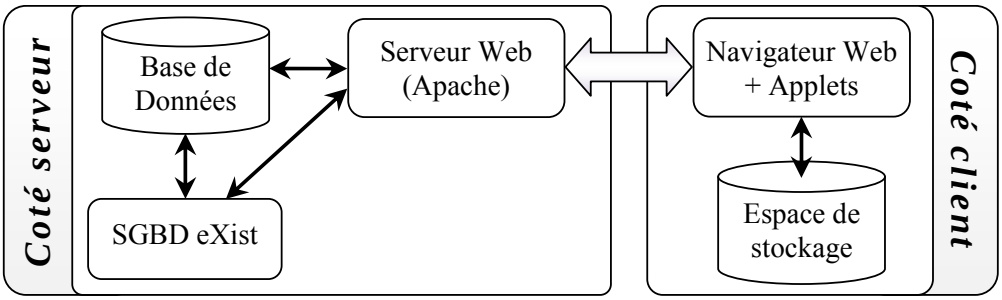


Figure 20. Architecture logicielle de notre système de coopération

On ne peut pas classer l'architecture logicielle de notre système sous une classe d'architecture client/serveur web précise car l'architecture proposée est un résultat d'hybridation de plusieurs types d'architecture afin d'atteindre aux impératifs du système. Notre architecture repose sur deux acteurs principaux : le serveur (l'objet du système qui offre le service de résolution des problèmes d'une manière coopérative) et client du service de résolution des problèmes. Nous présentons dans ce qui suit, les composants architecturaux des acteurs potentiels de notre système.

6.4.1 Côté serveur

Au niveau du serveur, nous trouvons trois composants majeurs :

- *Une base de données* : qui permet de stocker toutes les informations que nous trouvons persistantes tels que : les profils des agents coopérants, les problèmes résolus, etc.
- *Un système de gestion de base de données* : qui permet de gérer les informations stockées au niveau du serveur.
- *Un serveur web associé à des servlets* : ce serveur est en charge d'un côté de supporter les activités coopératives par le biais d'échange de message entre les agents coopérants et le serveur, et assurer le dépôt et le prélèvement des travaux de l'autre côté. Pour se faire il peut communiquer avec la base de données et l'*applet* fonctionnant sur le poste client.

6.4.2 Côté client

Au niveau du client, le navigateur Web peut exécuter trois applets :

- Une *applet* pour la gestion des contextes des agents coopérants, permettant à chaque agent de pouvoir modifier son *vecteur d'état* (l'ensemble d'informations constituant la session de l'agent).
- Une *applet* qui, en collaboration avec le serveur web, est chargé d'éditer un travail en coopération avec d'autres agents.
- Une *applet* qui, en collaboration avec le serveur web, est chargé de gérer les versions des travaux situant au niveau du serveur.

6.5 Propriétés de notre système

En vue atteindre les objectifs visées par notre système de coopération, nous spécifions dans cette section de thèse les propriétés du notre système ainsi ces manipulations possibles qui doivent êtres offertes pour les utilisateurs. En effet, nous limitons notre étude sur deux propriétés majeures de notre système, à savoir : la généricité et l'extensibilité.

6.5.1 La généricité

Notre système de coopération doit être en mesure indépendant de touts sujet de coopération particulier, à titre d'exemple : diagnostique médicale, surveillance d'un réacteur nucléaire, etc. Dans ce contexte, il faut offrir la possibilité au collectif d'agents coopérants de rejoindre un sujet de coopération en cours ou la création d'un nouveau sujet.

6.5.2 L'extensibilité

À tous moment, notre système doit permettre la mise en jeux pour les nouveaux agents coopérants. Aussi, doit apporter la préemption des ressources pour les agents coopérants défectueux en vue maintenir le blocage et assurer la survie du système. Ces fonctionnalités (mise en jeux et préemption) apportant l'aspect d'un système dynamique et extensible.

6.6 Conclusion

En nous situant dans un contexte contemporain du *TCAO* et afin de répondre à certaines des limites posées par les approches de coopération actuelles, nous avons spécifié les outils de développement ainsi justifié leurs critères de choix en vue aider les développeurs pour la mise en œuvre de notre système de coopération. En effet, nous avons proposé une architecture logicielle de notre approche de coopération en montrons son principe de fonctionnement. En plus, en vue assurer les objectifs visés par notre système, nous avons définir leur manipulations possibles qui doivent êtres offertes pour les utilisateurs.

Enfin, nous avons décidé que la réalisation pratique de notre approche de coopération est laissée pour des travaux futurs.

* * * * * * *

Conclusions et perspectives

A. Conclusions

Dans le contexte de cette thèse, nous essayons de focaliser les conclusions du travail présenté et de discuter les perspectives d'amélioration de notre approche de résolution des problèmes multidisciplinaires d'une manière coopérative. Pour ce faire, nous établissons, dans cette conclusion, un bref bilan des thèmes abordés, nous faisons ensuite le point sur notre contribution, puis nous exhibons quelques repères de chemins d'amélioration du système.

Au départ, nous avons mentionné que cette thèse s'inscrite dans le contexte des systèmes multi-agents coopératives, et plus précisément, ceux de résolution des problèmes complexes d'une manière coopérative. L'objectif visé par ce travail est la mise en place d'un système de coopération destiné aux agents coopérants (AC) qui s'engagent dans une activité coopérative pour satisfaire leurs besoins multidimensionnels et offrir un cadre de travail favorable pour accomplir leurs tâches en terme de résolution des problèmes pluridisciplinaires d'une manière coopérative. Pour ce faire, nous avons pris deux éléments principaux dans notre problématique: minimisation du temps nécessaire pour accomplir une activité coopérative et améliorer la qualité des résultats de coopération.

Dans ce contexte, nous avons proposé une nouvelle approche de coopération basée sur l'hybridation de quelques approches existantes pour satisfaire les éléments de la problématique énoncés précédemment, à savoirs :

1) *L'approche non-communicative :* nous adoptons cette approche de coopération, qui interdite toutes les formes de communication inter-AC dans notre système de coopération pour assurer la contrainte de minimisation du temps;

2) *L'approche par compétence :* nous adoptons cette technique pédagogique dans la spécification du savoir-faire des agents coopérants pour assurer la métrique de qualité des résultats de coopération en terme de spécialisation;

3) *L'approche de résolution de conflits :* nous résolvons les éventuels conflits dans la société des agents coopérants durant l'allocation des problèmes à résoudre par la métrique de la compétence requise pour assurer la contrainte de la qualité des travaux de coopération.

En effet, l'architecture de coopération proposée adopte deux acteurs principaux, à savoirs : la société des agents coopérants (*AC*) et un Framework de coopération appelé *CoMAS*. Dans ce contexte, nous avons introduit le concept de *vecteur d'état* pour déclarer explicitement les compétences de l'agent coopérant et faciliter la tâche d'allocation des problèmes à résoudre. En plus, nous avons proposé un processus de coopération comporte trois phases, à savoirs : buts, ressources et compétences.

Nous avons étudié le cas des systèmes auteurs coopératifs en e-Learning pour examiner le comportement de notre architecture de coopération à l'aide d'un cas réel. À ce stade, nous avons proposé un système auteur coopératif en e-Learning nommé *CoMAS-AS* et nous montrons son principe de fonctionnement à l'aide d'un exemple.

Dans la phase d'évaluation des performances, nous avons proposé trois raisons de perte du temps dans une activité coopération, à savoirs : temps d'acquisition, temps de sélection et temps d'attente. Ensuite, nous avons évalué les performances de notre approche de coopération en termes de temps nécessaire pour accomplir une activité coopérative et nous comparons les résultats de notre approche avec deux autres approches existant, tels que : l'approche basée sur la négociation et l'approche basée sur l'appel d'offre.

Enfin, dans la phase d'implémentation, nous avons décrit l'environnement de développement en vue aider les développeurs dans la tâche de mise en œuvre de notre système de coopération.

B. Perspectives d'amélioration

L'approche de coopération proposée ne s'arrête pas là et que des perspectives d'amélioration sont possibles. Nous indiquons ci-dessous une liste non exhaustive de ces améliorations que nous souhaitons les faire, d'abord au niveau de sa mise en œuvre, puis sur le plan conceptuel lui même.

Sur le plan d'implémentation, nous proposons la mise en œuvre de notre approche de coopération sous l'environnement de développement décrit dans le dernier chapitre de ce manuscrit de thèse. Nous proposons également de doté le système de coopération résultant par des outils de représentation des graphes pour aider l'ordonnanceur dans sa mission de rédaction des requêtes de précédences.

Sur le plan conceptuel, nous proposons également d'améliorer la politique d'allocation de notre approche de coopération par l'insertion des nouveaux paramètres dans les vecteurs d'états des agents coopérants en vue assister l'allocateur durant la résolution des éventuels conflits dans notre système.

Nous pensons que le développement de ces perspectives permet d'ouvrir d'autres portes de recherche sur notre système de coopération et l'activité coopérative en générale.

* * * * * * *

Liste des publications

A. Revue internationale

Belazoui A., Kazar O., Bourekkache S. (2016), *A Cooperative Multi-Agent System for modeling of authoring system in e-learning*, Journal of e-Learning and Knowledge Society, v.12, n.1, 135-148. ISSN: 1826-6223, e-ISSN:1971-8829.

B. Conférence Internationale

Arar C., Khireddine M.S., Belazoui A., Meguellati R. (2015), *An energy-efficient fault-tolerant scheduling algorithm based on variable data fragmentation*, in: 5th IFIP International Conference on Computer Science and its Applications (CIIA), Saida, Algeria.

Références bibliographiques

[1] Hodge S. (2007), *the origins of competency based training*, in: Australian journal of adult learning, 47(2): 179-209, available at: http://www.ala.asn.au

[2] Kanawati R. (1997), *Construction de collecticiels: étude d'architectures logicielles et de fonctions de contrôle*, Thèse de doctorat, INPG, France

[3] Bannon L. J. and Schmidt K. (1990), *CSCW: four characters in search of a context*, in: Studies in computer supported cooperative work, John M. Bowers and Steven D. Benford (Eds.). North-Holland Human Factors, in: Information Technology Series, 8: 3-16, North-Holland Publishing Co., Amsterdam, the Netherlands

[4] Hedjazi D. (2011), *Conception d'un modèle coopératif de support de la télémaintenance industrielle*, Thèse de Doctorat, Université de Batna

[5] David B. Chalon R. Delotte O. Ros J. and Boutros N. (2003), *Travail coopératif capillaire en dépannage, maintenance et interventions de crise*, in: proc. 5th International Congress on Industrial Engineering, Québec, Canada

[6] Talhi S. (2007), *Intégration des technologies de coopération et d'intelligence dans les environnements d'apprentissage à distance*, Thèse de Doctorat, Université de Batna

[7] Ellis C. A. Gibbs S. J. and Rein G. (1991), *Groupware: some issues and experiences*, in: Communications of the ACM, 34(1): 39-58, doi: 10.1145/99977.99987

[8] Karsenty A. (1994), *Le Groupware: de l'interaction homme-machine à la communication homme-homme*, in: Technique et Science Informatique (TSI), 13(1): 105-127

[9] Chatty S. Girard P. and Sire S. (1996), *Vers un support multimédia à la collaboration directe*. Technique et Science Informatique, 15(9).

[10] Grudin J, (1994), *Computer-Supported Cooperative Work: Its History and Participation*, in: IEEE Computer, 27(5): 19-26, IEEE Computer Society, doi: 10.1109/2.291294

[11] Beuscart R. Yousfi F. Dufresne E. and Derycke A. (1994), *Travail coopératif et groupware*, in: Degoulet P. and Fieschi M. (eds.), Informatique et Santé, 17(11):195-210, Springer-Verlag, Paris

[12] Ellis C. and Wainer J. (1994), *A conceptual model of groupware*, in: proc. of the 1994 ACM conference on Computer supported cooperative work (CSCW '94), ACM, New York, NY, USA, 79-88, doi: 10.1145/192844.192878

[13] Salembier P. (2002), *Cadres conceptuels et méthodologiques pour l'analyse, la modélisation et l'instrumentation des activités coopératives situées*, in : systèmes d'information et management, 7(2) : 37-56, available at: http://revuesim.org

[14] Diaz G. Mammeri Z. and Thomesse J. P. (1998), *Communication de groupe dans les applications multimédias coopératives: une synthèse*, in : NTERE'98 - colloque international sur les nouvelles technologies de la répartition, pp. 119-134, Montréal, Québec.

[15] Sharp A. and McDermott P. (2008), *Workflow Modeling: Tools for Process Improvement and Application Development*, pp. 390, ISBN: 978-1-59693-192-3, second edition, Artech House

[16] Van Hee K. and Van der Aalst M. P. (2004), *Workflow Management: Models, Methods, and Systems*, 384 pp. ISBN: 9780262720465, MIT Press, UK

[17] Villemur T. and Drira K. (1999), *Nouveaux services méthodologiques et environnements pour le travail coopératif distribué,* Rapport technique, Programme Telecom du CNRS, Projet TL97028.

[18] Diaz M. Owezarski P. and Villemur T. (1994), *une définition logique de la coopération basée sur le partage des données*, Rapport technique, LAAS rapport n°94008 Contrat CENT FT n°92.1B.178 projet CESAME lot 3

[19] Villemur T. Baudoin V. Owezarski S. and Diaz M, (1998), *A cooperative group model and its application to a multimedia virtual meeting platform*, Rapport technique, LAAS Rapport 98380.

[20] Diaz M. and Villemur T. (1993), *Présentation et classification des applications coopératives,* Rapport technique, LAAS Contrat CENT FT n°92.1B.178 Lot 3.

[21] Diaz M. Vernadat F. and Villemur T, (1996), *Spécification et réalisation formelle de systèmes coopératifs*, in : Colloque francophone sur l'ingénierie des protocoles CFIP'96, pp. 359-375, Rabat, Maroc

[22] Diaz M. and Pays G. (1994), *The Cesame project: formal design of high speed multimedia cooperative systems*, in: Annales Des Télécommunications, 49(5): 220-229, Springer-Verlag, doi: 10.1007/BF02998486

[23] Le Quere Y. Sevaux M. Tahon C. and Trentesaux D. (2003), *Modèle de coopération d'un processus de ré-ordonnancement distribué*, in: actes des JDA - Journées Doctorales d'Automatique, pp. 401-406, Valenciennes

[24] Saint-Voirin D. (2006), *Contribution à la modélisation et à l'analyse des systèmes coopératifs : application à la e-maintenance*, Thèse de Doctorat, LIFC, Université de Franche-Comté

[25] Amergé C. Boyera S. Corby O. Dieng R. Giboin A. Labidi S. and Lapalut S. (1994), *Acquisition et modélisation des connaissances dans le cadre d'une coopération entre plusieurs experts : application à un système d'aide à l'analyse de l'accident de la route*, Rapport technique, INRIA, Rapport final du contrat MRE n° 92 C 0757

[26] Koch M. and Gross T. (2006), *Computer-supported cooperative work - concepts and trends*, in: Feltz F. Otjacques B. Oberweis A. and Poussing N. (eds.), AIM Conference, GI, pp. 165-172

[27] Axelrod R, (1997), *the Complexity of Cooperation: Agent-Based Models of Competition and Collaboration*, 248 pp. ISBN: 9781400822300, Princeton University Press

[28] Biswas P. K. (2008), *towards an agent-oriented approach to conceptualization*, in: Journal of Applied Soft Computing, 8(1): 127-139, doi: 10.1016/j.asoc.2006.11.009

[29] Ferber J, (1994), *Coopération réactive et émergence*, in : intellectica, 19(4) : 19-52, available at: http://intellectica.org

[30] Wooldridge M. and Jennings N. R. (1995), *intelligent agents: theory and practice*, in: The Knowledge Engineering Review, 10(2):115-152, doi: 10.1017/S0269888900008122

[31] Demazeau Y. (1995), *from cognitive interactions to collective behaviour in agent-based systems*, in: First European Conference on Cognitive Science, Saint Malo

[32] Boussebough I, (2011), *Les systèmes multi-agents dynamiquement adaptables*. Thèse de doctorat, Université de Constantine

[33] Doran J. E. Franklin S. Jennings N. R. and Norman T. J. (1997), *on cooperation in multi-agent systems*, in: Knowledge Engineering Review, 12(3): 309-314, doi: 10.1017/S0269888997003111

[34] Gleizes M. P. Camps V. and Glize P. (1999), *A theory of emergent computation based on cooperative self-organization for adaptive artificial systems*, in: proc. 4th European Congress of Systems Science, Valencia.

[35] Velagapudi P. Prokopyev O. Sycara K. and Scerri P. (2007), *Maintaining shared belief in a large multi-agent team*, in: proc. of the Tenth International Conference on Information Fusion, Quebec City, CA

[36] Melaye D. and Demazeau Y. (2005), *Modèles et Réseaux de confiance*, in : Analyse bibliographique, Les cahiers Leibniz, N° 142, ISSN: 1298-020X

[37] Georgé G. P. (2004), *Résolution de problèmes par émergence : Étude d'un Environnement de Programmation Émergente*, Thèse doctorat, Université de Toulouse 3

[38] Ferber J. (1995), *Les systèmes multi-agents, vers une intelligence collective*, pp. 63-89, InterEdition

[39] Bouron M. T. (1992), *Structures de communication et d'organisation pour la coopération dans un univers multi-agents, une contribution a la définition d'un modèle de programmation agent base sur les concepts d'engagement et d'acte de langage*, Thèse doctorat, Université Paris 6

[40] Durfee E. H. Lesser V. R. and Corkill D. D. (1989), *Trends in cooperative distributed problem solving*, in: Knowledge and Data Engineering, pp. 63-83, IEEE, doi: 10.1109/69.43404

[41] Galliers J. R. (1988), *A Theoretical Framework for Computer Models of Cooperative Dialogue, Acknowledging Multi-agent Conflict*. Thèse de doctorat, Open University, UK, AAIDX87541

[42] Cohen P. R. Morgan J. and Pollack M. E. (2003), *Intentions in Communication*, in: System Development Foundation Benchmark Series, 520 pp. ISBN: 9780262517041, MIT Press. UK

[43] Bratman M. E, (1993), *What Is Intention ?*, in: intentions in communication, Philip R. Cohen, Jerry Morgan and Martha E. Pollack (eds.), System Development Foundation Benchmark Series, 520 pp. ISBN: 9780262517041, MIT Press, UK

[44] Drogoul A. and Ferber J. (1992), *Ethomodeling: a Multi-Agent Behavioral Simulation Model*, Rapport technique N° 02.92, LAFORIA, Université Paris 6

[45] Cohen P. R. and Levesque H. J. (1990), *Intention is choice with commitment*, in: Artificial Intelligence, vol. 42, pp 213-261, Elsevier Science Publishers B.V, North-Holland

[46] Davis R. and Smith R. G. (2003), *Negotiation as a metaphor for distributed problem solving*, Communication in Multi-agent Systems, vol. 2650 of the series Lecture Notes in Computer Science, pp 51-97, Springer Berlin Heidelberg, doi: 10.1007/978-3-540-44972-0_3

[47] Smith R. G. and Davis R. (1981), *Framework for cooperation in distributed problem solving*, in: International Conference on Systems, Man, and Cybernetics, 11(1): 61-70, IEEE, doi: 10.1109/TSMC.1981.4308579

[48] Conry S. E. Meyer, R. A. and Searleman J. T. (1985), *A Shared Knowledge Base for Independent Problem Solving Agents*, in: proc. of the Expert Systems in Government Symposium

[49] Erman L. D. and Lesser V. R. (1980), *Distributed interpretation: A model and experiment*, in: IEEE Transactions on Computers, c 29(12): 1144-1163, doi: 10.1109/TC.1980.1675519

[50] Hewitt C. and Kornfeld W. A. (1981), *The scientific community metaphor*, in: IEEE International Conference on Systems, Man and Cybernetics, 11(1): 24-33, doi: 10.1109/TSMC.1981.4308575

[51] Lesser V. R. and Corkill D. (1983), *The use of meta-level control for coordination in a distributed problem solving network*, in: proc. of International Joint Conference Artificial Intelligence, volume 8, pp. 748-756.

[52] Durfee E. H. and Lesser V. R. (1987), *Using partial global plans to coordinate distributed problem solvers*, in: proc. of the 10th international joint conference on Artificial, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 875-883.

[53] Ferber J. and Ghallab M. (1988), *Problématique des univers multi-agents intelligents*, in: proc. Journées nationales sur Intelligence Artificielle PRC-GRECO, pp. 295-320.

[54] Benmerzoug D, (2009), *Modèles et outils formels pour l'intégration d'applications d'entreprises*, Thèse en co-tutelle de l'Université Pierre et Marie Curie de Paris 6 et de l'Université Mentouri de Constantine

[55] Mazouzi H. (2001), *Ingénierie des protocoles d'interaction : des systèmes distribues aux systèmes multi-agents*, Thèse de doctorat de l'université paris 9 Dauphine, France

[56] Morge M. (2005), *Système dialectique multi-agents pour l'aide a la concertation*, Thèse de docteur de l'école Nationale Supérieure des Mines de Saint Etienne et de l'Université Jean Monnet

[57] Parunak H. V, (1996), *Visualizing agent conversations: using enhanced dooley graphs for agent design and analysis*, in: proc. of second international conference on multi-agent systems

[58] Populaire P. Demazeau Y. Boissier O. and Sichman L. (1993), *Description et implémentation de protocoles de communication en univers multi-agents*, in: proc. 1res journées francophones IAD et SMA, pp. 242-252, Toulouse

[59] Haddadi A. (1996), *Communication and cooperation in agent systems: a pragmatic theory*, in: Lecture Notes in Artificial Intelligence, Springer-Verlag Berlin Heidelberg, doi: 10.1007/3-540-61044-8

[60] Petri C. A. (1962), *Communication mit Automate*. Thèse de doctorat, Université technologique de Darmstadt, Allemand

[61] Woodcock P. and Davies J. (1996), *Using Z: Specification, Refinement, and Proof*, in: Prentice-Hall International Series in Computer Science, 386 pp, ISBN: 0139484728, Prentice Hall

[62] Spivey J. M. (1989), *the Z Notation: a Reference Manual*, 155 pp. ISBN: 0-13-983768-X, Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[63] ITU-T, (2002), *Specification and Description Language (SDL)*, in: ITU-T Recommendation Z.100, available at: https://www.itu.int

[64] Lakhrissi Y. (2010), *Intégration de la modélisation comportementale dans la conception par points de vue*. Thèse en co-tutelle de l'Université de Toulouse et de l'Université Mohammed V-Agdal de Rabat. Maroc

[65] Courtiat J. P. (1987), *Contribution à la description formelle de protocoles*. Thèse de doctorat, Université Paul Sabatier, Toulouse

[66] Garavel H, (1989), *Compilation et vérification de programmes LOTOS*. Thèse de doctorat, Université Josef Fourier- Grenoble I

[67] Brun P, (1998), *XTL: une logique temporelle pour la spécification formelle des systèmes interactifs*, Thèse de doctorat, Université Paris XI Orsay

[68] Wooldridge M. and Fisher M. (1994), *a decision procedure for temporal belief logic*, in: proc. of the First International Conference on Temporal Logic (ICTL '94), Dov M. Gabbay and Hans Jürgen Ohlbach (Eds.). Springer-Verlag, London, UK, 317-331.

[69] Dignum F. (1997), *Social Interactions of Autonomous Agents: Private and Global Views on Communication*, in: Formal Models of Agents, Lecture Notes in Computer Science, Springer International Publishing AG, 103–122

[70] Fisher M. (1994), *A Survey of Concurrent METATEM - the Language and its Applications*, in: proc. of the First International Conference on Temporal Logic (ICTL '94), Dov M. Gabbay and Hans Jürgen Ohlbach (Eds.), 480-505, Springer-Verlag, London, UK

[71] Camps V. (1998), *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, Université Paul Sabatier de Toulouse

[72] Lesser V. (1999), *Cooperative Multi agent Systems: A Personal View of the State of the Art*, in: IEEE Transactions on knowledge and data engineering, 11(1):133-142, doi: 10.1109/69.755622

[73] Abchiche N. Collinot A. David J. M. (1992), *Modelling Cooperation Reasoning*, in proc. of the Workshop on cooperation among heterogeneous intelligent agents, San Francisco, USA, AAAI-92, p. 2-10

[74] Abchiche-Mimouni N. (2012), *Adaptive organization for cooperative systems*, in: Frontiers in Artificial Intelligence and Applications, 243:128-138, doi: 10.3233/978-1-61499-105-2-128

[75] Boussebough I. Maamri R. and Sahnoun Z. (2010), *Adaptive multi-agent system: Cooperation and Structures Emergence*, in: Journal of software, 5(10):1170-1177, doi: 10.4304/jsw.5.10.1170-1177

[76] Dutta P. S. Jennings N. R. and Moreau L. (2006), *Adaptive distributed resource allocation and diagnostics using cooperative information-sharing strategies*, in: proc. of the fifth international joint conference on Autonomous agents and multi-agent systems (AAMAS '06), ACM, New York, NY, USA, 826-833, doi: 10.1145/1160633.1160783

[77] Prasad M. V. Lander S. E. and Lesser V. R, (1996), *Cooperative Learning over Composite Search Spaces: Experiences with a Multi-agent Design System*, in: proc. of the Thirteenth National Conference on Artificial Intelligence, vol. 1, pp. 68 - 73, AAAI Press / MIT Press, Portland, Oregon, USA

[78] Simonin O. and Ferber J. (2001), *Modélisation des satisfactions personnelle et interactive d'agents situés coopératifs*, in: proc. JFIADSMA'2001 – 9ième Journées Francophones d'Intelligence Artificielle Distribuée et Systèmes Multi-Agents, Montréal, Canada, pp. 215-226

[79] Ebadi T. Purvis M. and Purvis M. (2009), *A framework for facilitating cooperation in multi-agent systems*, in: Journal of Supercomputing, 51(3):393-417, doi: 10.1007/s11227-009-0372-8

[80] Ebadi T. (2012), *Facilitating Cooperation in Multi-agent Robotic Systems*, Thèse de doctorat, University of Otago, Dunedin, New Zealand

[81] Jmaiel M. and Hadj Kacem A. (2000), *a formal definition of cooperation and agency among multi-agent systems*, in: proc. of the International Conference on Artificial and Computational Intelligence for Decision, Control and Automation in Engineering and Industrial Applications, Monastir, Tunisia

[82] Kota R. Gibbins N. and Jennings N. R. (2008), *Decentralized Structured Adaptation in Agent Organizations*, in: proc. of First International Workshop on Organized Adaption in Multi-Agent Systems, Estoril, Portugal, May 13, 54-71, doi: 10.1007/978-3-642-02377-4_4

[83] Kobayashi K. Kurano T. Kuremoto T. and Obayashi M. (2012), *Cooperative behavior acquisition in multi agent reinforcement learning system using attention degree*, in: Lecture Notes in Computer Science, vol. 7665, 537-544, doi: 10.1007/978-3-642-34487-9_65

[84] Lee M. Lee J. Jeong H. Lee Y. Choi S. and Gatton T. M. (2006), *A Cooperation Model Using Reinforcement Learning for Multi-agent*, in proc. of International Conference on Computational Science and Its Applications, Glasgow, UK, May 8-11, 675-681, doi: 10.1007/11751649_74

[85] Plaza E. and Ontañón S. (2003), *Cooperative multi-agent learning*, in: Lecture Notes in Computer Science, vol. 2636, 1-17, doi: 10.1007/3-540-44826-8_1

[86] Bousbia N. and Balla A. (2007). *Processus de modélisation de contenus pédagogiques destinés à la FOAD*, in : e-TI - la revue électronique des technologies d'information, 3, ISSN 1114-8802, available at: http://www.revue-eti.net/

[87] Merzougui G. (2013), *Système auteur pour la création et la gestion de contenu pédagogique multimédia*, Thèse de doctorat de l'Université de Batna

[88] Psyché V. Olavo and Bourdeau J. (2003), *Apport de l'ingénierie ontologique aux environnements de formation à distance*, in : revue sticef.org : science et technologie de l'information et de la communication pour l'éducation et la formation, vol. 10

[89] Murray T. Blessing S. and Ainsworth S. (eds.) (2003), *Authoring Tools for Advanced Technology Learning Environments: Toward Cost-Effective Adaptive*, in: Interactive and Intelligent Educational Software, ISBN: 978-90-481-6499-8, Springer Netherlands, doi: 10.1007/978-94-017-0819-7

[90] LCMS (2003), *Etude des outils de gestion de ressources numériques pour l'enseignement*, Ministère de la Jeunesse, de l'Education Nationale et de la Recherche (France), Pôle conseil Business Interactif.

[91] Kuang M. B. (2000), *Robustesse des systèmes auteurs multimédias : contribution théorique et mise en œuvre*, Thèse de doctorat, Université de Paris 8

[92] Delestre N. (2000), *Metadyne : un hypermédia adaptatif dynamique pour l'enseignement, Thèse de Doctorat*, Université de Rouen

[93] Bachir S, (2013), *Une approche formelle pour la négociation automatique d'un e-commerce à base d'agent*, Thèse de doctorat, Université de Biskra

[94] Bellifemine F. L. Caire G. and Greenwood D. (2007), *Developing Multi-Agent Systems with JADE*, 300 pp. ISBN: 978-0-470-05747-6, Wiley

* * * * * * *

Résumé :

Ces dernières années, le développement des applications qui supportent les activités coopératives a attiré l'attention des chercheurs. Ainsi, plusieurs approches et méthodes ont été proposées pour atteindre cet objectif. Dans ce contexte, nous proposons une nouvelle solution pour la problématique des travaux collectifs, qui est le résultat de la combinaison de plusieurs approches de coopération existantes.

Au centre de notre approche un Framework de coopération tolérant aux fautes appelé *CoMAS* (Coopérative Multi-Agent System) qui permet en utilisant la préemption des ressources aux agents coopérants, en cas de défaillance, de maintenir le service et assurer la survie du système. Nous étudions le cas des systèmes auteurs coopératifs en e-Learning pour examiner le comportement du notre Framework face à une activité coopérative. La solution que nous proposons trouve application, surtout avec les activités coopératives ; C'est une solution à la fois générique et extensible pour tout le processus de coopération.

Mots-clés : Coopération; Système Multi-Agent; Résolution de Conflit; Compétence; Temps de coopération; Système Auteur; e-Learning.

ملخــص:

إن الأنظمة الداعمة للأعمال التعاضدية أثارت اهتمام الباحثين في الآونة الأخيرة مما أدى إلى ظهور مجموعة من المناهج والطرق لغرض تحسين نوعية الخدمة المقدمة من طرف هذه الأنظمة؛ في هذا الإطار، قمنا بتطوير مقاربة جديدة لحل مشكل العمل الجماعي ترتكز على تهجين مجموعة من المناهج، وهي كالتالي: المنهج الغير تحاوري لهدف تقليص الوقت اللازم لإتمام عمل جماعي ومنهج حل النزاعات المعتمد على الكفاءات لتحسين نوعية النتائج المحصل عليها.

نتيجة لهذا التصور، قمنا بتطوير إطار للعمل الجماعي متسامح مع العطب يقوم بسحب الموارد من الأعوان المشاركين في حالة تلفهم من أجل ضمان استمرارية النظام؛ بالإضافة إلى قيامنا بدراسة حالة الأنظمة التعاضدية للتأليف في التعليم عن بعد لأجل تقييم نوعية الخدمة المقدمة من طرف منهجنا في الأعمال التعاضدية. في الختام نقول أن مقاربتنا تمثل حلا عاما غير متعلقة بميدان خاص قابلة للتمديد وتغطي جميع أطوار العملية الجماعية.

الكلمات المفاتيح: نظام متعدد أعوان تعاضدي؛ حل النزاعات؛ الكفاءة؛ مدة التعاضد؛ نظام التأليف؛ التعليم عن بعد.

Abstract:

Nowadays, development of application that supports cooperative works has attracted recent attention. Thus, several approaches and methods were proposed to reach this aim. In this context, we propose a new solution to the problem of collective work that presents a result of hybridization of several cooperative approaches, as follows: the non-communicative approach to minimizing the required time to perform a cooperative activity and conflict resolution approach based competency for improving the quality of cooperative works.

Therefore, we propose a cooperation framework fault tolerant called *CoMAS* (Cooperative Multi-Agent System) brings preemption resources to cooperating agents in case of failure to prevent the blocking and ensure the survival of the system. Moreover, we study the case of cooperative authoring systems for e-learning to examine the behavior of our framework with a cooperative activity. Finally, our cooperation approach provides a generic and extensible solution that covers the whole cycle of a cooperation process.

Keywords: Cooperation; Multi-Agent System; Resolution of Conflict; Competence; Cooperation Time; Authoring System; e-Learning.